

IFIP AICT 376

Dimitris Gritzalis
Steven Furnell
Marianthi Theoharidou
(Eds.)

Information Security and Privacy Research

27th IFIP TC 11 International Information Security
and Privacy Conference, SEC 2012
Heraklion, Crete, Greece, June 2012
Proceedings

 Springer

Editor-in-Chief

A. Joe Turner, Seneca, SC, USA

Editorial Board

Foundations of Computer Science

Mike Hinchey, Lero, Limerick, Ireland

Software: Theory and Practice

Michael Goedicke, University of Duisburg-Essen, Germany

Education

Arthur Tatnall, Victoria University, Melbourne, Australia

Information Technology Applications

Ronald Waxman, EDA Standards Consulting, Beachwood, OH, USA

Communication Systems

Guy Leduc, Université de Liège, Belgium

System Modeling and Optimization

Jacques Henry, Université de Bordeaux, France

Information Systems

Jan Pries-Heje, Roskilde University, Denmark

ICT and Society

Jackie Phahlamohlaka, CSIR, Pretoria, South Africa

Computer Systems Technology

Paolo Prinetto, Politecnico di Torino, Italy

Security and Privacy Protection in Information Processing Systems

Kai Rannenberg, Goethe University Frankfurt, Germany

Artificial Intelligence

Tharam Dillon, Curtin University, Bentley, Australia

Human-Computer Interaction

Annelise Mark Pejtersen, Center of Cognitive Systems Engineering, Denmark

Entertainment Computing

Ryohei Nakatsu, National University of Singapore

IFIP – The International Federation for Information Processing

IFIP was founded in 1960 under the auspices of UNESCO, following the First World Computer Congress held in Paris the previous year. An umbrella organization for societies working in information processing, IFIP's aim is two-fold: to support information processing within its member countries and to encourage technology transfer to developing nations. As its mission statement clearly states,

IFIP's mission is to be the leading, truly international, apolitical organization which encourages and assists in the development, exploitation and application of information technology for the benefit of all people.

IFIP is a non-profitmaking organization, run almost solely by 2500 volunteers. It operates through a number of technical committees, which organize events and publications. IFIP's events range from an international congress to local seminars, but the most important are:

- The IFIP World Computer Congress, held every second year;
- Open conferences;
- Working conferences.

The flagship event is the IFIP World Computer Congress, at which both invited and contributed papers are presented. Contributed papers are rigorously refereed and the rejection rate is high.

As with the Congress, participation in the open conferences is open to all and papers may be invited or submitted. Again, submitted papers are stringently refereed.

The working conferences are structured differently. They are usually run by a working group and attendance is small and by invitation only. Their purpose is to create an atmosphere conducive to innovation and development. Refereeing is less rigorous and papers are subjected to extensive group discussion.

Publications arising from IFIP events vary. The papers presented at the IFIP World Computer Congress and at open conferences are published as conference proceedings, while the results of the working conferences are often published as collections of selected and edited papers.

Any national society whose primary activity is in information may apply to become a full member of IFIP, although full membership is restricted to one society per country. Full members are entitled to vote at the annual General Assembly. National societies preferring a less committed involvement may apply for associate or corresponding membership. Associate members enjoy the same benefits as full members, but without voting rights. Corresponding members are not represented in IFIP bodies. Affiliated membership is open to non-national societies, and individual and honorary membership schemes are also offered.

Dimitris Gritzalis Steven Furnell
Marianthi Theoharidou (Eds.)

Information Security and Privacy Research

27th IFIP TC 11 Information Security
and Privacy Conference, SEC 2012
Heraklion, Crete, Greece, June 4-6, 2012
Proceedings

Volume Editors

Dimitris Gritzalis

Marianthi Theoharidou

Athens University of Economics and Business, Department of Informatics
Information Security and Critical Infrastructure Protection Research Group
76 Patisision Ave., 10434 Athens, Greece

E-mail: {dgrit,mtheohar}@aueb.gr

Steven Furnell

University of Plymouth

School of Computing Communications and Electronics

A310, Portland Square, Drake Circus, Plymouth, PL4 8AA, UK

E-mail: s.furnell@plymouth.ac.uk

ISSN 1868-4238

ISBN 978-3-642-30435-4

DOI 10.1007/978-3-642-30436-1

Springer Heidelberg Dordrecht London New York

e-ISSN 1868-422X

e-ISBN 978-3-642-30436-1

Library of Congress Control Number: 2012937786

CR Subject Classification (1998): C.2, K.6.5, D.4.6, E.3, H.4, J.1

© IFIP International Federation for Information Processing 2012

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

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

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

It was an honor and a privilege to chair the 27th IFIP International Information Security Conference (SEC 2012), a 29-year old event that has already become a tradition for information security professionals around the world. SEC 2012 was organized by Technical Committee 11 (TC-11) of IFIP and this year took place on the Greek island of Crete, during June 4–6, 2012. As Program Committee Chairs, we were extremely pleased to serve a conference in a location with such natural beauty, and where the hospitality and friendliness of the people have been going together, hand-in-hand, with its very long history.

This volume contains the papers selected for presentation at SEC 2012, which proved to be a really competitive forum. After a first check by the PC chairs on meeting the submission criteria, scope and quality, 115 of the 167 initially submitted papers were then sent out for review by the conference Program Committee. All of these 115 papers were evaluated on the basis of their novelty and technical quality, and reviewed by at least two members of the program committee. Of these 115 papers, finally 42 were accepted as full papers. A further 11 submissions were accepted as short papers.

It is thanks to the commitment of several people that international conferences are able to happen. This holds true also for SEC 2012. The full list of individuals that volunteered their time and energy to help is really long, and we would like to express our sincere appreciation to the members of the Program Committee, to the external reviewers, and to the authors, who trusted their work in our hands. Many thanks go, also, to all conference attendees.

In particular, we would like to thank our distinguished keynote speaker, Udo Helmbrecht (ENISA) for accepting our invitation and honoring the conference with his presence and inspired talk. We also thank the local organizers and hosts, first among them being the Organizing Committee Chairs Marianthi Theoharidou and Nickolas Kyrloglou, as well as the Publicity Chair Sara Foresti, who took care of every detail to ensure that SEC 2012 would become a successful and memorable event. Our further appreciation also goes to Marianthi for strongly supporting us in the preparation and editing of this conference proceedings volume.

Finally, let us express a personal note. We would like to thank all TC-11 members for giving us the opportunity to serve SEC 2012 as Program Committee Chairs. It was the first time this opportunity is given to Steven Furnell. It was the fourth time, following events in Samos (SEC 1996), Athens (SEC 2003) and Pafos (SEC 2009), that this opportunity was given to Dimitris Gritzalis, who has thus further extended his record as a SEC conference chairing “dinosaur”.

Dimitris Gritzalis
Steven Furnell

Organization

Committees

General Chair

Sokratis Katsikas University of Piraeus, Greece

Program Chairs

Dimitris Gritzalis Athens University of Economics and Business,
Greece
Steven Furnell Plymouth University, (UK)

Organizing Committee Chairs

Marianthi Theoharidou Athens University of Economics and Business,
Greece
Nikolaos Kyrloglou Athens Chamber of Commerce and Industry,
Greece

Publicity Chair

Sara Foresti Università degli studi di Milano, Italy

Program Committee

Vijay Atluri Rutgers University, USA
Joachim Biskup University of Dortmund, Germany
Jan Camenisch IBM Research, Switzerland
Sabrina De Capitani
 di Vimercati Università degli studi di Milano, Italy
Nathan Clarke Plymouth University, UK
Jeff Crume IBM, USA
Frederic Cuppens TELECOM Bretagne, France
Nora Cuppens UEB, France
Bart De Decker K.U. Leuven, Belgium
Gurpreet Dhillon Virginia C/wealth University, USA
Theo Dimitrakos BT, UK
Ronald Dodge US Military Academy, USA
Simone Fischer-Huebner Karlstadt University, Sweden
Dieter Gollmann Hamburg University of Technology, Germany

VIII Organization

Jaap-Henk Hoepman	TNO and Radboud University Nijmegen, The Netherlands
Thorsten Holz	Ruhr University Bochum, Germany
Sotiris Ioannidis	FORTH, Greece
Bart Jacobs	Radboud University Nijmegen, The Netherlands
Sushil Jajodia	George Mason University, USA
Lech Janczewski	University of Auckland, New Zealand
Tom Karygiannis	NIST, USA
Vasilios Katos	University of Thrace, Greece
Panagiotis Katsaros	University of Thessaloniki, Greece
Stefan Katzenbeisser	T.U. Darmstadt, Germany
Dogan Kesdogan	University of Siegen, Germany
Costas Lambrinouidakis	University of Piraeus, Greece
Carl Landwehr	University of Maryland, USA
Ronald Leenes	Tilburg University, The Netherlands
Javier Lopez	University of Malaga, Spain
Evangelos Markatos	FORTH and University of Crete, Greece
Stephen Marsh	Communications Research Center, Canada
Ioannis Mavridis	University of Macedonia, Greece
Natalia Miloslavskaya	National Nuclear Research University, Russia
Yuko Murayama	Iwate Prefectural University, Japan
Eiji Okamoto	University of Tsukuba, Japan
Martin Olivier	University of Pretoria, South Africa
Evangelos Ouzounis	ENISA, (EU)
Jakob Illeborg Pagter	Alexandra Instituttet, Denmark
Maria Papadaki	Plymouth University, UK
Philippos Peleties	USB Bank, Cyprus
Sihan Qing	Chinese Academy of Sciences, China
Muttukrishnan Rajarajan	City University, UK
Kai Rannenber	Goethe University Frankfurt, Germany
Carlos Rieder	HSW Luzern, Switzerland
Pierangela Samarati	Università degli studi di Milano, Italy
Ryoichi Sasaki	Tokyo Denki University, Japan
Ingrid Schaumüller-Bichl	UAS Hagenberg, Austria
Anne Karen Seip	Finanstilsynet, Norway
Rossouw Von Solms	NMMU, South Africa
Theo Tryfonas	University of Bristol, UK
Craig Valli	Edith Cowan University, Australia
Jozef Vyskoc	VaF, Slovakia
Christian Weber	WebITsec, Germany
Tatjana Welzer	University of Maribor, Slovenia
Dirk Westhoff	HAW Hamburg, Germany
Louise Yngstrom	University of Stockholm, Sweden
Moti Yung	Google, USA

Additional Reviewers

Stelios Dritsas
Carmen Fernandez
Gerardo Fernandez
Antonios Gouglidis
Christian Kahl
Ella Kolkowska
Antonis Krithinakis
Meixing Le
Andreas Leicher
Dimitra Liveri
Milica Milutinovic
Wojciech Mostowski
Konstantinos Moulinos

David Nuñez
Antonis Papadagiannakis
Thanasis Petsas
Ahmad Sabouri
Bill Tsoumas
Fabian van den Broek
Giorgos Vasiliadis
Sicco Verwer
Fatbardh Veseli
Pim Vullers
Philipp Winter
Ge Zhang
Lei Zhang

Table of Contents

Attacks and Malicious Code

Relay Attacks on Secure Element-Enabled Mobile Devices: Virtual Pickpocketing Revisited	1
<i>Michael Roland, Josef Langer, and Josef Scharinger</i>	
Would You Mind Forking This Process? A Denial of Service Attack on Android (and Some Countermeasures)	13
<i>Alessandro Armando, Alessio Merlo, Mauro Migliardi, and Luca Verderame</i>	
An Approach to Detecting Inter-Session Data Flow Induced by Object Pooling	25
<i>Bernhard J. Berger and Karsten Sohr</i>	
Embedded Eavesdropping on Java Card	37
<i>Guillaume Barbu, Christophe Giraud, and Vincent Guerin</i>	

Security Architectures

Authenticated Key Exchange (AKE) in Delay Tolerant Networks	49
<i>Sofia Anna Menesidou and Vasilios Katos</i>	
OFELIA – A Secure Mobile Attribute Aggregation Infrastructure for User-Centric Identity Management	61
<i>Alexandre B. Augusto and Manuel Eduardo Correia</i>	
Smart OpenID: A Smart Card Based OpenID Protocol	75
<i>Andreas Leicher, Andreas U. Schmidt, and Yogendra Shah</i>	
Peer to Peer Botnet Detection Based on Flow Intervals	87
<i>David Zhao, Issa Traore, Ali Ghorbani, Bassam Sayed, Sherif Saad, and Wei Lu</i>	

System Security

Towards a Universal Data Provenance Framework Using Dynamic Instrumentation	103
<i>Eleni Gessiou, Vasilis Pappas, Elias Athanasopoulos, Angelos D. Keromytis, and Sotiris Ioannidis</i>	
Improving Flask Implementation Using Hardware Assisted In-VM Isolation	115
<i>Baozeng Ding, Fufeng Yao, Yanjun Wu, and Yeping He</i>	

HyperForce: Hypervisor-enForced Execution of Security-Critical Code 126
Francesco Gadaleta, Nick Nikiforakis, Jan Tobias Mühlberg, and Wouter Joosen

RandHyp: Preventing Attacks via Xen Hypercall Interface 138
Feifei Wang, Ping Chen, Bing Mao, and Li Xie

Access Control

Role Mining under Role-Usage Cardinality Constraint 150
John C. John, Shamik Sural, Vijayalakshmi Atluri, and Jaideep S. Vaidya

HIDE_DHCP: Covert Communications through Network Configuration Messages 162
Ruben Rios, Jose A. Onieva, and Javier Lopez

Handling Stateful Firewall Anomalies 174
Frédéric Cuppens, Nora Cuppens-Boulahia, Joaquín García-Alfaro, Tarik Moataz, and Xavier Rimasson

A Framework for Threat Assessment in Access Control Systems 187
Hemanth Khambhammettu, Sofiene Boulares, Kamel Adi, and Luigi Logrippo

Database Security

Support for Write Privileges on Outsourced Data 199
Sabrina De Capitani di Vimercati, Sara Foresti, Sushil Jajodia, Stefano Paraboschi, and Pierangela Samarati

Malicious Users’ Transactions: Tackling Insider Threat 211
Weihan Li, Brajendra Panda, and Qussai Yaseen

Privacy Attitudes and Properties

Privacy-Preserving Television Audience Measurement Using Smart TVs 223
George Drosatos, Aimilia Tasidou, and Pavlos S. Efraimidis

Tracking Users on the Internet with Behavioral Patterns: Evaluation of Its Practical Feasibility 235
Christian Banse, Dominik Herrmann, and Hannes Federrath

Smartphone Forensics: A Proactive Investigation Scheme for Evidence Acquisition	249
<i>Alexios Mylonas, Vasilis Meletiadis, Bill Tsoumas, Lilian Mitrou, and Dimitris Gritzalis</i>	

Social Networks and Social Engineering

Modeling Social Engineering Botnet Dynamics across Multiple Social Networks	261
<i>Shuhao Li, Xiaochun Yun, Zhiyu Hao, Yongzheng Zhang, Xiang Cui, and Yipeng Wang</i>	
Layered Analysis of Security Ceremonies	273
<i>Giampaolo Bella and Lizzie Coles-Kemp</i>	

Applied Cryptography, Anonymity and Trust

A Small Depth-16 Circuit for the AES S-Box	287
<i>Joan Boyar and René Peralta</i>	
Formal Verification of the mERA-Based eServices with Trusted Third Party Protocol	299
<i>Maria Christofi and Aline Gouget</i>	

Usable Security

My Authentication Album: Adaptive Images-Based Login Mechanism	315
<i>Amir Herzberg and Ronen Margulies</i>	
Balancing Security and Usability of Local Security Mechanisms for Mobile Devices	327
<i>Shuzhe Yang and Gökhan Bal</i>	
Analyzing Value Conflicts for a Work-Friendly ISS Policy Implementation	339
<i>Ella Kolkowska and Bart De Decker</i>	
When Convenience Trumps Security: Defining Objectives for Security and Usability of Systems	352
<i>Gurpreet Dhillon, Tiago Oliveira, Santa Susarapu, and Mário Caldeira</i>	

Security and Trust Models

Security-by-Contract for the OSGi Platform	364
<i>Olga Gadyatskaya, Fabio Massacci, and Anton Philippov</i>	

Cyber Weather Forecasting: Forecasting Unknown Internet Worms Using Randomness Analysis	376
<i>Hyundo Park, Sung-Oh David Jung, Heejo Lee, and Hoh Peter In</i>	
Incentive Compatible Moving Target Defense against VM-Colocation Attacks in Clouds	388
<i>Yulong Zhang, Min Li, Kun Bai, Meng Yu, and Wanyu Zang</i>	
Give Rookies a Chance: A Trust-Based Institutional Online Supplier Recommendation Framework	400
<i>Han Jiao, Jixue Liu, Jiuyong Li, and Chengfei Liu</i>	

Security Economics

A Game-Theoretic Formulation of Security Investment Decisions under Ex-ante Regulation	412
<i>Giuseppe D’Acquisto, Marta Flamini, and Maurizio Naldi</i>	
Optimizing Network Patching Policy Decisions	424
<i>Yolanta Beres and Jonathan Griffin</i>	
A Risk Assessment Method for Smartphones	443
<i>Marianthi Theoharidou, Alexios Mylonas, and Dimitris Gritzalis</i>	
Empirical Benefits of Training to Phishing Susceptibility	457
<i>Ronald Dodge, Kathryn Coronges, and Ericka Rovira</i>	

Authentication and Delegation

Multi-modal Behavioural Biometric Authentication for Mobile Devices	465
<i>Hataichanok Saevanee, Nathan L. Clarke, and Steven M. Furnell</i>	
Analysis and Modeling of False Synchronizations in 3G-WLAN Integrated Networks	475
<i>Christoforos Ntantogian, Christos Xenakis, and Ioannis Stavrakakis</i>	
Password Protected Smart Card and Memory Stick Authentication against Off-Line Dictionary Attacks	489
<i>Yongge Wang</i>	
Distributed Path Authentication for Dynamic RFID-Enabled Supply Chains	501
<i>Shaoying Cai, Yingjiu Li, and Yunlei Zhao</i>	
Enhanced Dictionary Based Rainbow Table	513
<i>Vrizlynn L.L. Thing and Hwei-Ming Ying</i>	

Short Papers

Authorization Policies for Materialized Views	525
<i>Sarah Nait-Bahloul, Emmanuel Coquery, and Mohand-Saïd Hacid</i>	
Enhancing the Security of On-line Transactions with CAPTCHA Keyboard	531
<i>Yongdong Wu and Zhigang Zhao</i>	
Fighting Pollution Attack in Peer-to-Peer Streaming Networks: A Trust Management Approach	537
<i>Xin Kang and Yongdong Wu</i>	
A Framework for Anonymizing GSM Calls over a Smartphone VoIP Network	543
<i>Ioannis Psaroudakis, Vasilios Katos, and Pavlos S. Efraimidis</i>	
A Browser-Based Distributed System for the Detection of HTTPS Stripping Attacks against Web Pages	549
<i>Marco Prandini and Marco Ramilli</i>	
Privacy-Preserving Mechanisms for Organizing Tasks in a Pervasive eHealth System	555
<i>Milica Milutinovic, Vincent Naessens, and Bart De Decker</i>	
Web Services Security Assessment: An Authentication-Focused Approach	561
<i>Yannis Soupionis and Miltiadis Kandias</i>	
Open Issues and Proposals in the IT Security Management of Commercial Ports: The S-PORT National Case	567
<i>Nineta Polemi and Theodoros Ntouskas</i>	
A Response Strategy Model for Intrusion Response Systems	573
<i>Nor Badrul Anuar, Maria Papadaki, Steven Furnell, and Nathan Clarke</i>	
Intrusion Tolerance of Stealth DoS Attacks to Web Services	579
<i>Massimo Ficco and Massimiliano Rak</i>	
Towards Use-Based Usage Control	585
<i>Christos Grompanopoulos and Ioannis Mavridis</i>	
Author Index	591

Relay Attacks on Secure Element-Enabled Mobile Devices^{*}

Virtual Pickpocketing Revisited

Michael Roland¹, Josef Langer¹, and Josef Scharinger²

¹ NFC Research Lab Hagenberg, University of Applied Sciences Upper Austria
{michael.roland,josef.langer}@fh-hagenberg.at

² Department of Computational Perception, Johannes Kepler University Linz
josef.scharinger@jku.at

Abstract. Near Field Communication’s card emulation mode is a way to combine smartcards with a mobile phone. Relay attack scenarios are well-known for contactless smartcards. In the past, relay attacks have only been considered for the case, where an attacker has physical proximity to an NFC-enabled mobile phone. However, a mobile phone introduces a significantly different threat vector. A mobile phone’s permanent connectivity to a global network and the possibility to install arbitrary applications permit a significantly improved relay scenario. This paper presents a relay attack scenario where the attacker no longer needs physical proximity to the phone. Instead, simple relay software needs to be distributed to victims’ mobile devices. This publication describes this relay attack scenario in detail and assesses its feasibility based on measurement results.

1 Introduction

Near Field Communication (NFC) is an advancement of inductively coupled proximity Radio Frequency Identification (RFID) technology and smartcard technology. NFC has three operating modes: peer-to-peer mode, reader/writer mode and card emulation mode. Peer-to-peer mode is an operating mode specific to NFC and allows two NFC devices to communicate directly with each other. In reader/writer mode, NFC devices can access contactless smartcards, RFID transponders and NFC tags. In card emulation mode, an NFC device emulates a contactless smartcard and, thus, is able to communicate with existing RFID readers.

Three communication standards are available for card emulation mode: ISO/IEC 14443 Type A, Type B and FeliCa (JIS X 6319-4). Which mode is actually used depends on the NFC chipset and the geographic region. With current NFC-enabled mobile phones, card emulation is usually based on Type A.

^{*} This work is part of the project “4EMOBILITY” within the EU programme “Regionale Wettbewerbsfähigkeit OÖ 2007–2013 (Regio 13)” funded by the European regional development fund (ERDF) and the Province of Upper Austria (Land Oberösterreich).

Besides the communication standard, card emulation may also vary in the way how card emulation is performed. On the one hand, a card can be emulated in software (e.g. on the device's application processor). On the other hand, card emulation can be performed by a dedicated smartcard chip – a so-called secure element (SE). Such a chip can be a dedicated SE IC (integrated circuit) that is embedded into the NFC device. Another possibility is the combination of the SE functionality with another smartcard/security device that is used within the NFC device – like a UICC (universal integrated circuit card; often referred to as Subscriber Identity Module/SIM card) or an SD (secure digital) memory card.

Typical use-cases for card emulation are security critical applications such as access control and payment. Therefore, emulation by software on a non-secure application processor is not widely used. As of today, only some dedicated NFC reader devices – like ACS's ACR 122U – and only a small number of NFC-enabled mobile phones – specifically those equipped with RIM's BlackBerry 7 operating system [1,12] – support software card emulation.

The majority of mobile NFC devices use dedicated smartcard chips for card emulation. Examples are the Nokia 6131, the Nokia 6212, the Samsung GT-S5230N (“Player One”) and the Samsung Nexus S. The Nokia 6131 and the Nokia 6212 have an embedded SE. The Samsung GT-S5230N uses an SWP-enabled (Single Wire Protocol) UICC as SE and the Samsung Nexus S has both, an embedded SE and support for an SWP-enabled UICC. Only some recent NFC phones developed by Nokia have no support for card emulation at all [1].

Typical secure elements are standard smartcard ICs as used for regular contact and contactless smartcards. The only difference is the interface they provide: Instead (or in addition) to a classic smartcard interface, the secure element has a direct interface¹ for the connection to the NFC controller.

As secure elements have the same hardware and software platforms as regular smartcards, they also share the same security standards. A secure element provides secure storage, a secure execution environment and hardware-based support for cryptographic operations. The IC is protected against various attacks that aim at retrieval or manipulation of stored data and processed operations. Smartcard chips and their design process are usually evaluated and certified according to high security standards. The same applies to their operating systems. Thus, the secure element fulfills the requirements necessary for security critical applications like access control and even payment.

While smartcards by themselves are considered safe and secure, especially contactless cards are vulnerable to relay attacks. This issue is well-known (see [6,7,10]) but can be circumvented by isolating the card from the surrounding world when it is not in use. However, the integration of smartcard functionality into NFC-enabled mobile phones introduces a new and unexplored attack vector. For example, a fundamental difference between a regular smartcard and an NFC-enabled mobile phone is the network connectivity: A smartcard can easily be isolated from the surrounding world by means of shielding. In contrast to that, a mobile phone and its emulated smartcard are permanently connected to a global

¹ E.g. NFC Wired Interface (NFC-WI) or Single Wire Protocol (SWP).

network (cf. intrusion paths identified by Jeon et al. [9]). A further problem arises from the possibility to install arbitrary – possibly untrusted – applications on current mobile phones (specifically smart phones).

In this publication we investigate the potential of these new and unexplored weaknesses. We focus on a relay attack scenario which could be abused to remotely use a victim’s emulated smartcard without the victim’s knowledge. A system for relay attacks over the internet is introduced. Measurement results of the delays induced by this relay system are provided to verify this relay system. Finally, we evaluate the feasibility of relay attacks based on these measurement results.

1.1 Smartcard Communication

Low-level communication protocols for smartcards depend on the communication interface and are either character-based or frame-based. For contactless smartcards a frame-based protocol is standardized in ISO/IEC 14443. Application level protocols attach on top of these low-level protocols.

An application level communication protocol for smartcards is defined in ISO/IEC 7816-4. This protocol applies to both, contact and contactless smartcards. Command-and-response pairs are called APDUs (application protocol data units). Commands are always sent from the reader to the card while responses are always sent from the card to the reader.

1.2 Android’s Secure Element API

While Android-based NFC-enabled mobile phones, like the Nexus S or the Galaxy Nexus, have an embedded secure element and also support UICC-based secure elements, there currently is no public API to access the secure element on the Android platform.

However, Google has already introduced its Google Wallet, which is available for the Nexus S in certain regions. For this wallet to work, Google secretly integrated an API called `com.android.nfc.extras` into their Android platform. This API can be used to access the embedded secure element and is available since Android 2.3.4. Yet, this interface is not included in the public software development kit (SDK) and, thus, is hidden from the average programmer.

The secure element API consists of two classes: `NfcAdapterExtras` and `NfcExecutionEnvironment`. `NfcAdapterExtras` is used to enable and disable external card emulation (`setCardEmulationRoute()`) and to retrieve an instance of the secure element’s `NfcExecutionEnvironment` class (`getEmbeddedExecutionEnvironment()`). `NfcExecutionEnvironment` is used to exchange APDUs with the embedded secure element. This class provides methods to open and close the internal connection to the secure element (`open()`, `close()`) and to exchange APDU sequences with the secure element (`transceive()`).

In Android 2.3.4 this secure element API could be accessed by any application that held the permission to use NFC. In later versions this has been changed to a special permission named `com.android.nfc.permission.NFCEE_ADMIN`. This

special permission is only granted to applications which are signed with the same certificate as the NFC system service. Consequently, access to the secure element is restricted to applications trusted by the manufacturer/provider of the NFC system service.

2 Related Work

In [13], we evaluate APIs for SE access on various platforms. The level of protection for such APIs varies widely. A major weakness, that all access control schemes for SE APIs have in common, is that they all require the operating system and the mobile device hardware to be trusted. For instance, an attacker who has control over the operating system or the device hardware can either bypass security measures of the SE API or even bypass the whole SE API.

We further introduce two new attack scenarios against secure element-enabled mobile phones [13]:

- A denial of service (DoS) attack that can be abused to permanently lock an embedded SE and, consequently, render an NFC-enabled mobile phone unusable for card emulation applications.
- A relay attack that can be abused to access a SE from anywhere over an Internet connection.

Both attack scenarios can be applied to several existing NFC-enabled mobile phones, e.g. Nokia 6131, Nokia 6212 and Samsung Nexus S.

Hancke [6] first presented a successful relay attack against ISO/IEC 14443 smartcards in 2005. Compared to that scenario, where the demodulated RF signals are transmitted over an alternative carrier, the relay attack introduced in [13] relays application level commands (APDUs). This is necessary as SE APIs typically provide an APDU-based interface.

Application layer security protocols cannot prevent relay attacks (cf. *Mafia fraud* by Desmedt et al. [2] and conclusions about RFID and NFC protocols by Hancke [6], Kfir and Wool [10] and Francis et al. [4]). The reason is that the relay can be seen as an elongation of the communication medium. Messages are transferred through the relay channel without any modification. Thus, the only measureable difference is the propagation delay through the relay channel.

3 Attacking Mobile Phones

The application programming interface and the resources of mobile devices are usually restricted by access control policies. Access to critical parts of the system is typically shielded from untrusted applications. However, these are usually software-based restrictions enforced by the operating system. Thus, if an untrusted application can manipulate the behavior of the operating system or if it can elevate its privileges to the level of a trusted application, it can easily circumvent any access restrictions.

For many mobile phone platforms there exist methods to circumvent security measures. Popular techniques used on many smart phones are “jail breaking” and “rooting”. Jail breaking refers to escaping the restrictions imposed by the operating system, so that an application can access resources it usually could not access. Rooting refers to an even sever scenario where the user or an application gains full access to the whole system. Both methods are often used intentionally by device owners/legitimate users to circumvent digital rights management or to gain “improved” control over their device.

However, intentional jail breaking and rooting imposes a significant security risk. Not only the legitimate user gains access to – otherwise restricted – resources but also an attacker gets these same possibilities. Thus, a jail broken or rooted phone is significantly more vulnerable to attacks.

On the one hand, rooting can be done by using vendor-supplied methods. Such methods typically exist for development phones (e.g. for the Google Nexus series of Android smart phones). They are usually implemented in a safe way that protects the user from malicious activities. For example, rooting a Google Nexus phone according to the official instructions will wipe all data on the phone. Thus, this method of rooting cannot be used to gain access to sensitive user data that resides on the device.

On the other hand, jail breaking and rooting can be done by exploiting vulnerabilities in software or hardware. These exploits are not only viable for intentional jail breaking and rooting by the device owners/legitimate users. The same exploits can be integrated in virtually any application. That way a malicious application could elevate its permissions without the (legitimate) user’s knowledge.

Lately the topic of mobile phone security experiences significantly increasing awareness. Recent research activities include the assessment of vulnerabilities and threats and the uncovering of actual attack scenarios. According to Kaspersky Lab’s monthly malware statistics [5] the trend towards threats and malware for Android (and mobile platforms in general) has dramatically increased within the last year. Therefore, it seems unlikely that this trend will be interrupted any time soon.

This trend is confirmed by the most recent exploits for the Android platform:

- Levitator², published in Oct. 2011, works up to Android 2.3.5,
- zergRush³, published in Oct. 2011, works up to Android 2.3.3,
- GingerBreak⁴, published in Apr. 2011, works up to Android 2.3.3,
- ZimperLich, KillingInTheName, RageAgainstTheCage, Exploid and others for earlier versions.

Soon after a vulnerability gets fixed, a new exploit is published. If this trend continues, it’s only a matter of time until exploits for the most recent versions of the Android platform become available.

² <http://jon.oberheide.org/files/levitator.c>

³ <http://forum.xda-developers.com/showthread.php?t=1296916>

⁴ <http://c-skills.blogspot.com/2011/04/yummy-yummy-gingerbreak.html>

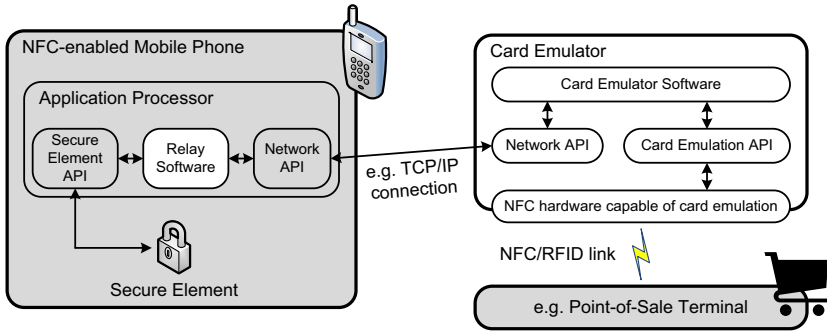


Fig. 1. Relay scenario: Relay software is installed on the victim’s phone. The software relays APDUs between the secure element and the card emulator across a network (cellular network, WiFi, Bluetooth...) The card emulator emulates a contactless smartcard that interacts with a card reader (point-of-sale terminal, access control reader...) The card emulator routes all APDU commands received from the point-of-sale terminal through the network interface to the relay software on the victim’s mobile phone. As soon as the response APDU is received from the relay software, it is forwarded to the reader.

As Höbarth and Mayrhofer [8] show, it is even possible to create frameworks for permanent on-device privilege escalation. Such a framework would use the most recent exploits for a certain platform to gain temporary super-user privileges. These elevated privileges would then be used to permanently root the device. Such a framework can be integrated by an attacker into any malicious application.

4 Relay Attack on the Secure Element

In [13], we initially proposed a relay scenario that allows an attacker to remotely use a victim’s secure element over a network connection. At, for instance, a point-of-sale or an access control gate, nobody would suspect that the communication is actually relayed to a remote device.

The scenario of the relay attack is shown in Fig. 1. It consists of four parts:

- a mobile phone (under control of its owner/legitimate user),
- a relay software (under control of the attacker),
- a card emulator (under control of the attacker), and
- a reader device (e.g. at a point-of-sale terminal or at an access control gate).

The relay software is installed on the victim’s mobile phone. This application is assumed to have the privileges necessary for access to the secure element and for communicating over a network. These privileges can be either explicitly granted to the application or acquired by means of a privilege escalation attack. The

relay application waits for APDU commands on a network socket and forwards these APDUs to the secure element. The responses are then sent back through the network socket.

The card emulator is a device that is capable of emulating a contactless smart-card in software. The emulator has RFID/NFC hardware that acts as a contactless smartcard when put in front of a smartcard reader. The emulator software forwards the APDU commands (and responses) between a network socket and the emulator's RFID/NFC hardware.

There are several different options when choosing a device for card emulation:

- *Building a new device from scratch:* This method gives full control over the whole design process. The card emulator can be put into any inconspicuous looking shape. The whole RFID protocol stack can be controlled starting from the lowest layer. Thus, even the protocol levels below APDU communication can be influenced. For instance the unique identifier (UID) that is used during the anti-collision sequence can be freely chosen. However, building a new emulator device from scratch also involves the highest design costs and effort.
- *Using a ready-made RFID card emulation device:* Card emulation devices – like the IAIK HF DemoTag⁵ or the Proxmark⁶ – already provide the hardware platform and a rudimentary software stack for card emulation. With this choice, it is still possible to control the whole RFID protocol stack. However, the ready-made devices cannot easily be fit into any desired shape.
- *Using an NFC reader:* Some NFC reader devices – like the ACS ACR 122U – can be put into software card emulation mode. In this mode, the device waits for APDU commands from an external reader device and forwards them to the computer. The computer then generates a response that is forwarded to the reader. The lower protocol layers are handled automatically by the reader firmware. One disadvantage of this approach is that typical NFC chipsets do not allow the user to freely choose all parameters for the lower protocol layers. For example, with the ACR 122U, the unique identifier for the anti-collision procedure can only start with '08', which denotes a random UID. Another disadvantage is that this device can only emulate the ISO/IEC 14443 Type A communication protocol.
- *Using an NFC-enabled mobile phone:* Another alternative is the use of NFC-enabled mobile phones as software card emulation devices. Currently, only RIM's BlackBerry mobile phones are known to support software card emulation. Yet, other mobile phones' firmware could possibly be adapted to support software card emulation as well. Using a mobile phone as card emulation device has several advantages. First, the mobile phone already has the form-factor that is expected for NFC contactless transactions (i.e. as the device that actually carries the secure element). Second, the mobile phone has a network interface that can be used to connect to the relay software.

⁵ <http://jce.iaik.tugraz.at/Products/RFID-Components/HF-RFID-Demo-Tag>

⁶ <http://www.proxmark.org/>

Third, the mobile phone has all the processing capabilities to transfer APDU commands between its network interface and its card emulation hardware. The API documentation for the BlackBerry software card emulation API [12] suggests, that both, ISO/IEC 14443 Type A and Type B, communication protocols can be emulated and, that even low-level parameters – like the UID – can be freely chosen.

4.1 Limitations by the Communication Protocol

Hancke et al. [7] conclude that the timing constraints of ISO/IEC 14443 are too loose to provide adequate protection against relay attacks.

ISO/IEC 14443 specifies certain delays and timeouts. First, there is the *frame delay time* (ISO/IEC 14443-3 Type A) between commands sent by the reader and responses sent by the card. For commands that are used during anti-collision, the *frame delay time* defines a strict timing between requests and responses. This is necessary to detect collisions during the anti-collision procedure. For all other commands, the *frame delay time* specifies only a minimum delay. As our relay system operates on the APDU layer, these delays are handled by the card emulator and do not apply to the relay path.

Second, there is the *frame waiting time* (ISO/IEC 14443-4). The *frame waiting time* specifies the maximum timeout between a command frame sent by the reader and the response received from the card. This timeout is defined by the card and can range between about 302 us and 4949 ms. The timeout can be extended on a per-command basis by the card using *frame waiting time extension* commands. Thus, this timeout does not affect the APDU layer. The timeout extension can be handled by the card emulator and does not apply to the relay path. Even if *frame waiting time extension* would not be used, relaying APDUs may take almost 5 seconds without violating this timeout.

4.2 Implementation

We implemented a proof-of-concept of the relay system to verify our assumptions. Our relay system (see Fig. 2) consists of the following parts:

- Samsung Nexus S with Android 2.3.4,
- relay software (Android app) that accesses the hidden secure element API (`com.android.nfc.extras`) and relays commands over a TCP socket,
- card emulation software (Python script) that controls the card emulation hardware and relays commands over a TCP socket,
- ACS ACR 122U NFC reader in software card emulation mode,
- HID OMNIKEY 5321 USB contactless reader,
- reader application (Java SE).

As we did not have access to a mobile phone that had *real* applications (e.g. a credit card or an access control applet) on its secure element, we decided to access the GlobalPlatform card manager application (issuer security domain) for our tests of the relay system. Therefore, our reader application sent the following APDU commands:

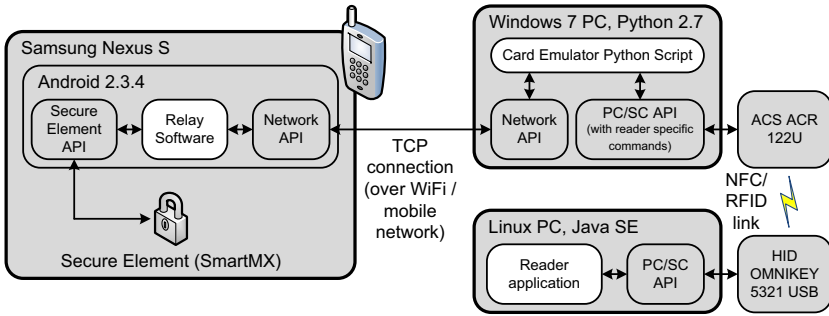


Fig. 2. Proof-of-concept relay system: Pre-existing components are drawn with gray background. Our customized components are drawn with white background.

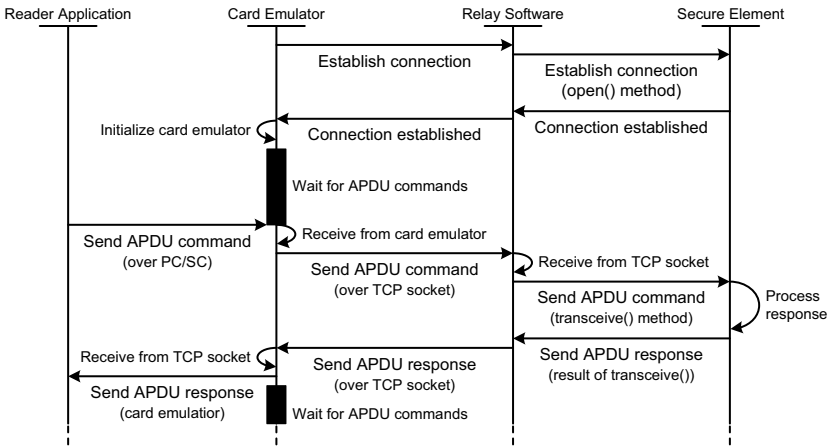


Fig. 3. Command/APDU flow diagram

1. SELECT card manager by AID: 00A4040008A000000003000000, expected response: File control information template (105 byte)
2. GET_DATA for data object '65': 00CA006500, expected response: Reference data not found error (2 byte)
3. GET_DATA for data object '66': 00CA006600, expected response: Card data/security domain management data (78 byte)

Fig. 3 shows the command/APDU flow diagram of the relay system. Initially the card emulator establishes a connection to the victim's secure element. When the card emulator is put in range of an RFID/NFC reader, the reader sends an APDU command. This command is forwarded to the secure element. The secure element generates a response and the relay software, then, returns this response back through the card emulator to the reader.

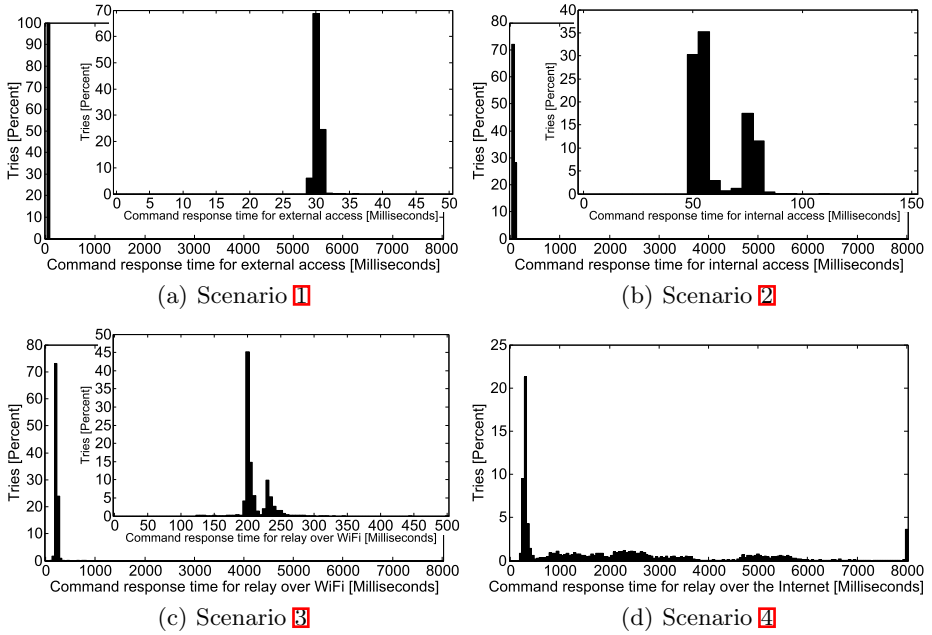


Fig. 4. Histograms of delay between command and response at the reader side for the APDU “SELECT card manager by AID” for 5000 repetitions. The histogram is divided into 160 bins. Each bin has a width of 50 ms. The last bin also contains all measurements above 8000 ms. (a) is zoomed from 0 to 50 ms with 1-ms-bins. (b) is zoomed from 0 to 150 ms with 5-ms-bins. (c) is zoomed from 0 to 500 ms with 5-ms-bins.

In our implementation we designed the card emulator as a server to which the relay software on any victim’s mobile phone can connect. That way, the card emulator can choose one of the connected secure elements as soon as a reader is in range. This method also bypasses firewalls that protect the victim’s device from incoming TCP connections.

4.3 Measurement Results

To verify the feasibility of our relay system, we compared four different scenarios:

1. Direct access to the secure element with an external reader (i.e. no relay),
2. direct access to the secure element with an app on the phone,
3. access through the relay system using a direct WiFi link between the phone and the card emulator,
4. access through the relay system using the mobile phone network and an Internet link between the phone and the card emulator.

Fig. 4 shows the histograms of the delay between command and response at the reader side for the APDU “SELECT card manager by AID” for 5000 repetitions of the command-response sequence. The other APDUs mentioned in section 4.2

lead to similar results that only differed in delays due to command lengths. Except for scenario [2](#), the phone/the emulator was isolated from the reader between each repetition.

The delay for scenario [1](#) centers on about 30 ms. On-device access to the secure element (scenario [2](#)) already takes significantly longer (50 to 80 ms). The delay over a WiFi connection (scenario [3](#)) ranges from 190 to 260 ms. Thus, the WiFi relay link adds a delay in the range of 100 and 210 ms. For scenario [4](#) the delays start at about 200 ms and have a significant peak around 300 ms. Yet, more than 55 percent of the measured delays are above 1000 ms, more than 19 percent are above 4000 ms, and more than 2 percent are above 10000 ms.

Typical limits for contactless transactions in transport ticketing and payment are between 300 to 500 ms (cf. [14](#)). These limits apply to overall transactions, which, typically, consist of multiple command-response pairs. The EMV specification for contactless payment systems [3](#) specifies a limit of 500 ms for a contactless payment transaction. However, a payment terminal is not required to interrupt a transaction if it takes longer than this limit. The limit is merely meant as a benchmark target to maintain user experience.

Consequently, both relay scenarios are likely to fail these timings if transactions consist of several command-response pairs. Nevertheless, as the limits are not meant as hard timeouts (after which transactions are canceled), we do not see any problematic side effects of failed timings. For the relay scenario across the mobile network and the Internet, most of the tests showed a delay below 4000 ms. While this is significantly longer than typical delays for contactless transactions, it is still below the timeout limits (without timeout extension) imposed by the ISO/IEC 14443 standard. As contactless transactions (especially with mobile phones) are quite new and users are still not used to them, we assume that even long delays (in the order of 20 to 30 seconds per transaction) will not raise suspicions.

5 Conclusion and Outlook

We presented a new relay attack scenario against secure elements. With this scenario an attacker can use a secure element in a remote mobile phone over the Internet. Due to limited security features of current mobile phone systems, attackers are likely to perform such attacks even if the secure element APIs have mechanisms to prevent unauthorized access.

We described a possible implementation based on Android devices. We conducted several measurements to prove the feasibility of our implementation. The results show that our attack scenario is technically possible due to the lack of strict timing requirements in the communication protocols. Attacks are possible even over long distances. Nevertheless, a relay attack induces significantly longer delays than with usual contactless transactions.

Future work is necessary to find adequate solutions for avoiding on-device relay attacks. A possible direction would be the adoption of trusted computing concepts for mobile phone systems. That way, the SE could recognize if the

application processor is in a trusted state. Another possibility is the introduction of strict timeouts in contactless transactions. Timeouts could also be based on a history of measured command-response delays from previous transactions to detect significant deviations in comparison to previous transactions. A third possibility would be the explicit activation of card emulation by user interaction (e.g. by pressing a button on the mobile phone that is directly connected to the SE.) However, this would significantly complicate over-the-air card management.

References

1. Clark, S.: RIM releases BlackBerry NFC APIs. Near Field Communications World (May 2011), <http://www.nfcworld.com/2011/05/31/37778/rim-releases-blackberry-nfc-apis/>
2. Desmedt, Y., Goutier, C., Bengio, S.: Special Uses and Abuses of the Fiat-Shamir Passport Protocol (extended abstract). In: Pomerance, C. (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 21–39. Springer, Heidelberg (1988)
3. EMVCo: EMV Contactless Specifications for Payment Systems – Book A: Architecture and General Requirements, Version 2.1 (March 2011)
4. Francis, L., Hancke, G., Mayes, K., Markantonakis, K.: Practical NFC Peer-to-Peer Relay Attack Using Mobile Phones. In: Ors Yalcin, S.B. (ed.) RFIDSec 2010. LNCS, vol. 6370, pp. 35–49. Springer, Heidelberg (2010)
5. Gostev, A.: Monthly Malware Statistics: August 2011 (September 2011), <http://www.securelist.com/analysis/204792190>
6. Hancke, G.P.: A Practical Relay Attack on ISO 14443 Proximity Cards (January 2005), <http://www.rfidblog.org.uk/hancke-rfidrelay.pdf> (retrieved September 20, 2011)
7. Hancke, G.P., Mayes, K.E., Markantonakis, K.: Confidence in smart token proximity: Relay attacks revisited. Computers & Security 28(7), 615–627 (2009)
8. Höbarth, S., Mayrhofer, R.: A framework for on-device privilege escalation exploit execution on Android. In: Proceedings of IWSSI/SPMU (June 2011)
9. Jeon, W., Kim, J., Lee, Y., Won, D.: A Practical Analysis of Smartphone Security. In: Smith, M.J., Salvendy, G. (eds.) HCII 2011, Part I. LNCS, vol. 6771, pp. 311–320. Springer, Heidelberg (2011)
10. Kfir, Z., Wool, A.: Picking Virtual Pockets using Relay Attacks on Contactless Smartcard. In: Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM 2005), pp. 47–58 (September 2005)
11. McLean, H.: Nokia: No mobile wallet support in current NFC phones. Near Field Communications World (July 2011), <http://www.nfcworld.com/2011/07/21/38715/nokia-no-mobile-wallet-support-in-current-nfc-phones/>
12. RIM: Blackberry API 7.0.0: Package net.rim.device.api.io.nfc.emulation (2011), <http://www.blackberry.com/developers/docs/7.0.0api/net/rim/device/api/io/nfc/emulation/package-summary.html>
13. Roland, M., Langer, J., Scharinger, J.: Practical Attack Scenarios on Secure Element-enabled Mobile Devices. In: Proceedings of the Fourth International Workshop on Near Field Communication (NFC 2012), Helsinki, Finland, p. 6 (March 2012)
14. Smart Card Alliance: Transit and Contactless Open Payments: An Emerging Approach for Fare Collection (November 2011), http://www.smartcardalliance.org/resources/pdf/Open_Payments_WP_110811.pdf

Would You Mind Forking This Process? A Denial of Service Attack on Android (and Some Countermeasures)*

Alessandro Armando^{1,2}, Alessio Merlo^{1,3,**}, Mauro Migliardi⁴,
and Luca Verderame¹

¹ DIST, Università degli Studi di Genova, Italy
{armando,alessio.merlo}@dist.unige.it
luca.verderame@ai-lab.it

² Security & Trust Unit, FBK-irst, Trento, Italy
armando@fbk.eu

³ Università e-Campus, Italy
alessio.merlo@unicampus.it

⁴ DEI, University of Padova, Italy
mauro.migliardi@unipd.it

Abstract. We present a previously undisclosed vulnerability of Android OS which can be exploited by mounting a Denial-of-Service attack that makes devices become totally unresponsive. We discuss the characteristics of the vulnerability – which affects all versions of Android – and propose two different fixes, each involving little patching implementing a few architectural countermeasures. We also provide experimental evidence of the effectiveness of the exploit as well as of the proposed countermeasures.

1 Introduction

With more than 45% of US sales of smartphones in 3Q2011 [1], Android is arguably one of the greatest success stories of the software industry of the last few years. By leveraging a generic (albeit optimized for limited resource consumption) Linux kernel, Android runs on a wide variety of devices, from low-end to top-notch hardware, and supports the execution of a large number of applications available for download both inside and outside the Android Market.

Since most of the applications are developed by third-parties, a key feature of Android is the ability to sandbox applications, with the ultimate objective to achieve the following design goal:

A central design point of the Android security architecture is that no application, by default, has permission to perform any operation that would adversely impact other applications, the operating system, or the user.

<http://developer.android.com/guide/topics/security/security.html>

* This work has been partially funded by EU project FP7-257876 SPaCioS.

** Corresponding author.

Sandboxing is achieved combining the isolation guaranteed by the use of Virtual Machines together with the enforcement mechanism that can be obtained from the Linux access control by giving each application a different Linux identity (i.e. Linux user). Each of these two mechanisms is well known and has been thoroughly tested to achieve a high level of security; however, the interaction between them has not yet been fully explored and may still hide unchecked vulnerabilities.

In this paper we present a previously unknown vulnerability in Android OS that allows a malicious application to force the system to fork an unbounded number of processes and thereby mounting a Denial-of-Service (DoS) attack that makes the device totally unresponsive. Rebooting the device does not necessarily help as a (very) malicious application can make herself launched at boot-time. Thus, our findings show that the aforementioned Android design goal is not met: a malicious application can indeed severely impact all other applications, the operating system, and ultimately the user. To overcome this impasse we propose two solutions, each involving very small changes in the system, that fix the problem. We present experimental results confirming that all versions of Android OS (including the most recent ones, namely versions 4.0 and 4.0.3) suffer from the vulnerability. The experiments also confirm that our proposed fixes counter the DoS attack.

We have promptly reported our findings to Google, Android and the US-CERT. The vulnerability has been registered in the CVE database and has been assigned identifier CVE-2011-3918.

Structure of the Paper. In the next section we provide a brief description of Android OS. In Section 3 we present the vulnerability and in Section 4 we illustrate two possible solutions. In Section 5 we present some experimental results that confirm the effectiveness of the DoS attack as well as of the proposed countermeasures. In Section 6 we investigate works related to the security of the Android platform. We conclude in Section 7 with some final remarks.

2 The Android Architecture

The Android Architecture consists of 5 layers. The bottom layer (henceforth the *Linux layer*) is the Linux kernel. On top of the Linux layer live four Android-specific layers that we collectively call the *Android layers*.

2.1 The Android Layers

The Android Layers are from top to bottom: the Application layer, the Application Framework layer, the Android Runtime layer, and the Libraries layer:

- *Application Layer.* Applications are on the top of the stack and comprise both user and system applications which have been installed and execute on the device.

- *Application Framework Layer*. The Application Framework provides the main services of the platform that are exposed to applications as a set of APIs. This layer provides the System Server, that is a component containing the main modules for managing the device (e.g. *Activity Manager* and *Package Manager*) and for interacting with the underlying Linux drivers (e.g. *Telephony Manager* and *Location Manager* that handle the mobile hardware and the GPS module, respectively)
- *Android Runtime Layer*. This layer comprises the Dalvik virtual machine, the Android’s runtime’s core component, specifically optimized for efficient concurrent execution of disjoint VMs in a resource constrained environment. The Dalvik VM executes application files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint.
- *Libraries Layer*. The Libraries layer contains a set of C/C++ libraries providing useful tools to the upper layers. They are widely used by Application framework services. Examples of libraries are *bionic libc* (a customized implementation of libc for Android) and *SQL lite*.

2.2 The Linux Layer

Android relies on Linux kernel version 2.6 for core system services. Such services include process management and Inter-Process Communication (IPC). Each Android application, together with the corresponding Dalvik VM, is bound to a separate Linux process. Android uses a specific process, called Zygote, to enable code sharing across VM instances and to provide fast start-up for new processes. The Zygote process is created during Linux boot-strap. Every time a new Linux process is required (e.g., for starting a new Android application), a command is sent through a special *Unix domain socket* called Zygote socket. The Zygote process listens for incoming commands on the Zygote socket and generates a new process by forking itself as a Linux process. Differently from what happens in a normal Unix system, specialization of the child process behavior is not obtained by loading a new executable image, but only by loading the Java classes of the specific application inside the VM.

Communication between apps in Android is carried out through Unix sockets or the Binder driver. Sockets are used when data are small and well codified such as in Zygote socket case. In other cases (e.g., when data are big and heterogeneous), the Binder driver is used. Binder IPC mechanism relies on a kernel driver. Each process registers itself to the Binder driver and gets back a file descriptor. A process that wants to communicate with another process can simply send data through the file descriptor via an IOCTL command. The Binder kernel module sends received data directly to the destination process.

2.3 Interaction between the Android Layers and the Linux Layer

The interaction between the Android and the Linux layers is depicted in Fig. [11](#). When an application is launched a `startActivity` call is sent to the *Activity Manager Service*, a part of the System Server. The *Activity Manager Service*

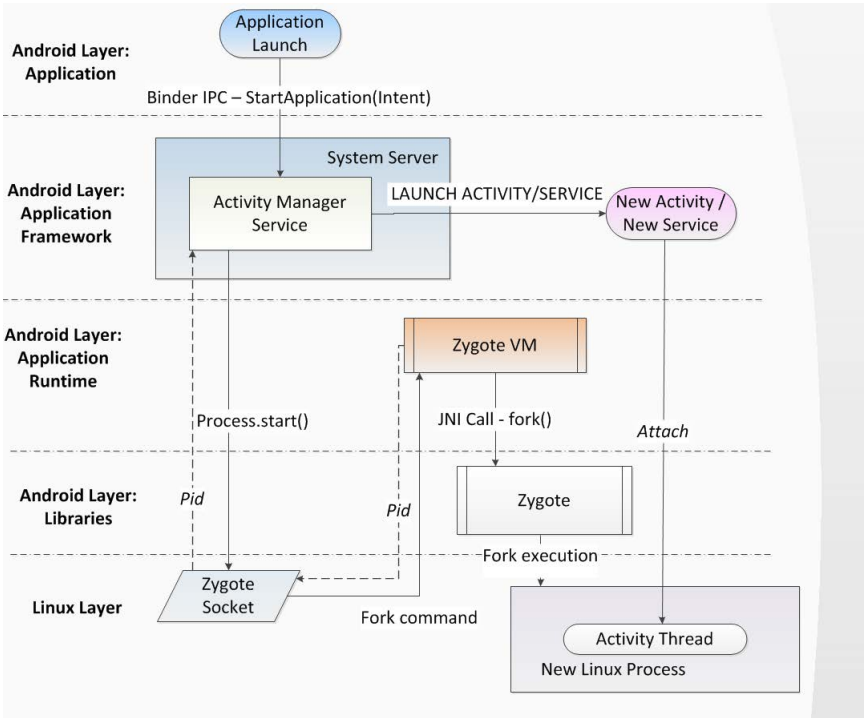


Fig. 1. Creation of a new process in Android

determines if the application has already an attached process at the Linux layer or if a new one is needed. The first case happens when an instance of the application has been previously started and it is currently executing in background; thus, the Activity Manager Service gets the corresponding process and brings back the application to foreground.

In the latter case, the Activity Manager Service calls `Process.start()`, a method of the static class `android.os.process`. This method connects the Activity Manager Service to the Linux layer via the Zygote socket and sends the fork command. The Zygote process has the exclusive right to fork new processes, thus, when Android requires the creation of a new process, the command must be issued to the Zygote process through the Zygote socket.

The command sent to the Zygote socket includes the name of the class whose static `main` method will be invoked to specialize the child process. The System Server uses a standard class (i.e., `android.app.ActivityThread`) when forking. In this class, a binding operation between the Linux process and an Android application is attempted. If no application is available for binding, the same class asks the Linux layer to kill the process.

If the spawning of the new process and the binding operation succeed, the Zygote process returns its child's PID to the Activity Manager Service.

3 The Vulnerability

The Zygote socket is owned by `root` but has permissions 666 (i.e. `rw-rw-rw`), implying that any Android application can read and write it and hence send commands to the Zygote process. This choice is justified by the need of the process hosting the System Server (whose UID is 1000, it is not owned by `root`, nor it belongs to the `root` group) to request the spawning of new processes. However, this has the side effect to enable any process to ask for a fork.

To avoid misuse, the Zygote process enforces a security policy that restricts the execution of the commands received on the Zygote socket. This policy is implemented in the `runOnce()` function:

```

1  boolean runOnce() throws ZygoteInit.MethodAndArgsCaller
   {
2      ...
3      applyUidSecurityPolicy(parsedArgs, peer)
4      applyCapabilitiesSecurityPolicy(parsedArgs, peer);
5      applyDebuggerSecurityPolicy(parsedArgs);
6      applyRlimitSecurityPolicy(parsedArgs, peer);
7      ...

```

The policy prevents from

1. issuing the command that specifies a UID and a GID for the creation of new processes if requestor is not `root` or System Server (cf. line 3)
2. creating a child process with more capabilities than its parent (cf. line 4), and
3. enabling debug-mode flags and specifying `rlimit` bound if requestor is not `root` or the System is not in "factory test mode" (cf. lines 5 and 6).

Moreover, only few limitations are put on the (static) class used to customize the child process. In particular, two checks are performed namely checking whether 1) the class contains a static `main()` method and 2) it belongs to the `System` package, which is the only one accepted by the Dalvik System Class loader.

Unfortunately, these security checks do not include a proper control of the identity (i.e., UID) of the requestor, therefore allowing each Linux process (and, its bound Android application or service) to send not necessarily legitimate but acceptable in the current Android security framework fork command to the Zygote socket as long as a valid static class is provided.

We discovered that by using the System static class `com.android.internal.util.WithFramework` it is possible to force the Zygote process to fork, generating a dummy process which is kept alive at Linux layer. Such class does not perform any binding operation with an Android application, thus not triggering the removal of unbound new processes as the default `android.app.ActivityThread` class does.

In this way, all the security policies applied by the Zygote process are bypassed, leading to the building of a persistent Linux process which occupies memory resources in the device.

Thus, by flooding the Zygote socket with such requests, an increasingly large number of dummy processes is built until all the memory resources are exhausted (Fork bomb attack).

The Android layers are unable to notice the generation of dummy processes and, consequently, to intervene.

On the other hand, the creation of processes at the Linux layer is legal and managed directly by the kernel. Thus, none of the involved layers is able to recognize such behavior as malicious.

As soon as the dummy processes consume all the available resources, a safety mechanism reboots the device. Thus, by launching the attack during bootstrapping, it is possible to lock the device into an endless boot-loop, thereby locking the use of the device.

Notice that to mount the attack, the malicious application does not require any special permission, therefore it looks harmless to the user upon installation.

4 Countermeasures

We describe two possible approaches to fix the previously described vulnerability:

1. **Zygote process fix.** This fix consists of checking whether the fork request to the Zygote process comes from a legal source (at present, only the System Server, although our patch is trivially adaptable to future developments).
2. **Zygote socket fix.** This fix restricts the permissions on the Zygote socket at the Linux layer.

4.1 Checking Fork Requests Inside the Zygote Process

As said in Section 3, the Zygote process does not perform any specific check on the identity of the process that requests the fork operation. Nevertheless, the Zygote socket is a Unix Domain socket created during the boot-strap of the Linux system. An important feature of Unix domain sockets is the *credential passing mechanism* which allows to identify endpoints connected to the socket by means of their PID, UID and GID.

This implies that the Zygote process can retrieve the identity (i.e., UID) of the requesting process. The extended policy takes advantage of this feature and applies a new filter based on the UID of the requesting process. In particular, since the process corresponding to the System Server has UID 1000 (statically defined), the extended security policy filters the requests reaching the Zygote socket by accepting only fork requests from UID 1000 and UID 0 (*root*):

```

void applyForkSecurityPolicy(peer){
    int peerUid = peer.getUid();
    if (peerUid == 0 || peerUid == Process.SYSTEM_UID) {
        // Root or System can send commands
        Log.d("ZYGOTE_PATCH", "root or SS request: ok");
    }

    else { Log.d("ZYGOTE_PATCH", "user request"+ peerUid +
        ": blocked");
        throw new ZygoteSecurityException("user UID" +
            peerUid + " tries to fork new process");
    }
}

```

We implemented the previous policy by adding this check at the end of the native policy in the `runOnce()` method of the Zygote process.

4.2 Modifying the Linux Permissions of the Zygote Socket

The idea is to reduce the Linux permissions for the Zygote socket. Currently, the Zygote socket is owned by `root` and—as we said above—it has permissions `666`. It is possible to modify both the owner (no `root`) and permissions of Zygote socket from `666` (i.e., `rw-rw-rw`) to `660` (i.e., `rw-rw---`). In this way the System Server retains read and write access. We implemented this modification in three steps:

1. Create a new owner for the Zygote socket. We added a new UID (i.e. 9988) in the file `android_filesystem_config.h` which contains statically assigned UID for Android system services. Then, in the same file we bind an ad hoc user `zygote_socket` and the new UID.

```

...
#define AID_ZYGSOCKET 9988 / Zygote socket UID;
#define AID_MISC 9998 /* access to misc storage */
#define AID_NOBODY 9999

#define AID_APP 10000 /* first user app */
...
static struct android_id_info android_ids[] = {
    { "root",      AID_ROOT, },
    { "system",   AID_SYSTEM, },
    ...
    { "misc",     AID_MISC, },
    { "zygote_socket", AID_ZYGSOCKET, },
    { "nobody",   AID_NOBODY, },
};
...

```

- Change the owner and permissions of the Zygote socket. The user `zygote_socket` is associated with the Zygote socket by modifying `init.rc` and by setting its permissions to 660.

```
service zygote /system/bin/app_process -Xzygote /
system/bin --zygote --start-system-server
socket zygote stream 660 zygote_socket zygote_socket
onrestart write /sys/android_power/request_state
wake
onrestart write /sys/power/state on
onrestart restart media
onrestart restart netd
```

- Include the UID of the Zygote socket owner in the group list of the System Server. Since also the System Server is generated through a fork request to the Zygote process, we modified the parameter of the fork command corresponding to the set of groups the new process belongs to. We added the UID to such set as follows:

```
...
String args[] = {
    "--setuid=1000",
    "--setgid=1000",
    "--setgroups=1001,1002,1003,1004,1005,1006,1007,
1008,1009,1010,1018,3001,3002,3003,9988",
    "--capabilities=130104352,130104352",
    "--runtime-init",
    "--nice-name=system_server",
    "com.android.server.SystemServer",
};
...
```

This modification implies that user applications cannot connect to the Zygote socket while the System Server (which is in the Zygote socket's group) can still issue the fork command.

5 Experimental Results

We tested the DoS vulnerability on all versions of Android OS currently available (i.e. $\leq 4.0.3$) by building an Android malicious application (i.e. DoSChecker) as described in Sect. 3. We made tests both on actual and simulated devices. All our tests have produced an unbounded number of dummy processes, thus revealing that all versions suffer from the vulnerability described in this paper.

The Testing Environment. We used a laptop equipped with *Android SDK r16*. We tested the actual devices listed in Tab. 1; newer Android versions, like 4.0

and 4.0.3, have been tested, instead, on Android device emulator. The behavior of the actual and simulated devices have been traced with *Adb Logcat* tool and *Adb shell* via a Windows shell.

Since a DoS is strictly related to the amount of resources, we tested actual devices with heterogeneous hardware in order to assess the relation between the availability of resources and the time spent to accomplish a successful attack.

Table 1. Actual devices used in experiments

Device Model	Android Versions
Lg Optimus One p550	2.2.3, 2.2.4 (stock LG), 2.2.4 (rooted)
Samsung Galaxy S	2.3.4 (stock Samsung), 2.3.7 (rooted Cyanogen 7)
Samsung Next GT-S5570	2.3.4 (stock Samsung), 2.3.4 (rooted)
Samsung Galaxy Tab 7.1	3.1 (stock Samsung), 3.1 (rooted)
HTC Desire HD	2.3.2 (stock HTC), 2.3.2 (rooted)

5.1 Exploiting the Vulnerability

Testing with Actual Devices. Once activating the DoSChecker application, devices with limited resources (e.g. LG Optimus One) freeze in less than a minute while others freeze in at most 2 minutes (e.g. Galaxy Tab). Our tests show an almost linear dependence of the duration of the attack from the amount of resources available in the device. During the attack, users experience a progressive reduction of the system responsiveness that ends with the system crash and reboot. While the number of dummy processes increases, Android tries to kill legitimate applications to free resources, but has no access to the dummy processes. This behavior leads to the killing of other application’s processes including system processes (such as home process). In several cases also the *System Server* crashes.

Once an application is killed, Android tries to restart it after a variable period of time but, in such scenario, DoSChecker fills the memory just freed with dummy processes, causing the failure of the restart procedure. Once physical resources are exhausted and Android is not able to restarts applications the device is restarted. DoSChecker has the same behavior both on standard and rooted (i.e. devices where non-system software components may temporary acquire root permissions) Android devices.

Testing with Emulated Devices. The use of the Android emulator provided us a twofold opportunity. First, the Android emulator allowed us to check the DoS vulnerability on newer Android versions such as Android 4.0 and 4.0.3; the testing procedure is similar and the experimental result is still the forking of an unbounded number of processes. However, we observed that in the emulated environment, where the amount of available resources depends on the host PC, the amount of dummy processes generated would greatly overcome the hardware capability of any device currently available on market.

Running DoSChecker as a Boot Service. Since the exploitation of the vulnerability takes to the reboot of the device, we added DoSChecker as a service starting at boot. Such operation is pretty straightforward and it does not require any specific Android permission. In particular, we added a java class in DoSChecker that, acting as a Broadcast receiver, intercepts the system broadcast messages and starts DoSCheck as a service whenever a bootstrap terminates (i.e. when the system broadcast message `android.intent.action.BOOT_COMPLETED` is received). Our tests show that the exploitation of the vulnerability at boot prevents Android from successfully completing the reboot process, thus getting the mobile device stuck. The only ways to recover the phone in such scenario, are to identify and manually uninstall the malicious application via *adb tool* or reflashing the device.

5.2 Testing the Countermeasures

We implemented the two countermeasures (i.e. Zygote process and Zygote socket fixes) proposed in Sect. 4. In particular, for each Android version, both on the actual and emulated devices, we built two patched versions, each implementing a countermeasure, by recompiling Android from scratch. The building process provides two output images called `system.img` and `ramdisk.img`. `System.img` contains all the Android system classes compiled while `ramdisk.img` corresponds to the Android RAM disk image which is loaded during bootstrapping and which includes, among others, the `init.rc` file. The *Zygote process fix*, which extends the security policy applied by the Zygote process, affects `system.img` only. The *Zygote socket fix*, which needs a modification of `init.rc`, affects both `system.img` and `ramdisk.img`.

Our tests show that both countermeasure are effective and prevent the fork of dummy process, thereby solving the vulnerability. Moreover, by using actual devices we have empirically proven that the proper functioning of devices was preserved (i.e. the proposed countermeasures do not affect the normal flow of Android) in all the tested cases.

6 Related Works

Security on Android platform is quite a new research field. Current literature can be classified into three trends: i) static analysis of Android applications, ii) assessment of Android access control and permissions policies, iii) malware and virus detection.

Static analysis is related to the development of white box or black box methodologies for detecting malicious behaviors in Android applications before installing them on the device. To this aim, Enck et al. [2] executed a horizontal study of Android applications aimed at discovering stealing of personal data. Besides, Fuchs et al. [3] proposed Scandroid as a tool for automatically reasoning about the security of Android applications.

Static analysis could help identifying the calls to the Zygote socket. However, UNIX sockets might be used for legitimated goals and recognizing the specific socket name might be made problematic even with a simple string concatenation.

In any case, to our knowledge no current static analysis tool identifies exploits of the described vulnerability.

The main part of current literature is focused on access control and permissions. For instance, [4] proposes a methodology for assessing the actual privileges of Android applications. This paper also proposes Stowaway, a tool for detecting over-privilege in compiled Android applications. Nauman et al. [5] proposes Apex as a policy enforcement framework for Android that allows a user to selectively grant permissions to applications as well as to impose constraints on the usage of resources. A different approach is proposed by Ongtang et al. [6] who present Secure Application INTeraction (SAINT), a modified infrastructure that governs install-time permissions assignment. Other works are focused on issues related to privilege escalation. Bugiel et al. [7] propose XManDroid (eXtended Monitoring on Android), a security framework that extends the native monitoring mechanism of Android to detect privilege escalation attacks. In [8] authors state that, under proper assumptions, a privilege escalation attack is possible on Android framework. Some research were made on analyzing native Android security policies [9] focusing on possible threats and solutions to mitigate the problem [10] of privilege escalation.

Nonetheless, the vulnerability disclosed in this paper requires no special permission, thus these approaches are not a valid solution.

Regarding virus and malware detection, Dagon et al. [11] made a comprehensive assessment of the state of the art of mobile viruses and malwares which can affect Android devices. In [12] a methodology for mobile forensics analysis in Android environments is proposed. In [13] Crowdroid, a malware detector executing a dynamic analysis of the application behavior, is proposed. Schmidt et al. [14] inspect Android executables to extract their function calls and compare them with malware executables for classification aim.

Specific malware signatures for exploiting the vulnerability described in this paper could be generated, however none is available today.

Besides, from a general point of view, all these works have been driven by the need to improve the privacy of the user. In such direction, Zhou et al. [15] argue the need of a new native privacy mode in Android smartphones.

At the best of our knowledge, this is the first work related to Denial of Service issues on Android platform.

As a final remark, none of the previous works investigates relations and security issues related to interactions between the Android and the Linux layers.

7 Conclusions

In this paper, we presented a previously undisclosed vulnerability on Android devices which is the first vulnerability on Android that leads to a DoS attack of this severity. We also developed a sample malicious application, (i.e. DoSCheck) which exploits the vulnerabilities, and we proposed two fixes for securing the Android OS against the vulnerability. We reported such vulnerability to Android security team which will include a patch in an upcoming update of the Android OS. Furthermore, we plan to publicly release both DoSCheck code and patched

systems in the very near future, accordingly with a responsible disclosure policy we are discussing with Android group and Open Handset Alliance.

References

1. Gartner Group. Press Release (November 2011), <http://www.gartner.com/it/page.jsp?id=1848514>
2. Enck, W., Ocateau, D., McDaniel, P., Chaudhuri, S.: A study of android application security. In: Proceedings of the 20th USENIX Conference on Security, SEC 2011, p. 21. USENIX Association, Berkeley (2011)
3. Fuchs, A.P., Chaudhuri, A., Foster, J.S.: Scandroid: Automated security certification of android applications
4. Felt, A.P., Chin, E., Hanna, S., Song, D., Wagner, D.: Android permissions demystified. In: Proceedings of the 18th ACM Conference on Computer and Communications Security, CCS 2011, pp. 627–638 (2011)
5. Nauman, M., Khan, S., Zhang, X.: Apex: extending android permission model and enforcement with user-defined runtime constraints. In: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security, ASIACCS 2010, pp. 328–332. ACM, New York (2010)
6. Ongtang, M., McLaughlin, S., Enck, W., McDaniel, P.: Semantically rich application-centric security in android. In: ACSAC 2009: Annual Computer Security Applications Conference (2009)
7. Bugiel, S., Davi, L., Dmitrienko, A., Fischer, T., Sadeghi, A.-R.: Xmandroid: A new android evolution to mitigate privilege escalation attacks. Technical Report TR-2011-04, Technische Univ. Darmstadt (April 2011)
8. Davi, L., Dmitrienko, A., Sadeghi, A.-R., Winandy, M.: Privilege Escalation Attacks on Android. In: Burmester, M., Tsudik, G., Magliveras, S., Ilić, I. (eds.) ISC 2010. LNCS, vol. 6531, pp. 346–360. Springer, Heidelberg (2011)
9. Chin, E., Felt, A.P., Greenwood, K., Wagner, D.: Analyzing inter-application communication in Android. In: Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services, MobiSys 2011, pp. 239–252. ACM, New York (2011)
10. Shabtai, A., Fledel, Y., Kanonov, U., Elovici, Y., Dolev, S.: Google android: A state-of-the-art review of security mechanisms. CoRR, abs/0912.5101 (2009)
11. Dagon, D., Martin, T., Starner, T.: Mobile phones as computing devices: The viruses are coming! IEEE Pervasive Computing 3(4), 11–15 (2004)
12. Di Cerbo, F., Girardello, A., Michahelles, F., Voronkova, S.: Detection of Malicious Applications on Android OS. In: Sako, H., Franke, K., Saitoh, S. (eds.) IWCF 2010. LNCS, vol. 6540, pp. 138–149. Springer, Heidelberg (2011)
13. Burguera, I., Zurutuza, U., Nadjm-Therani, S.: Crowdroid: behavior-based malware detection system for android. In: Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices, SPSM 2011 (2011)
14. Schmidt, A.-D., Bye, R., Schmidt, H.-G., Clausen, J., Kiraz, O., Yuksel, K.A., Camtepe, S.A., Albayrak, S.: Static analysis of executables for collaborative malware detection on android. In: IEEE International Conference on Communications, ICC 2009, pp. 1–5 (June 2009)
15. Zhou, Y., Zhang, X., Jiang, X., Freeh, V.W.: Taming Information-Stealing Smartphone Applications (on Android). In: McCune, J.M., Balacheff, B., Perrig, A., Sadeghi, A.-R., Sasse, A., Beres, Y. (eds.) Trust 2011. LNCS, vol. 6740, pp. 93–107. Springer, Heidelberg (2011)

An Approach to Detecting Inter-Session Data Flow Induced by Object Pooling*

Bernhard J. Berger and Karsten Sohr

Center for Computing Technologies (TZI), Universität Bremen
Bibliothekstr.1, 28359 Bremen, Germany
{berber,sohr}@tzi.de

Abstract. Security tools, using static code analysis, are employed to find common bug classes, such as SQL injections and cross-site scripting vulnerabilities. This paper focuses on another bug class that is related to the object-pool pattern, which allows objects to be reused over multiple sessions. We show that the pattern is applied in a wide range of Java Enterprise frameworks and describe the problem of inter-session data flows, which comes along with the pattern. To demonstrate that the problem is relevant, we analyzed different open-source and a proprietary commercial software, with the help of a detection approach we introduce. We were able to show that the problem class occurred in these applications and posed a threat to the confidentiality of the closed-source software.

1 Introduction

Security tools, using static code analysis, are employed by software producers more and more frequently to detect security bugs introduced during the implementation. These static analyses can find bug classes, such as SQL injections and cross-site scripting vulnerabilities. The commercial success of those tools shows that industry is realizing the importance of software security [10].

One kind of applications that are prone to the vulnerabilities mentioned above are business applications, offering services or web interfaces to customers. These systems share the property that they process data from different users at the same time that must be handled confidentially. To separate data from different users, the applications provide a session, an object that represents the server-side state [24]. The common bug classes pose a threat to the confidentiality of the user's data and the availability of the system.

A frequently-used framework to implement business applications is the Java platform, Enterprise Edition that supports different APIs to ease the development of business applications [19]. The core framework is complemented by different libraries like Apache Struts [27] or Spring [26], which provide additional capabilities and should reduce development time.

Within this paper, we describe an additional threat to the confidentiality of a user's data that relates to the object-pool pattern described by Kircher

* This work was supported by the German Federal Ministry of Education and Research (BMBF) under the grant 01IS10015B (ASKS project).

and Jain [14]. This pattern is used within different enterprise frameworks to improve the performance by reusing class instances for different requests [25]. Nevertheless, these pooled instances may create the possibility of confidential data-flows between different users of a system if they contain fields that are not handled correctly. This behaviour is known as the “Object Cesspool anti pattern” in the development community. Furthermore, we will show a light-weight and bytecode-based approach to detecting such vulnerabilities. The described approach is evaluated with the help of a closed-source and two open-source systems. We detected possible security vulnerabilities in the analyzed software and the vendor of the commercial software approved that a part of the reported findings were exploitable .

The remainder of the paper is structured as follows. An overview of related work is given in Section 2 followed by a detailed description of the problem, as well as a possible way to address it in Section 3. To show the validity of our approach, we evaluate our proposed solution in Section 4 and conclude with a summary of our work and a look on the next steps we will pursue in the future.

2 Related Work

There is a plethora of works that aim to find security vulnerabilities with the help of static analysis [4,7,29,18]. One approach that handles vulnerabilities for Java enterprise applications was developed by Livshits and Lam [16]. They present a framework to detect different input-related vulnerabilities, such as SQL injections, cross-site scripting, HTTP response splitting, path traversal, and command injections. To find possible vulnerabilities, they identify sources of user input and track it through the entire application to certain methods that are known to be prone to a specific kind of vulnerability. When a possible data flow is found, it is an evidence for a vulnerability and will be reported.

Hammer and Snelting describe how static analysis can be used to check whether the confidentiality or the integrity of the data processed by a program can be threatened by a user [13]. For their information-flow control, they use a program dependence graph, a representation already known in the area of static program analysis [15,3,11]. In a succeeding work, Hammer evaluates his approach with some real-world examples and lists results of his performance measurements [12].

Another well-known vulnerability class that can be detected with static analysis is time-of-check-to-time-of-use (TOCTTOU) vulnerabilities [5]. Here, an attacker tries to manipulate a file an application tries to read between the time it checks some properties (e.g. access rights or existence) and the usage of a file. Since the two operations are not atomic, an attacker might manipulate the file between the two operations, such as creating a symbolic link to another file to trick the application into deleting or overwriting it. It is a special kind of race condition between the programmer and the attacker.

3 Inter-Session Data Flow

In the following section, we first summarize whether certain frameworks or their implementation may use the object-pool pattern and how to detect these pooled objects. Afterwards, we give a detailed description of the problem and complement it with an example. Finally, we introduce our attempt to detecting the described problem.

3.1 Analyzed Frameworks

We searched different specifications and documentations for an evidence if the object-pool pattern may be applied in an implementation. We started with some core specifications, typically used in Java enterprise applications, such as the Enterprise JavaBean specification, the JavaServer Pages specification and the Java Servlet specification. Beyond that, we took a look at the Struts 1 framework, since it is used by our partners in the research project ASKS to implement their user interfaces.

Enterprise JavaBean. An Enterprise JavaBean (EJB) is a managed component, running in an application container and follows a well-defined lifecycle that is steered by the application container. There are different commercial and non-commercial containers that implement one of the EJB specifications.

A special kind EJB is the stateless session bean, a component that is accessible for local and remote clients and implements business logic. The term “stateless” means that it does not hold a state to process a request sent by a client. Therefore, it is a candidate to be used within the object-pool pattern. The EJB specification in the version 2.1, as well as 3.0, states: “Since stateless session bean instances are typically pooled, the time of the clients invocation of the create method need not have any direct relationship to the containers invocation of the `PostConstruct/ejbCreate` method on the stateless session bean instance.” [8, p. 8] and [9, p. 72]. This formulation does not prescribe the behavior of an application container, but gives a hint that the object-pooling mechanism may be used by application container implementations.

Depending on the version of the EJB specification, a stateless session bean must implement the interface `javax.ejb.SessionBean` or be annotated with `javax.ejb.Stateless`.

Java Servlet. A Java Servlet is a managed entity that reacts on requests and produces some response that is returned to the origin of the request. An example is the `HttpServlet-Interface` that reacts on HTTP queries and an implementing class can generate any valid HTTP response. The Servlet API, as well as the lifecycle that a Servlet container has to provide, are defined in the Java Servlet specification. The specification mentions that the servlet container may choose to pool such objects [17, p. 7].

The base interface for all Java Servlets is `javax.servlet.Servlet`. To detect all Servlets, one has to find all classes that implement this interface.

JavaServer Pages. JavaServer Pages (JSP) is a Java-based template language to easily generate dynamic web pages. A programmer can write a textual file which he enhances with some quoted Java code to insert dynamic data at runtime. A typical field of application of JSPs is the generation of HTML and XML documents, which are served to a web browser. According to the JavaServer Pages specification [23], the base interface for all JSPs is `javax.servlet.jsp.HttpJspPage` which is a Java Servlet. Therefore, JSPs are possibly pooled, too.

Struts 1. Apache Struts is a framework that implements the model-view-controller pattern [22] for JSPs. The framework uses traditional JSPs for the view part of the pattern and adds *Actions* to assume the controller role. An action is a Java class that inherits from `org.apache.struts.action.Action`.

The online documentation of Struts 1¹ says explicitly that one should not use instance variables within actions because of possible multi-threading issues that originate from the fact that each action is instantiated once and reused for all requests.

Remark. The specifications mention that pooling of objects may be used which means that it is an implementation-defined behaviour of the frameworks or application containers. We inspected some open-source implementations of the specifications under investigation, such as the JBoss application container [21], the Glassfish application container, the Tomcat web server, and the Struts implementation and found out that all of them use object pooling.

A developer, who uses the aforementioned frameworks, does not explicitly use the pooling mechanism, instead he implements certain interfaces or uses some given annotations and the framework automatically pools instances of these objects. None of the specifications mentioned above provide some kind of reset method that is called when an object is returned to the pool and whose task is to clear the state of an instance.

3.2 Problem

As already mentioned, enterprise applications allow different users to connect simultaneously. Unique identifiers assigned to each user allow the container to distinguish the different users and provide an independent session for each of them. Ideally, each session would be handled in its own memory area to avoid private data to leak between sessions.

If the response time of a software system is an important non-functional requirement, software architects may decide to use the object-pool pattern since it can improve the speed of an application, by reusing class instances [25]. Normally, an object is temporarily bound to the client that is using the object, whereas we observed in the case of *Struts* that it is also used concurrently by different clients.

¹ http://struts.apache.org/1.x/userGuide/building_controller.html

The concept of object pooling contradicts to a strict separation of sessions since it explicitly shares object instances—which may contain sensitive information. This may lead to the aforementioned information flow between different sessions and therefore between different users, as long as the sensitive data is not removed when the instance is returned to the pool. This is not done by the frameworks we analyzed.

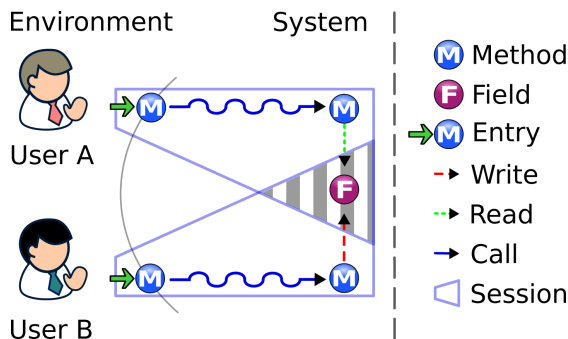


Fig. 1. Communication between sessions

The problem is depicted in Figure 1. Two different users are calling entry methods of a system, which call some internal methods to process the user request. Let us assume that the field on the right-hand side belongs to a pooled object. User B's request is processed first, an instance is fetched from the pool and a method is called that writes data to the field. The pooled instance now holds data of B and is returned to the object pool. The next request is issued by User A, who receives the same instance from the pool and calls a method that reads the data from the field and processes them. This constellation is a situation where sensitive information can leave the session (the framed area within Figure 1) of User A and flow into the session of User B or vice versa.

3.3 Example

A concrete example of an inter-session data flow problem is given in Listing 1.1, which shows a Struts action written in Java (see Section 3.1 for more information).

The action class `ExampleAction` has a private field `userEMail` of type `String` that can be accessed by the two methods `enterEMail` and `sendEMail`. These methods can be called by an HTTP request, depending on an HTTP parameter. The former method stores an e-mail address provided by the query into the field `userEMail` and does some additional operations, such as checking the provided value for correctness. The latter method uses the stored value from `userEMail` to send some private information to it.

The obvious problem in Listing 1.1 is the case in which User A sets her e-mail address and User B triggers the `send` method, which leads to the situation where data of User B might be sent to User A.

```

1 // imports are omitted
2
3 public class ExampleAction extends Action {
4     private String userEmail;
5
6     public ActionForward enterEmail(ActionMapping mapping,
7                                     ActionForm form,
8                                     HttpServletRequest request,
9                                     HttpServletResponse response)
10        throws IOException, ServletException {
11         EMailForm emailForm = (EMailForm)form;
12         this.userEmail = emailForm.getEmail();
13         ...
14         checkAndStoreEmailAddress(this.userEmail);
15     }
16
17     public ActionForward sendEmail(ActionMapping mapping,
18                                    ActionForm form,
19                                    HttpServletRequest request,
20                                    HttpServletResponse response)
21        throws IOException, ServletException {
22         ...
23         sendEmail(this.userEmail);
24         ...
25     }
26 }

```

Listing 1.1. Struts example code

3.4 Detection Approach

To investigate whether the problem is relevant in real life, we implemented a detection algorithm to identify possible inter-session data flows using the Bauhaus tool-suite [20]. The basis for our analysis is a resource-flow graph (RFG), which contains typed nodes for every declared element and typed edges for relations between those elements. For the Java language, there are, for instance, nodes for classes, interfaces, methods, and fields, whereas edges represent call relations, inheritance, field access, or the used types. The call graph, for example, is a part of our resource-flow graph.

To construct an RFG, we use the application's bytecode, which gives us the possibility to analyze a software without having its source code. To analyze JSPs we use the JSP compiler Jasper from Apache Tomcat [28]. With the help of the JSP compiler a JSP file can be translated into normal Java source code, which can be processed by the default Java compiler. This way, we do not need extra support for JSP files. The RFG for an application is generated from the class files belonging to the system under investigation, as well as the depending libraries.

A simple approach for detecting such data flows would be to search for all pooled classes and report them if they contain a field. This way, one can identify all potential vulnerabilities, but it produces false positives since not every field leads to a flow of sensitive information. To reduce the rate of false positives, we implemented a detection algorithm that is a bit more sophisticated (see Figure 2).

The algorithm consists of three automatic successive steps. First, we identify all pooled instances within an application (lines 2 – 7). For this step, the method *isPooled* uses the knowledge how to detect these objects which we have already described in Section 3.1. This step is similar to the naive approach. In the second step, we identify all methods that may be called by a user (lines 8 – 9)—in contrast to those methods that are called by the framework. These entries are the same ones that are used by injection-related analyses, for example, described by Livshits et al. [16]. Starting from these entry methods, we traverse the static call graph (call to function *getTransitivelyCalled* in line 10) to determine the methods that are reachable from the entry points. In the last step, we collect all fields, belonging to the pooled class that are accessed within the reachable methods (lines 11 – 20). Finally, a tuple is created, consisting of the accessing method and the access type (read, write), and is added to the list of results of the field that is accessed (lines 13, 18).

```

Input: RFG Resource Flow Graph
Output: Result A list of tuples, containing the method and the access type, for each field

1 begin
2   pooled  $\leftarrow \emptyset$ ;
3   foreach Class class  $\in$  RFG do
4     | if isPooled(class) then
5     | | pooled  $\leftarrow$  pooled  $\cup$  {class};
6     | end
7   end
8   foreach Method method  $\in$  RFG do
9     | if isEntry(method) then
10    | | foreach Method callee  $\in$  getTransitivelyCalled(method) do
11    | | | foreach Field field  $\in$  getReadFields(callee) do
12    | | | | if getContainingClass(field)  $\in$  pooled then
13    | | | | | Result[field].append((callee, Read));
14    | | | | end
15    | | | end
16    | | | foreach Field field  $\in$  getWrittenFields(callee) do
17    | | | | if getContainingClass(field)  $\in$  pooled then
18    | | | | | Result[field].append((callee, Write));
19    | | | | end
20    | | | end
21    | | end
22    | end
23  end
24 end

```

Fig. 2. Detection algorithm for inter-session data flows

The results of our approach can be divided into four different classes, according to the fact whether they are read, written, or read and written.

read only. The case that a field is just read from entries may be harmless, but not in all cases. If the field is a base data-type, such as *int*, *float* or *boolean* or it is an class instance that is not manipulated (for example, immutable instances), the field cannot lead to an inter-session data flow. In this case, the state of the program is not manipulated and therefore it is not possible for sensitive data to flow to another session. In the case that a class instance

is read, it is possible that a method of the instance is called that manipulates its state, which may lead to an inter-session data flow.

write only. If a field is just written, there is no way for sensitive information to flow since no other session will read the data that may be stored in the object. Therefore, these findings are flagged as false positives.

read and write. This group of results definitely can lead to data flows between sessions. The examples in Section 3.2 and Section 3.3 belong to this class.

no access. If a field is not accessed at all, it poses no threat to the confidentiality of the data and therefore this is a false positive finding in the naive approach, too.

Our approach reports the groups “read only” and “read and write” as findings, in contrast to the naive approach described above. We are aware of the fact that the reported findings are not free of false positives and false negatives. This shortcoming will be discussed in Section 4.2.

4 Evaluation

For our evaluation, we considered three applications, two being open source and the other being proprietary software. The criterion for selecting these was whether they used one of the frameworks analyzed in Section 3.1.

4.1 Evaluated Programs

The first open-source application we analyzed is the Java version of *enNode2* [2]. A running instance of *enNode2* is part of the Exchange Network, which has the aim to exchange environmental information between different States, Territories, Tribes, and the U.S. Environmental Protection Agency. The second application, called *GSS* [1], is a file storage service used for the Greek research and academic community.

The closed-source application is made available by one of our partners and is a successful commercial business application that helps companies to declare goods electronically for the import and export. The software is offered to customers on a software as a service basis, which makes the confidentiality of the user data an important and not easy to ensure requirement, due to the size of the existing source code (over 600k LoC and more than 1000 JSP files).

The different frameworks of interest that are used by the analyzed programs are listed in Table 1. Within the table, “EJBs” stands for stateless enterprise session beans, not for all type of existing Enterprise JavaBeans. One can see that *enNode2* is a web application that does not use Enterprise JavaBeans, whereas *GSS* does not employ the Struts framework.

Furthermore, Table 1 shows the size of the different systems, which was measured with `sloccount` [2]. One can see that the proprietary software is the largest application, by far. Additionally, the table shows the time that was necessary

² <http://www.dwheeler.com/sloccount/>

Table 1. General information of the analyzed systems

Project	EJBs	JSPs	Servlets	Struts 1	LoC [k]	RFG [min]	Analysis [min]
enNode2		✓	✓	✓	85	2:27	0:43
GSS	✓	✓	✓		36	0:24	0:07
Commercial	✓	✓	✓	✓	614	4:42	1:51

to construct the RFG for the program and the time our analysis took. The construction and analysis time shows that our approach is applicable for real-world systems.

4.2 Analysis Results

In the previous section, we showed which frameworks were used by which programs and gave a rough size estimation. Next, we highlight in Table 2 to which degree the classes that are implemented are pooled by a framework. 24% of all implemented classes in *enNode2* are pooled, and 68% of these have at least one field. In *GSS* just 8% of all classes are pooled, but each of these classes contains fields. Finally, *Commercial* has a pooled rate of 30% and a quite high rate of 84% of the pooled classes that contain fields. In addition, Table 2 shows that the classes that contain fields have a medium rate of fields.

Table 2. Frequency of pooled classes

Project	Num. of Classes	Num. of Pooled Classes	Pooled with Fields	Avg. Num. of Fields
enNode2	438	107 (24 %)	68 %	6.48
GSS	164	14 (8 %)	100 %	6.29
Commercial	4901	1485 (30 %)	84 %	8.72

In total, we found 473 fields of pooled classes within the implementation of *enNode2*, 88 fields in *GSS* and 10894 in *Commercial*. To determine whether the results may lead to inter-session data flows, we grouped the fields according to their usage as described in Section 3.4 and show the results in Table 3.

All fields that belong to the groups “write only” and “no access” are false positives and would have been reported by the simple approach in contrast to our approach. Therefore, our approach removes at least between 30% and 69% of the reported findings.

Table 3. Results grouped by access type

	enNode2	GSS	Commercial
read only	54.12 %	30.68 %	57.35 %
write only	0.42 %	0.00 %	4.56 %
read and write	4.23 %	0.00 %	11.92 %
no access	41.23 %	69.32 %	26.16 %

4.3 Discussion

With our approach, we can detect fields of pooled classes that may lead to inter-session data flow. Our approach has a lower false positive rate than the naive one since it identifies fields that cannot transfer data between sessions (see Section 3.4).

Nevertheless, the findings reported by our approach still contain false positives³, since we just use control flow and field accesses for our detection. In order to improve the quality of our analysis, we are planning different enhancements that will be described in Section 5, such as a data-flow based approach. However, we were able to show with the help of our approach that the problem of inter-session data flow is relevant in the applications that we analyzed.

A possible reason for the quite high rate of unused fields may be the use of reflection, a typical source of false positives in static analysis. In some situations, it is not possible to determine which class is instantiated or which field is accessed. There is a work by Bodden et al. that uses run-time monitoring to track the aforementioned information and enhances the intermediate representation with the gathered details at analysis time to improve the results of static analysis [6].

To validate the results of our approach, we manually checked the assignment to the different groups of a part of our results and found that all checked findings were categorized correctly. During the checks, we noted that the classes generated by Jasper for the JSP files all contained generated fields that belonged to the read-only group. A manual inspection suggests that these fields do not transfer data between sessions and therefore are false positives that we cannot identify with our approach.

Beyond that, we inspected, in collaboration with the vendor, the reported problems of the commercial software. The vendor was able to identify several findings that have caused trouble in the past. In particular, customers complained that their data were messed up with data from other customers sporadically and the programmers were not able to identify the cause of the problem because they could not reproduce the symptoms, which the users reported. For some of our findings, we created automated reproducers that repeatedly called some functionality until the results indicated that the data do not belong to the current user. After removing the fields, we were unable to reproduce the erroneous behaviour which shows the effectiveness of our approach.

³ Nota bene: The naive approach reports the same false positives.

5 Conclusion and Outlook

We described the problem of inter-session data flows that may arise from the wrong usage of the object-pool pattern in the context of different Java Enterprise frameworks. Furthermore, we developed a light-weight approach to detecting such flows that produces a lower rate of false positive than a naive approach and showed that this kind of problems appeared in open-source as well as in proprietary software. The size of our case study is relatively small and must be increased in our future work. Nevertheless, our approach helped us to identify and remove existing security holes in a commercial software.

In order to improve our approach, we are planning to implement several refinements. First, we plan to automatically detect the object-pool pattern, which makes the preliminary manual inspection of the framework documentation superfluous. Second, we are going to track the flow of user-related data through a system and find out whether the data are stored in memory locations that may be accessed from different sessions, in contrast to our current mere control-flow based approach. This task is similar to taint checking, employed to find injection vulnerabilities. Last but not least, we want to exclude the cases where checks are implemented which make sure the data cannot flow, such as locking and resetting areas that access the field.

References

1. File storage service with REST-like API, rich web GUI, webDAV (November 2011), <http://code.google.com/p/gss/>
2. Open Source Exchange Network Node, supporting the National Environmental Exchange Network (November 2011), <http://code.google.com/p/en-node2/>
3. Anderson, P., Zarins, M.: The codesurfer software understanding platform. In: Proceedings of 13th International Workshop on Program Comprehension, IWPC 2005, pp. 147–148 (May 2005)
4. Ashcraft, K., Engler, D.: Using Programmer-Written Compiler Extensions to Catch Security Holes. In: Proceedings of the 2002 IEEE Symposium on Security and Privacy, pp. 143–159. IEEE Computer Society, Washington, DC (2002)
5. Bishop, M., Dilger, M.: Checking for Race Conditions in File Accesses. *Computing Systems* 9, 131–152 (1996)
6. Bodden, E., Lam, P., Hendren, L.: Clara: A Framework for Partially Evaluating Finite-State Runtime Monitors Ahead of Time. In: Barringer, H., Falcone, Y., Finkbeiner, B., Havelund, K., Lee, I., Pace, G., Roşu, G., Sokolsky, O., Tillmann, N. (eds.) RV 2010. LNCS, vol. 6418, pp. 183–197. Springer, Heidelberg (2010), <http://www.bodden.de/pubs/blh10clara.pdf>
7. Chess, B.: Improving Computer Security using Extended Static Checking. In: Proceedings of the 2002 IEEE Symposium on Security and Privacy, pp. 160–173. IEEE Computer Society, Washington, DC (2002)
8. DeMichiel, L.G.: Enterprise JavaBeans™ Specification, Version 2.1. Sun Microsystems (2003)
9. DeMichiel, L.G., Keith, M.: JSR 220: Enterprise JavaBeans™, Version 3.0. Sun Microsystems (2006)

10. Feiman, J., MacDonald, N.: Magic quadrant for static application security testing. Tech. rep., Gartner, Inc. (2010)
11. Graf, J.: Speeding up context-, object- and field-sensitive sdg generation. In: 2010 10th IEEE Working Conference on Source Code Analysis and Manipulation (SCAM), pp. 105–114 (2010)
12. Hammer, C.: Experiences with PDG-Based IFC. In: Massacci, F., Wallach, D., Zannone, N. (eds.) ESSoS 2010. LNCS, vol. 5965, pp. 44–60. Springer, Heidelberg (2010)
13. Hammer, C., Snelting, G.: Flow-Sensitive, Context-Sensitive, and Object-sensitive Information Flow Control Based on Program Dependence Graphs. *International Journal of Information Security* 8(6), 399–422 (2009)
14. Kircher, M., Jai, P.: Pooling. In: Proceedings of the 2002 European Conference on Pattern Languages of Programs (2002)
15. Krinke, J.: Identifying similar code with program dependence graphs. In: Proceedings of Eighth Working Conference on Reverse Engineering, pp. 301–309 (2001)
16. Livshits, B., Lam, M.S.: Finding Security Vulnerabilities in Java Applications with Static Analysis. In: Proceedings of the 14th USENIX Security Symposium, pp. 271–286 (2005)
17. Mordani, R.: Java™ Servlet Specification, Version 3.0 Rev a. Sun Microsystems (2010)
18. Nagy, C., Mancoridis, S.: Static Security Analysis Based on Input-Related Software Faults. In: Proceedings of the 2009 European Conference on Software Maintenance and Reengineering, pp. 37–46. IEEE Computer Society, Washington, DC (2009)
19. Oracle: Java EE at a Glance (November 2011), <http://www.oracle.com/technetwork/java/javae>
20. Raza, A., Vogel, G., Plödereder, E.: Bauhaus – A Tool Suite for Program Analysis and Reverse Engineering. In: Pinho, L.M., González Harbour, M. (eds.) Ada-Europe 2006. LNCS, vol. 4006, pp. 71–82. Springer, Heidelberg (2006)
21. Red Hat, Inc: Session EJB and MDB Configuration (2011), <http://docs.jboss.org/ejb3/docs/reference/build/reference/en/html/session-bean-config.html>
22. Reenskaug, T.: Models – Views – Controllers. Tech. rep., Xerox PARC (1979), <http://heim.ifi.uio.no/~trygver/1979/mvc-2/1979-12-MVC.pdf>
23. Roth, M., Pelegrí-Llopart, E.: JavaServer Pages™ Specification, Version 2.0. Sun Microsystems (2003)
24. Schumacher, M., Fernandez-Buglioni, E., Hybertson, D., Buschmann, F., Sommerlad, P.: Security Patterns: Integrating Security and Systems Engineering. John Wiley & Sons Ltd. (2006)
25. Souza, F., Arteiro, R., Rosa, N., Maciel, P.: Performance Models for the Instance Pooling Mechanism of the JBoss Application Server. In: IEEE International on Performance, Computing and Communications Conference, IPCCC 2008, pp. 135–143 (2008)
26. SpringSource: SpringSource.org. (November 2011), <http://www.springsource.org>
27. The Apache Software Foundation: Apache Struts (November 2011), <http://struts.apache.org>
28. The Apache Software Foundation: Apache Tomcat (November 2011), <http://tomcat.apache.org/>
29. Wassermann, G., Su, Z.: Sound and Precise Analysis of Web Applications for Injection Vulnerabilities. In: Proceedings of the 2007 ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2007, pp. 32–41. ACM, New York (2007)

Embedded Eavesdropping on Java Card

Guillaume Barbu^{1,2}, Christophe Giraud¹, and Vincent Guerin¹

¹ Oberthur Technologies

Technologies and Innovation,

4 allée du Doyen George Brus, 33600 Pessac, France

² Institut Télécom / Télécom ParisTech, CNRS LTCI,

Département COMELEC,

46 rue Barrault, 75634 Paris Cedex 13, France

{g.barbu,c.giraud,v.guerin}@oberthur.com

Abstract. In this article we present the first Combined Attack on a Java Card targeting the APDU buffer itself, thus threatening both the security of the platform and of the hosted applications as well as the privacy of the cardholder. We show that such an attack, which combines malicious application and fault injection, is achievable in practice on the latest release of the Java Card specifications by presenting several case studies taking advantage for instance of the well-known GlobalPlatform and (U)SIM Application Toolkit.

Keywords: Java Card, APDU Buffer, Fault Attack, Logical Attack, Combined Attack.

1 Introduction

When introduced in the mid-nineties, Java Cards revolutionized the development process for smart cards applications. Indeed before then, applications were always developed in a native way, i.e. by taking into account the specificities of the corresponding hardware on which the application is going to be executed. This meant in particular that if a developer wanted to execute the same application on several different devices, he had to develop as many implementations as devices. Java Cards on the other hand allow the developer to implement an application independently from the device on which it is going to be executed. Such an abstraction layer is provided by the Virtual Machine which interprets the Java code, called *bytecode*, and executes the corresponding instructions for a specific device. Therefore, executing a brand new Java Card application on each and every Java Card on the market costs only one development, leading to the very fast deployment of such an application which cannot be achieved when using native products. Moreover, Java Cards allow one to easily load new applications when the card is in the field whereas such a functionality extension is very difficult to achieve when using a native card.

Originally used in the mobile environment, Java Cards are now widely used in banking and identity environments where the constraints in terms of security

are very strong. Indeed, Java Cards are generally considered as intrinsically safer than native ones due to the security brought by the Java Card Runtime Environment which for instance constantly checks that objects of an application do not access objects of another application. However, as it is often the case when a new system appears, many attackers try to circumvent the inherent security of Java Card by using so-called *ill-formed applications*. To do so, the attacker modifies the binary representation of a Java Card application in order to allow it to access unauthorised objects [1–5]. Fortunately, such logical attacks can be counteracted by the use of a *bytecode verifier* [6, §4.9.2] whose aim is to ensure that an application is conform to the Java Card specifications [7–10]. In addition, implementors may operate certain verifications at run time in a so-called *defending virtual machine*, by opposition to the *offensive* ones which merely interpret the bytecode and totally rely on the security brought by the bytecode verification. However, such a verification was not mandatory up to the very recent Java Card 3.0 *Connected Edition* specifications which make the use of an on-card bytecode verifier compulsory. Therefore, each and every Java Card prior to the Java Card 3.0 *Connected Edition* is likely to be vulnerable to software attacks if the bytecode verification is not performed or if the embedded virtual machine does not implement additional security checks at run time.

At the same time as Java Cards were introduced, two new kinds of attacks specific to the embedded environment were published. These attacks take advantage of the *physical* properties of the embedded device on which the application is being executed. The first kind of these physical attacks, called *Side Channel Analysis*, takes into account the physical interactions between a device and its environment to obtain information about the secrets manipulated by the device. Examples of such interactions are the power consumption [11] or the electromagnetic radiation [12] of the device. The second kind of attacks, called *Fault Attacks*, aim to disturb the execution of an application. Such a disturbance could lead to a faulty output or to executing the application with granted privileges [13, 14]. Nowadays, the main mean of disturbing an embedded device is to use light beams [15] or electromagnetic pulses [16]. Physical attacks were mainly studied in the literature to break cryptographic implementations but they can also target any function implemented on embedded devices.

The idea to combine software and physical attacks appears recently in [17]. Such attacks, called *Combined Attacks*, aim at allowing a malicious application to bypass the security of Java Cards even using a bytecode verifier. Since then, many Combined Attacks have been published to attack several vital points of a Java Card such as the operand stack or the garbage collector [18–22].

In this paper, we use a Combined Attack to compromise another vital point of a Java Card which has not yet been targeted: the *APDU buffer*. This buffer is used to exchange all data that passes between the smart card and the terminal. It is therefore a central element of any smart card. In the following parts, we will show how an attacker can spy on the content of the APDU buffer or modify it through concrete examples on Java Cards. This study exhibits the fact that the developer must always take into account that the APDU buffer can

be compromised at any time and not only during communication with the card reader.

The rest of this paper is organised as follows. In Section 2, we detail the usage of the APDU buffer on smart cards before exposing the main specificities of this buffer in a Java Card. We also briefly present the characteristics of the two Java Card platforms which are described in the latest specifications, namely the *Classic* and *Connected* Editions. In Section 3, we show how a Combined Attack can allow an attacker to access the content of the APDU buffer during the execution of an application running on either Java Card *Classic* or *Connected* Edition. In Section 4 we present several cases studies, in particular to break Secure Channels. Finally Section 5 concludes the paper.

2 Forewords on the APDU Buffer and Targeted Platforms

The attack we propose is about the attacker's ability to illegally access the Application Protocol Data Unit buffer within an applet of her own. In this section, we start by discussing the possible usage of the APDU buffer and the potential security issues. Then we outline a couple of statements from the JCRE specification relative to the security of the APDU buffer in a Java Card platform. Finally, we present the two platforms we consider in the scope of this article.

2.1 The APDU Buffer Usage

At first glance, one could say that having hand over the APDU buffer is only a Man-In-The-Middle attack between the card and the card reader. However when looking more carefully, one can observe that the APDU buffer contains not only received or emitted data but it is also a temporary buffer that the application uses during its execution. We expose hereafter two examples which illustrate how powerful an adversary is compared to a Man-In-The-Middle attack when she can spy on and/or modify the APDU buffer.

The first example is based on *Secure Channel* which is the most common countermeasure to counteract Man-In-The-Middle. The principle of a Secure Channel is to share a key between the card and the reader and then to ensure confidentiality and integrity of the communication by using this key with cryptographic functions such as encryption or MAC verification for instance. For simplicity reasons and memory consumption optimisation, such operations are often performed in the APDU buffer. In such a case a Man-In-The-Middle cannot alter the exchanged data without being detected nor recover the sensitive information which are exchanged. However, an attacker having hand over the APDU buffer can not only spy on the communication, bypassing the confidentiality insurance, but she can also modify the command after the MAC verification leading to very powerful potential attacks.

A second example concerns the management of the APDU buffer during commands requiring a very large amount of memory space, such as asymmetric cryptographic computations for instance. In such a case, the developer can use the APDU buffer as a temporary buffer during the application execution to extend the memory space capability of the device. However, an attacker spying on the APDU buffer during the cryptographic computation can compromise the security of the system if sensitive values such as cryptographic keys are manipulated in this buffer. Moreover, if the attacker can modify the values manipulated in the APDU buffer, a logical fault can be injected on a temporary value leading to an erroneous cryptographic output. Such a faulty output can then be used to recover the corresponding cryptographic secret key by using Differential Fault Analysis [13].

The examples described above emphasize the strength of an attacker if she succeeds in spying on and modifying the APDU buffer during the execution of an application. In the following, we will present in more detail the specificities of the APDU buffer in the context of a Java Card.

2.2 Specificities of the APDU Buffer in a Java Card

On a Java Card platform, a global array is a particular type of array object that is owned by the JCRE but accessible by different applications. The APDU buffer object contains such an array which is accessible through the `javacard.framework.APDU.getBuffer()` virtual method. Indeed this array is the buffer containing the incoming APDU command and the outgoing APDU response.

The sensitive nature of such arrays is quite obvious since they are potentially shared amongst all applications. Therefore the JCRE specifications mandate several restrictions and verifications concerning its use.

Firstly, to prevent an application from accessing a global array when it should not, the following restriction applies:

"Accessing Class Instance Object Fields (...). Otherwise, if the bytecode is `putfield` and the field being stored is a reference type and the reference being stored is a reference to a temporary JCRE entry point object or a global array, access is denied." [10, §2.4.2.8]

Secondly, to avoid any data leakage from one application to another, the following statement is specified:

*"Because of the global status of the APDU buffer, it **MUST** be cleared to zeroes whenever an applet is selected, before the applet container accepts a new APDU command."* [10, §2.4.2.2]

In the following, let us present the two different Java Card platforms we consider in this paper.

2.3 Targeted Platforms

In its latest version, the Java Card standard has been divided into two different Editions: the *Classic* and the *Connected*. Let us briefly present below these two editions.

Java Card 3.0 Classic Edition The Java Card 3.0 *Classic* Edition appears as a regular update of the Java Card 2.2.2 standard. It is as of today the most widespread type of Java Card platform. Therefore several frameworks have been defined with a strong link with such platforms. This is the case for instance of the GlobalPlatform environment and the Card/(U)SIM Application Toolkits.

Java Card 3.0 Connected Edition. The Java Card 3.0 *Connected* Edition stands as the major evolution of the latest release of the Java Card specifications. It offers several new capabilities, such as an embedded web server, coming with standard network protocols, a widely upgraded API, or on-card bytecode verification. Besides the on-card bytecode verification, making an ill-formed-applet-based attack hardly possible, the main features used in the following of this article are the multithreading support and the notion of *Restartable Tasks*.

The multithreading consists in allowing the concurrent execution of several processes (*threads* of execution) on a given system. On a single-core system such as a smart card, multithreading is then typically implemented by alternatively giving the system resources to the different threads of execution.

In the scope of this article, the multithreading will be used through the definition of a restartable task, also introduced in the latest Java Card specifications. The notion of *Restartable Tasks* is based on a task registry in which an application can register/unregister tasks it wishes to launch automatically whenever the system is powered on. In particular, an application can register an object instance of a class implementing the interface `Runnable` (by extending the class `Thread` typically) into the registry by a call to the static method `TaskRegistry.register(Runnable t)`. Subsequently, the `run()` method of this object instance will be automatically executed in a new thread of execution every time the system starts up.

This section has introduced the APDU buffer and its specificities. As just seen, the specifications are aware of its sensitiveness, hence the quoted restrictions. In Section 3 we present different ways to overcome these restrictions depending on the targeted platform.

3 The APDU Buffer Storage Attacks

In this section we firstly detail a fault attack allowing an attacker to store the APDU buffer despite the JCRE restrictions. Subsequently, we show how to exploit such a capability to mount a full attack path on platforms implementing either the *Classic* or the *Connected* Editions of the Java Card 3.0 specifications.

3.1 A Fault Attack to Store the APDU Buffer

Attacks against Java Card platforms often take advantage of ill-formed applications loaded on-card without going through the bytecode verification process. We will describe below how the combination of a malicious, but yet well-formed, application with a single physical fault injection can allow an attacker to gain a permanent access to the APDU buffer array whatever Edition the Java Card implements.

As stated in Section 2.2, the JCRE is responsible for preventing an application from storing references to global arrays, and in particular to the APDU buffer array. Therefore, the JCRE must perform runtime checks to enforce this rule. Without loss of generality, we assume that the JCRE operates the check described in Listing 1.1 when executing a `putfield` instruction.

Listing 1.1. Detection of APDU buffer storage attempt in `putfield`

```
// ref points to the object to store
if (isGlobalArray(ref)) {
    // Handle storage attempt
    throw SecurityException
}
```

Obviously, the aim of the attacker is to force the jump in the *else-branch* in our case. Since such a disturbance can be achieved thanks to a fault injection [13], the attacker can run within her own applet a method trying to store the reference of the APDU buffer array into a global array and disturb the conditional branching execution to avoid the `SecurityException`.

Let us assume that the attacker has been successful with the fault injection. As a result, she has been able to store the reference of the APDU buffer into a non-volatile field. Using this field, she is then likely to access the APDU buffer at any time. The following shows how such a capability can be exploited by an attacker on the Java Card 3.0 *Classic* and *Connected* Editions.

3.2 Attacking a Java Card 3.0 Classic Edition

By using the attack presented in Section 3.1, we assume that the attacker is able to access the APDU buffer at any time. However, without any interaction with other entities on-card, she cannot take advantage of this privilege in other ways than accessing the command and response of her own application.

In order to be able to exploit her new facility, the attacker's application must be given a chance to run when the APDU buffer is meant for another application. It is therefore necessary that it exposes one or several method(s) that might be called by another entity on the platform through shareable interfaces. The point is that when called, these shared method would allow the attacker to read or corrupt the APDU buffer "belonging" to the entity calling the method.

One can think that such a situation where an entity on-card calls the shared method of another applet would only appear in an attack proof of concept.

However, we demonstrate in Section 4 that different contexts can lead to this situation in practice.

3.3 Attacking a Java Card 3.0 Connected Edition

In the following, we show how an attacker can take advantage of the Java Card 3.0 *Connected* Edition multithreading facility to exploit the privilege of accessing the APDU buffer whenever she wants.

Considering the attacker has been able to store the APDU buffer, we can then imagine a restartable task whose `run()` method infinitely loops and spies upon the APDU buffer, such as detailed in Listing 1.2.

Listing 1.2. The eavesdropping restartable task

```

/**
 * - fieldBuf is the stored APDU buffer reference.
 * - BUF_LEN is the assumed APDU buffer length.
 * - tmpBuf is a byte array initialized with a size of BUF_LEN.
 * - os is an OutputStream used by the attacker
 */
public void run() {
    while (true) {
        // APDU buffer is different, copy its content.
        if (arrayCompare(fieldBuf, 0, tmpBuf, 0, BUF_LEN) != 0) {
            System.arraycopy(fieldBuf, 0, tmpBuf, 0, BUF_LEN);
            os.write(tmpBuf);
        }
    }
}

```

The attacker is then potentially able to dump every byte written in the APDU buffer inside her `run` method. Moreover, she can also modify the content of the APDU buffer by writing into instead of copying it.

We have seen how a Combined Attack on a Java Card can allow an attacker to spy the content of the APDU buffer during the execution of an application. In the following, we expose different case studies based on this capability.

4 Case Study

In this section, we exhibit two case studies from two important specifications of the Java Card ecosystem, namely the GlobalPlatform (GP) environment [23, 24] and the Card and (U)SIM Application ToolKit (CAT/(U)SAT) [25, 26]. In addition, we detail a possible exploitation of the restartable task described in Section 3.3.

4.1 Attacking through the GP Environment: OPEN

GP is an entity developing and publishing specifications relative to the deployment and management of embedded applications on secure chip technologies.

As part of the GP specifications [23], we find the description of the GP environment: the OPEN.

The GP Environment: OPEN As per [23], the OPEN is the on-card entity responsible for command dispatch, card content management operations, security management operations and secure inter-application communication. According to this last responsibility, the OPEN including its contactless extension [24], is the GP entity that sends notifications to other on-card entities when certain events occur. In order to keep track of the different on-card entities, the OPEN owns and uses an internal GP registry as an information resource. This registry contains information for managing the card, executable load files, applications, *Security Domain* associations, and privileges.

Shareable Interface Method Call from the OPEN. The event notifications are operated through calls to a shareable interface method. The only limitation is then for the attacking application to register for such notifications. In the scope of GP's contactless services, applications that implement the `CLApplet` interface shall be notified of changes occurring to their GP registry entry. These changes can have various origins, depending on the Contactless Registry Service (CRS). For instance, the application is notified of its installation on the platform, of the modification of the contactless communication protocol, etc... (the complete list of events can be found in [27]). These notifications are implemented by calls to the `notifyCLEvent` method of this interface.

Therefore, the applet detailed in Listing 1.3, which has gained a permanent access to the APDU buffer array as described in Section 3, is likely to analyse its content each time the registry entry of the applet is updated. As such updates occur quite often, the attacker is then able to access frequently to the APDU buffer. One could note that these updates of the GP registry are

Listing 1.3. APDU analysis on event notification when the attacking applet implements `CLApplet`

```

public class MyApplet extends Applet, implements CLApplet {
    ...
    public void notifyCLEvent(short event) {
        analyseAPDU(); // using the stored reference
    }
}

```

typically privileged operations performed by the `CRSApplication`. Therefore, the data potentially contained in the APDU buffer is likely to be particularly sensitive. This could be for instance data having led to a successful authentication or granted authorization.

In this section we have shown how the access to the APDU buffer could lead to gain sensitive information relative to the security of both the platform and the hosted applications. The following section describes how the privacy of the card holder can also be threatened.

4.2 Attacking through the CAT/(U)SAT

Mobile communication is in constant evolution since the early 90s. With the well known (U)SIM card and UICC (resp. for (Universal) Subscriber Identity Module and Universal Integrated Circuit Card), it is today the most important market in the smart card industry. The CAT [25] and (U)SAT [26] are standards from the mobile communication. Their main goal is to define how the smart card should interact with the outside world and initiate commands independently of both the handset and the network. We show in the following that these can be misused by an attacker with the APDU buffer access privilege.

The CAT Runtime Environment and the (U)SAT Framework. As part of these toolkits, the CAT and (U)SAT Application Programming Interfaces (APIs) for Java Card are respectively specified in [28] and [29]. Java Card toolkit applets are then likely to control access to the network, displaying menus on the handset, etc...

These features are achieved thanks to the *Toolkit Registry*, which similarly to the GP registry defined in the previous section, allows a *Toolkit Applet* to register to events fired by the runtime environment.

Eavesdropping and Corrupting the Short Message Service. As for the attack described in the previous section, event notifications are operating through calls to a shareable interface method. In order to register to some events, an applet must implement the interface `uicc.toolkit.ToolkitInterface` and call the `setEvent(short event)` method of its `ToolkitRegistry` instance (available by a call to `uicc.toolkit.ToolkitRegistrySystem.getEntry()`). Subsequently, the implemented `processToolkit(short event)` will be triggered each time an event to which the applet is registered occurs.

In the context of the attack we describe, the attacker has then all the reasons to register to all possible events, in order to have her eavesdropping method called as often as possible. In particular, we study the case of events associated to the reception of a short message through the Short Message Service (SMS). The attacker's toolkit applet is described in Listing 1.4.

Provided, short messages are located in the APDU buffer, the attacker is able to intercept them and to either redirect them to the outside world or modify their content as she pleases. This is indeed one of the many ways the attacker can use the APDU buffer in this context. Other potential applications can also take advantage of the pro-active capability of the CAT environment to redirect outgoing messages or calls to taxed services for instance.

The two previous case studies were targeting the Java Card 3.0 *Classic* Edition and earlier. The next one described how the so-called eavesdropping restartable task can be exploited on a Java Card 3.0 *Connected* Edition.

4.3 Attacking through the Eavesdropping Restartable Task

In this section we depict a scenario where the attacker is able to eavesdrop or tamper with the communication (even if secured by a secure channel) and

Listing 1.4. Eavesdropping and corrupting the SMS

```

public class MyApplet extends Applet,
                       implements ToolkitInterface {
    ToolkitRegistry r;
    public MyApplet() {
        r = ToolkitRegistrySystem.getEntry();
        ...
        r.setEvent(ToolkitConstants.EVENT_UNFORMATTED_SMS_PP_UPD);
    }
    public void processToolkit(short ev) {
        if(ev == ToolkitConstants.EVENT_UNFORMATTED_SMS_PP_UPD){
            analyseAPDUSMS(); // using the stored reference
        }
        ...
    }
}

```

temporary data. In both cases we can use the result presented in [21], exploiting I/O flooding to force a thread scheduling at a specific time. As a consequence, the attacker finds herself in the situation described in Section 2.1 where she can access the APDU buffer almost whenever she pleases. In the following, we detail a case study proving the potential threat of such a situation.

Breaking the Secure Channel. As stated in Section 2.1, a Secure Channel is a mechanism provided by GP to ensure both the confidentiality and integrity of the terminal-card communication through cryptographic mechanisms. We show here that the restartable task we have introduced in Section 3.3 can be used to break a Secure Channel.

The initialisation of a Secure Channel is made thanks to two APDU commands, namely INIT_UPDATE and EXT_AUTHENTICATE, with specific CLA and INS bytes (respectively 80 50 and 84 82). Therefore, the eavesdropping task can detect the beginning of a Secure Channel session by detecting these commands. Subsequently, the deciphering (resp. ciphering) and MAC checking (resp. computing) of an incoming (resp. outgoing) APDU is operated thanks to a call to the method `unwrap(byte[] baBuffer, short sOffset, short sLength)` (resp. `wrap(byte[] baBuffer, short sOffset, short sLength)`) of the `SecureChannel` interface. Our point is that if this method is called with the APDU buffer array as parameter, the attacker owning the restartable task will be able to both eavesdrop and corrupt the communication. The attack scenario is depicted in Figure 1.

The `SecureChannel` is indeed used in numerous applications in all smart card fields of application, from finance to health-care. If deemed successful, the described attack would have serious consequences regarding security and privacy.

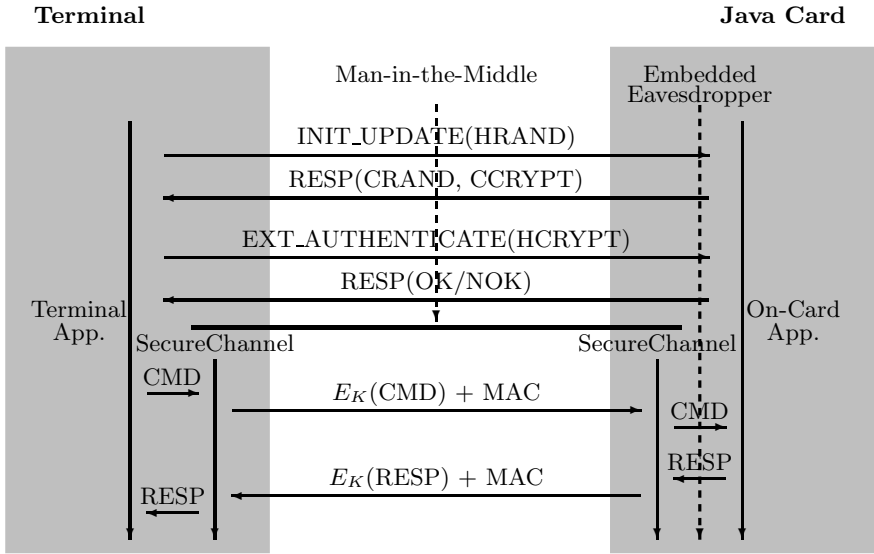


Fig. 1. Breaking the SecureChannel with the Eavesdropping Restartable Task

5 Conclusion

In this article, we have introduced a novel Combined Attack tampering with the Application Protocol Data Unit buffer. This attack leads to an outstanding privilege: accessing the APDU buffer array at any time. This attack was motivated by the fact that the APDU buffer is indeed far from being only the communication channel between the card and the terminal.

In order not to limit the range of the attack, we have described different ways to take advantage of this privilege on platforms implementing both the *Classic* and *Connected* Editions of the Java Card 3.0 specifications. Finally, we have exhibited practical exploitations of the attack on both platforms using widely spread frameworks (the GP API and the CAT/(U)SAT API) for the first platform and the multithreading facility for the second.

These exploitations highlight the crucial necessity to protect the access to the APDU buffer array, by taking into account especially attackers with fault injection capability.

References

1. Witteman, M.: Java Card Security. Information Security Bulletin 8, 291–298 (2003)
2. Mostowski, W., Poll, E.: Malicious Code on Java Card Smartcards: Attacks and Countermeasures. In: Grimaud, G., Standaert, F.-X. (eds.) CARDIS 2008. LNCS, vol. 5189, pp. 1–16. Springer, Heidelberg (2008)
3. Sere, A.A., Iguchi-Cartigny, J., Lanet, J.L.: Automatic Detection of Fault Attack and Countermeasures. In: WESS 2009, pp. 1–7 (2009)
4. Hogenboom, J., Mostowski, W.: Full Memory Attack on a Java Card. In: 4th Benelux Workshop on Information and System Security (2009)
5. Iguchi-Cartigny, J., Lanet, J.L.: Developing a Trojan Applet in a Smart Card. *Journale on Computers and Virology* 6, 343–351 (2010)

6. Lindholm, T., Yellin, F.: Java Virtual Machine Specification, 2nd edn. Addison-Wesley, Inc. (1999)
7. Sun Microsystems Inc.: Virtual Machine Specification – Java Card Platform, Version 3.0.1 (2009)
8. Sun Microsystems Inc.: Application Programming Interface, Java Card Platform, Version 3.0.1 Connected Edition (2009)
9. Sun Microsystems Inc.: Java Servlet Specification, Java Card Platform, Version 3.0.1 Connected Edition (2009)
10. Sun Microsystems Inc.: Runtime Environment Specification, Java Card Platform, Version 3.0.1 Connected Edition (2009)
11. Kocher, P., Jaffe, J., Jun, B.: Differential Power Analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999)
12. Gandolfi, K., Mourtel, C., Olivier, F.: Electromagnetic Analysis: Concrete Results. In: Kog, Ç.K., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 251–261. Springer, Heidelberg (2001)
13. Giraud, C., Thiebeauld, H.: A Survey on Fault Attacks. In: CARDIS 2004, pp. 159–176. Kluwer Academic Publishers (2004)
14. Bar-El, H., Choukri, H., Naccache, D., Tunstall, M., Whelan, C.: The Sorcerer’s Apprentice Guide to Fault Attacks. IEEE 94, 370–382 (2006)
15. Skorobogatov, S., Anderson, R.: Optical Fault Induction Attacks. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 2–12. Springer, Heidelberg (2003)
16. Quisquater, J.J., Samyde, D.: Eddy Current for Magnetic Analysis with Active Sensor. In: e-Smart 2002 (2002)
17. Barbu, G.: Fault Attacks on Java Card 3 Virtual Machine. In: e-Smart 2009 (2009)
18. Barbu, G., Duc, G., Hoogvorst, P.: Java Card Operand Stack: Fault Attacks, Combined Attacks and Countermeasures. In: Prouff, E. (ed.) CARDIS 2011. LNCS, vol. 7079, pp. 297–313. Springer, Heidelberg (2011)
19. Barbu, G., Hoogvorst, P., Duc, G.: Application-Replay Attack on Java Cards: When the Garbage Collector Gets Confused. In: Scandariato, R. (ed.) ESSoS 2012. LNCS, vol. 7159, pp. 1–13. Springer, Heidelberg (2012)
20. Vetillard, E., Ferrari, A.: Combined Attacks and Countermeasures. In: Gollmann, D., Lanet, J.-L., Iguchi-Cartigny, J. (eds.) CARDIS 2010. LNCS, vol. 6035, pp. 133–147. Springer, Heidelberg (2010)
21. Barbu, G., Thiebeauld, H.: Synchronized Attacks on Multithreaded Systems - Application to Java Card 3.0 -. In: Prouff, E. (ed.) CARDIS 2011. LNCS, vol. 7079, pp. 18–33. Springer, Heidelberg (2011)
22. Barbu, G., Thiebeauld, H., Guerin, V.: Attacks on Java Card 3.0 Combining Fault and Logical Attacks. In: Gollmann, D., Lanet, J.-L., Iguchi-Cartigny, J. (eds.) CARDIS 2010. LNCS, vol. 6035, pp. 148–163. Springer, Heidelberg (2010)
23. GlobalPlatform Inc.: GlobalPlatform Card Specification 2.2.1 (2011)
24. GlobalPlatform Inc.: GlobalPlatform Card Specification 2.2, Amendment C, Contactless Services (2010)
25. European Telecommunications Standards Institute: Card Application Toolkit (CAT) (Release 10) (2011)
26. European Telecommunications Standards Institute: Universal Subscriber Identity Module (USIM) Application Toolkit (USAT) (Release 10) (2011)
27. GlobalPlatform Inc.: Java Card Contactless API and Export File for Card Specification v2.2.1 (org.globalplatform.contactless) v1.0 (2011)
28. European Telecommunications Standards Institute: UICC Application Programming Interface (UICC API) for Java Card (Release 9) (2011)
29. European Telecommunications Standards Institute: (U)SIM Application Programming Interface ((U)SIM API) for Java Card (Release 10) (2011)

Authenticated Key Exchange (AKE) in Delay Tolerant Networks

Sofia Anna Menesidou and Vasilios Katos

Information Security and Incident Response Unit
Department of Electrical and Computer Engineering
Democritus University of Thrace
University Campus, Xanthi 67100, Greece
{smenesid,vkatos}@ee.duth.gr
<http://isir.ee.duth.gr/>

Abstract. Key exchange is considered to be a challenging problem in Delay Tolerant Networks (DTNs) operating in space environments. In this paper we investigate the options for integrating key exchange protocols with the Bundle Protocol. We demonstrate this by using a one-pass key establishment protocol. In doing so, we also highlight the peculiarities, issues and opportunities a DTN network maintains, which heavily influences the underlying security solution.

Keywords: Bundle Security Specification.

1 Introduction

Delay or Disruption Tolerant Networks (DTNs) are becoming popular both in terrestrial and deep space environments as they maintain certain advantages over traditional internetworking protocols such as TCP/IP. The benefits of adopting DTN technologies are clear in environments where connectivity in terms of end to end path availability cannot be guaranteed for the lifetime of a communications session.

Although DTNs by nature may support high availability (which in DTN terminology is referred to as reliability), they are not short of security issues. This is primarily due to the constraints of the unwelcoming and hostile in terms of communication environments the DTNs operate in. The three main limitations composing a typical space internetworking environment are the limited bandwidth, the relatively high bit error rates and the periods lacking connectivity where in some cases open loop communications is the only option.

The limited bandwidth dictates that the overheads should be kept to a minimum. As such, elaborate and message-rich cryptographic protocols are not suitable for deep space DTN applications. The integrity issues introduced by the high

bit error rates are mainly accidental rather malicious by nature, and therefore cryptographic integrity checksums could be simplified. Lastly, the large delays in principle make interactive and many-pass protocols unsuitable. Interactive security protocols involve a series of computations to be performed by all participating entities in an asynchronous yet orderly manner. There is a wealth of efficient protocols in the literature which cannot always be adopted due to the limitations of the environment. However, it seems that making assumptions that allow a limited use of these protocols, one can establish a security context on an higher initial cost (in terms of bandwidth) and then leverage the arranged setup to perform non-interactive type of security protocols without any significant decrease of security. For example, public key cryptography and protocols based on Diffie Hellman type of exchanges are not suitable for an ongoing and regular use, but limiting their invocation at the beginning of an association between the parties would result to a system offering practical and acceptable level of security. This is possible as the network topology in a space internetwork is fairly fixed.

On the other hand, certain security assumptions do not necessarily hold in a space internetworked environment. A fixed topology mentioned above requires a significant physical effort to change and as a result opportunities for a Man-In-The-Middle attack between two trusted nodes are rather slim. Yet, in DTN environments and particularly in deep space communications where each and every opportunity for sending data should be exploited to the highest possible means for economic reasons (amongst others) there may be a situation where it would be feasible and preferable to send data through a DTN node which is not trusted. In such case the sender is knowingly sending her data through a man in the middle which may or may not behave maliciously. In terms of DTN and deep space communication, a malicious action by the adversary targets confidentiality and/or integrity of the data; availability is generally treated by the DTN itself.

All the above suggest that the security goals for a space internetworked environment should be carefully selected and prioritized in order to select the most suitable authenticated key establishment protocol. This paper studies the assumptions and requirements for selecting a suitable AKE protocol in space internetworking applications against the limitations, opportunities and particular issues that apply in such environments.

2 Related Work

The area of key management in delay tolerant networks is relatively new and many research challenges remain to date; the DTN Research Group acknowledges key management as an open issue [10]. Traditional key management and AAA-like architectures are not suitable for DTN networks due to the environment limitations and technical constrains [3]. The work done until now is based on the assumption of shared keying material [26]. However, no method for automatic key distribution or agreement is yet defined within the bundle architecture.

The author in [10] states some requirements for key management in delay tolerant networks but no solution is yet proposed. Until now, a few solutions have been proposed to address this problem.

The authors in [2] introduce a solution based on Identity-Based Cryptography (IBC). IBC is a cryptographic method that enables message encryption and signature verification using a public identifier. In [15] the authors use the non-interactive Sakai-Ohgishi-Kasahara (SOK) key agreement scheme which is based on Boneh-Franklin IBC scheme. However, such IBC solutions appeared to superficially solve the problem [11].

In [4] the authors present a number of security goals and attributes for a key agreement protocol. In the same work they compare the key agreement protocols based on the intractability of Diffie-Hellman problem such as the ephemeral and static Diffie-Hellman, the KEA, the Unified Model and the MQV protocols. They also compare the one-pass version of these protocols. One-pass AK protocols are more efficient because they use only one message transmission. However such protocols have some security drawbacks because they do not offer known-key security and forward secrecy [4]. Another survey of the existing key establishment protocols is the work in [23]. The authors describe and compare a number of protocols using both symmetric and asymmetric techniques.

The work in [5] proposed one-pass authenticated key establishment protocol based on the Bellare Rogaway model for one-way communications. Their scheme is a slight adaptation on the basic elliptic curve Diffie-Hellman (ECDH) protocol with an authentication mechanism based on bilinear pairings. Even though their scheme is the strongest against the general key-compromise impersonation (K-CI) attack compared to one-pass versions of MQV, HMQV and CMQV, the use of bilinear pairings, makes this scheme less efficient [6]. Work in [19] proposes a two-pass authenticated key agreement protocol with key confirmation (2P-AKACP) and the one-pass version of this protocol (1P-AKACP) for one-way communications. Both protocols are based on the discrete logarithm problem (DLP) and have three phases: the registration, the transfer and verification, and the key generation. In addition the security and the computational complexities of these schemes outperform the protocols that are based on [22], [8,9]. However their claim proved incorrect and a several types of attacks presented in both protocols [7].

More recently, the author in [25] presents a dynamic and non-interactive key management for opportunistic networks using the Bundle Security Protocol (BSP). This means that the key management scheme will be used to derive keys for HMAC-SHA1 authentication, RSA digital signature and AES encryption which are the cipher-suits of BSP. Their scheme is based on the bilinear mappings over elliptic curves. In [14] the authors propose a dynamic virtual digraph (DVD) model for DTN public key distribution. They heuristically define the DVD model by extending the traditional graph theory and they also propose a two-channel public key distribution scheme.

3 Authenticated Key Establishment

3.1 The Setting

A representative scenario is depicted in Fig. 1. Assume that the rover A, satellite C and ground station belong to Organisation A, whereas satellite B belongs to Organisation B. All devices are DTN enabled and nodes B and C can serve as intermediary routers. In deep space DTN terms, loss of connection availability can be accurately predicted and routing decisions can be planned in advance. As such, the sending node will be able to make risk assessment decisions and adopt the appropriate security controls. In our example scenario there is no line of sight between the rover and the trusted satellite C - hence no available communications channel - and therefore the former needs to route through the untrusted satellite B. An example security policy would require that the contents of the payload must not be available to node B, so confidentiality must be supported.

Encryption can be triggered on an intermediary node, if such node schedules (routes) transmission of the data through an untrusted node. Unlike conventional TCP/IP, each DTN node implements the Bundle Protocol where a *custodian* of the data is defined and the data may reside on the DTN router for a large amount of time. Therefore, while the data is arranged to be transmitted some time in the not-so-close future (in TCP/IP timings), the router/custodian of the data may have the option to further process the payload *in situ*. In fact, an optimal solution could involve a source sending immediately the plaintext data to its neighboring trusted node if there is a transmission window of opportunity, and some custodian along the transmission path may decide to encrypt the data. Once the data is encrypted, the decryption should be expected to take place at the DTN endpoint. The decision to encrypt the data could be influenced by the routing as mentioned earlier, but also by the QoS conditions. For example, there may be a security policy to encrypt data by default when they follow a certain path, but this requirement could be overridden if there are data that need to be sent urgently (that is, a preference of QoS over confidentiality), since a missed opportunity to transmit the data may result to long, unacceptable delays. These peculiarities appear in a DTN environment and introduce interesting challenges and issues surrounding the selection and application of cryptographic protocols.

One of the main challenges yet to be addressed in such environments is key management. The recently published Bundle Security Protocol Specification [24] does not cover key management and the authors explicitly state that such exclusion is a result of an informed decision. The analysis and proposed solution that follows is an attempt to identify the constraints and requirements of the described scenario above and to suggest a suitable set of security protocols for the key transport problem and more specifically for authenticated key establishment.

3.2 Preliminaries, Goals and Requirements

As already mentioned in the scenario above, confidentiality is the security requirement that must be fulfilled. In order to succeed that we need an authenticated key agreement protocol between the rover A and the satellite B. However,

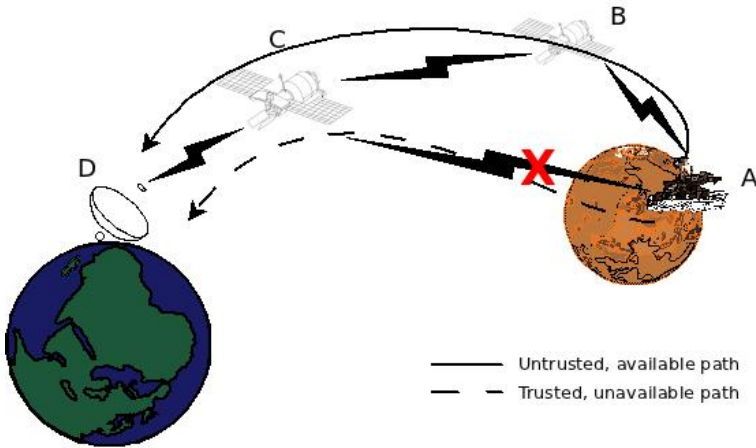


Fig. 1. A deep space communications example

we assume that both entities have pre-established long term keys. Based on this scenario the protocol we propose must satisfy a key agreement mechanism of the ISO/IEC 11770 [13]. The first time both entities will agree on a shared secret key, a three-pass protocol of the key agreement mechanism 10 of the ISO standard will be used. This mechanism uses elliptic curve cryptography to establish a shared secret with mutual implicit authentication. The next time both entities are going to establish a shared secret key based on mechanism 2 which is an one-pass key agreement protocol that establishes a shared secret to both entities with implicit key authentication but no entity authentication.

The key agreement protocols we selected for comparison are the Key Exchange Algorithm (KEA) [20], the one-pass version (KEA1) and the version with key confirmation (KEAA), the Unified Model (UM) [1], the one-pass version (UM1) and the version with key confirmation (UMA), the Menezes-Qu-Vanstone (MQV) [18], the one-pass version (MQV1) and the version with key confirmation (MQVA), the Revised Nyberg-Rueppel protocol (RNR) [21], the one-pass Authenticated Key Agreement with key confirmation protocol (1P-AKACP) and the two-pass version (2P-AKACP) [19], the Chalkias-Hristou-Stephanides-Alexiadis protocol (CHHSA) [5] and the Horster-Michels-Petersen protocol (HMP) [12]. From the above protocols the Nyberg-Rueppel (but not the revised version) and the Horster-Michels-Petersen protocol supports message recovery.

We adopt the definitions, attributes and requirements for Authenticated Key Establishment from [4]. In Tables 1 and 2 we present a summary of the candidate protocols against these definitions and attributes.

3.3 The Protocol

As already pointed out, the session key can be renewed with one-pass authenticated key exchange protocol. For instance, when a node A (security-source)

Table 1. One-pass Protocol comparison

	KEA1	UM1	MQV1	RNR	1P-AKACP	CHSA	HMP
Fundamental security goals							
Implicit key authentication	Y	Y	Y	Y	Y	Y	Y
Explicit key authentication	N	N	N	Y^I	N	N	N
Desirable security attributes							
Known-key security	N	N	N	N	N	N	N
Forward secrecy	N	N	N	N	N	N	N
Key-compromise impersonation	N ^[5]	N ^[5]	N ^[5]	-	N ^[7]	Y	-
Unknown key-share	N^+ ^[5]	Y	N^+ ^[5]	-	Y	Y	-
Desirable performance attributes							
Minimal number of passes	1	1	1	1	1	1	1

Y^I : Yes only to Initiator

N^+ : assurance is not provided unless modifications are made

Table 2. Multiple-pass Protocol comparison

	KEA	UM	MQV	2P-AKACP	KEAA	UMA	MQVA
Fundamental security goals							
Implicit key authentication	Y	Y	Y	Y	Y	Y	Y
Explicit key authentication	N	N	N	Y	Y	Y	Y
Desirable security attributes							
Known-key security	Y	Y	Y	Y	Y	Y	Y
Forward secrecy	N	Y	Y	Y	N	Y	Y
Key-compromise impersonation	Y	N	Y	Y	Y	N	Y
Unknown key-share	N ^[17]	Y	N	Y	Y	Y	N ^[16]
Desirable performance attributes							
Minimal number of passes	2	2	2	2	3	3	3

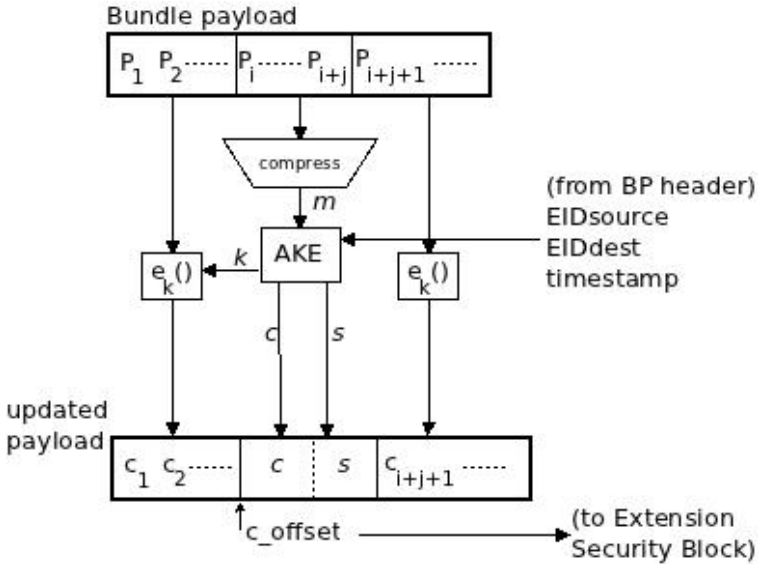
wants to send some data to node D (security-destination) with a new session key, k , the most efficient way to do it is to transmit the data and the new k simultaneously in the same message. More specifically, the main idea is to use an asymmetric authenticated encryption with message recovery technique to encrypt the protocol's parameters of the new k and with this session key to encrypt the data, provided that the offset of the parameters displayed in the transmitted message. The security-destination will be able to recover the new k and to decrypt the transmitted data with the recovered k .

We propose an adoption of the Horster-Michels-Petersen [12] protocol. We selected this protocol over its main competitor Nyberg-Rueppel because it has a lower communication cost. This is mainly due to the fact that HMP does not offer non-repudiation which is not a requirement in our scenario. Based on HMP protocol security-source A creates a compressed message m , computes the parameters c and s and sends $(c, s, \{\text{data}\}_k, \text{timestamp}, EID_{\text{source}}, EID_{\text{destination}})$ to the security-destination D. The new session key calculated as $k = h(m, \text{timestamp}, EID_{\text{source}}, EID_{\text{destination}})$, where timestamp is a field of the primary bundle block (bundle header) and EID_{source} and

Table 3. AKE protocol

1.	A: $m \in Z_p$ generates $n \in Z_p$ secretly and randomly computes $c = h(g_D^n)^{-1}m \bmod p$ computes $c' = c \bmod q$ computes $s = n - x_A c' \bmod p$ $k = h(m, \text{timestamp}, EID_{\text{source}}, EID_{\text{destination}})$
2.	A \rightarrow D: $(c, s, \{\text{data}\}_k, \text{timestamp}, EID_{\text{source}}, EID_{\text{destination}})$
3.	D: computes $c' = c \bmod q$ recovers $m = h(g_B^s g_A^{c' x_B})c \bmod p$ $k = h(m, \text{timestamp}, EID_{\text{source}}, EID_{\text{destination}})$

$EID_{\text{destination}}$ are fields of the bundle payload block. The aforementioned parameters c and s can be incorporated at the bundle payload filed among the encrypted data created with the new session key. We can also use the Bundle Extension Block (ESB) to keep information such as the offsets of parameters c and s . Node D will be able to distinguish c and s because of the offsets and to recover the message. Consequently D can calculate the k and decrypt the data. Assume that the security aware communicating parties have agreed on the public parameters g, Z_p in the standard Diffie Hellman fashion and q is a divisor of $p - 1$. Entities A and D have ephemeral keys x_A and x_D respectively which correspond to their public keys $g_A = g^{x_A}$ and $g_D = g^{x_D}$. Table 3 summarises the AKE protocol.

**Fig. 2.** AKE protocol within the bundle

The process for integrating the above protocol steps with the DTN architecture is presented in Fig. 2. Following the conventions shown in Fig. 1 earlier, assume that A (with public key g^A) needs to send data to D. Now since A knows that the data will need to be transferred to B who is untrusted in terms of confidentiality but, in DTN terms, will serve as the custodian, she would need to encrypt the data. In our example there are two alternatives:

1. A has exchanged public keys with the destination D, g^D .
2. A has only exchanged public keys with its immediate trusted neighbours in this case C, g^C .

From A's view the application of the security protocol and the resulting computational effort will be the same in both cases. The difference is that the protocol will be completed by a different party, D or C respectively. The Bundle Security Protocol (BSP) Specification has a data structure that allows seamless integration with either case. More specifically, the BSP contains the Abstract Security Block Structure (ASBS) where the security source and security destination endpoint identifiers are defined. In case (1) the security destination will contain the ID of D, whereas in case (2) the security destination will refer to the ID of C. This distinction is crucial for two main reasons. First, the specification states that upon receiving a secure bundle, a security destination may need to take some actions. For example, if C is the security destination, but not the ultimate destination of the bundle, this may mean that C may need to decrypt the encrypted payload, re-encrypt it, or even disclose the session key to the destination. The precise action will depend on whether there are other untrusted nodes further down the transmission path, the capabilities of the final destination node or even if there are planned delays for forwarding the bundle. For example, a node in space may be aware that a window of opportunity for sending the bundle may be in say, after half a day, so it could perform decryptions while the data are under its custody. Alternatively, a node may prefer for similar reasons to expedite the transmission of the bundle, as it may know that not doing so may be a missed opportunity, asking for the next custodian to perform the encryption.

Second, the source or custodian of the bundle may not have had the opportunity to share long term keys with the destination, or in the case of PKIs, it may not have sufficient and timely access to the public keys. In a sense, a data source may delegate another trusted node to perform the encryption of the data. Therefore practical flexibility is vital in an environment with extremely large delays. An example activity diagram of the optimisation logic and encryption decision making is shown in Fig. 3.

3.4 Evaluation

Although the HMP protocol has low communication overhead by design, in DTN space environments where bandwidth remains an issue, any savings on the bits transferred may have a significant impact on the overall communication quality. We have integrated the HMP protocol with the Bundle Protocol, but

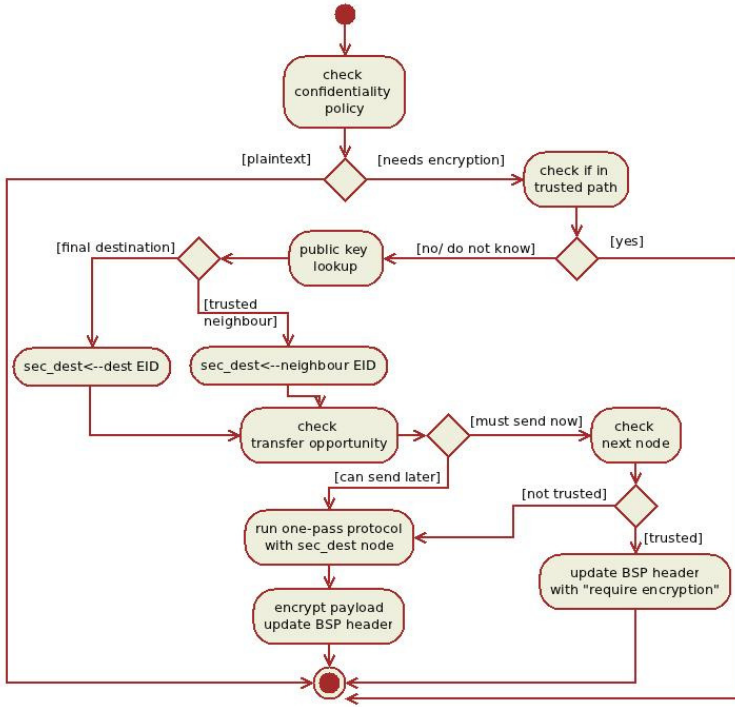


Fig. 3. Custodian's example encryption decision making activity diagram

this approach can be ported to a number of other published protocols. The design specification of the BP header as well as its extension blocks that use variable length attributes with self delimiting type of encoding, allows a variety of combination of protocols supporting a wide range of primitives. From a security perspective this is a very important feature as protocol updates in the DTN infrastructure will avoid security degradations. As such, our proposed scheme inherits the weaknesses of the underlying protocol.

The proposed approach includes the injection of the protocol messages in the payload and more specifically as part of the message. This is feasible for authenticated key exchange protocols with message recovery and the reason for doing so is purely for higher bandwidth utilization. As this message is also used for producing key material, the corresponding plaintext data need to be compressed to obtain high entropy and increase the corresponding effective key length. More precisely, the only inflation to the payload due to the cryptographic protocol is due to c . A side effect for this compression would be the lower communication costs, but this advantage can be easily lost if we introduce some further redundancy (say by means of a cryptographic checksum), according to the requirements of the specific, HMP protocol we have selected. It should be noted, however, that compression may also be a challenge for some nodes in deep space, as their hardware and energy resources may be limited. In this case having a

selection of different key exchange protocols available to allow application of different security policies, seems to be a necessary functional requirement.

4 Conclusion and Areas of Ongoing and Future Research

In this paper we have attempted to provide some directions and propose an approach for addressing the challenging problem of authenticated key exchange in space DTN environments. As the DTN is relatively new, the current state of the art is mainly limited to the “language” the security nodes should speak upon which the security services would be built. Limited work has been done in the area of key management and more specifically in key exchange.

We have demonstrated how to adopt a communication efficient authenticated key exchange protocol and make it suitable for a DTN environment. We have confirmed that the recently published Bundle Security Specification Protocol is appropriately designed to accommodate a plethora of key exchange protocols. We also realised that in an environment with relatively sparse resources the security decisions depend on the opportunities of the exploiting these resources and this needs to be reflected in the security policy. On this end, we proposed an encryption decision making workflow based on a popular communications scenario.

In terms of future research activities, the decision making policies need to be evaluated for correctness. This can be done by formal model checking methods, as these policies do not exhibit a large number of states and as such state space explosion will not be an issue.

Another area of ongoing research is the experimental integration of the proposed approach with an existing testbed in order to empirically evaluate this solution and explore other scenarios and protocols. We maintain a shared testbed with other research institutions and this activity is scheduled for the near future.

Acknowledgement. The research leading to these results has received funding from the European Community’s Seventh Framework Programme (FP7/2007-2013_FP7-SPACE-2010-1, SP1 Cooperation, Collaborative project) under grant agreement no. 263330 (project title: SPACE-DATA ROUTERS for Exploiting Space DATA). This paper reflects only the authors views and the Union is not liable for any use the may be made of the information contained therein.

References

1. Ankney, R., Johnson, D., Matyas, M.: The Unified Model, contribution to X9F1 2. ANSI X9.42, Agreement (1995)
2. Asokan, N., Kostianen, K., Ginzboorg, P., Ott, J., Luo, C.: Towards securing disruption-tolerant networking, Technical Report NRC-TR-2007-007 (2007)
3. Bhutta, N., Ansa, G., Johnson, E., Ahmad, N., Alsiyabi, M., Cruickshank, H.: Security analysis for Delay/Disruption Tolerant satellite and sensor networks. In: International Workshop on Satellite and Space Communications (IWSSC), pp. 358–359 (2009)

4. Blake-Wilson, S., Menezes, A.: Authenticated Diffie-Hellman Key Agreement Protocols. In: Tavares, S., Meijer, H. (eds.) SAC 1998. LNCS, vol. 1556, pp. 339–361. Springer, Heidelberg (1999)
5. Chalkias, K., Halkidis, S.T., Hristu-Varsakelis, D., Stephanides, G., Alexiadis, A.: A Provably Secure One-Pass Two-Party Key Establishment Protocol. In: Pei, D., Yung, M., Lin, D., Wu, C. (eds.) Inscrypt 2007. LNCS, vol. 4990, pp. 108–122. Springer, Heidelberg (2008)
6. Chalkias, K., Baldimtsi, F., Hristu-Varsakelis, D., Stephanides, G.: Two Types of Key-Compromise Impersonation Attacks against One-Pass Key Establishment Protocols. In: E-Business and Telecommunication Networks (book chapter). Springer (2008)
7. Chalkias, K., Baldimtsi, F., Hristu-Varsakelis, D., Halkidis, S.T., Stephanides, G.: Attacks on the AKACP Protocol. IACR Cryptology Eprint Archive (2010)
8. Elkamchouchi, H., Eldefrawy, M.: A New Approach for Key Controlled Agreement. In: 24th National Radio Science Conference, NRSC 2007, pp. 1–7. Ain Shams University, Egypt (2007)
9. Elkamchouchi, H., Eldefrawy, M.: An Efficient and Confirmed Protocol for Authentication Key Agreement. In: 25th National Radio Science Conference, NRSC 2008, pp. 1–8. Tanta University, Egypt (2008)
10. Farrell, S.: DTN Key Management Requirements, work in progress as an internet-draft (2007), <http://tools.ietf.org/html/draft-farrell-dtnrg-km-00>
11. Farrell, A., Symington, S.F., Weiss, H., Lovell, P.: Delay-Tolerant Networking Security Overview, internet-draft (2009), <http://tools.ietf.org/html/draft-irtf-dtnrg-sec-overview-06>
12. Horster, P., Michels, M., Petersen, H.: Authenticated encryption schemes with low communication costs. IEEE Electronics Letters 30(15), 1212–1213 (1994)
13. International standard: Information technology - Security techniques - Key management - Part3: Mechanisms using asymmetric techniques. 2 edn. (2008)
14. Jia, Z., Lin, X., Tan, S.-H., Li, L., Yang, Y.: Public key distribution scheme for delay tolerant networks on two-channel cryptography. Journal of Network and Computer Applications (2011)
15. Kate, A., Zaverucha, G., Hengartner, U.: Anonymity and Security in Delay Tolerant Networks. In: 3rd International Conference on Security and Privacy in Communications Networks and the Workshops, Secure Communication, pp. 504–513 (2007)
16. Krawczyk, H.: HMQV: A High-Performance Secure Diffie-Hellman Protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546–566. Springer, Heidelberg (2005)
17. Lauter, K., Mityagin, A.: Security Analysis of KEA Authenticated Key Exchange Protocol. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 378–394. Springer, Heidelberg (2006)
18. Law, L., Menezes, A., Qu, M., Solinas, J., Vanstone, S.: An efficient protocol for authenticated key agreement. Technical report CORR 98-05, University of Waterloo (1998)
19. Mohammad, Z., Chen, Y.-C., Hsu, C.-L., Lo, C.-C.: Cryptanalysis and Enhancement of Two-pass Authenticated Key Agreement with Key Confirmation Protocols. IETE Technical Review (Institution of Electronics and Telecommunication Engineers, India) 27(3), 252–265 (2010)
20. National Security Agency: SKIPJACK and KEA algorithm specification, Version 2.0 (1998)

21. Nyberg, K.: On one-pass authenticated key establishment schemes. In: Workshop on Selected Areas in Cryptography (SAC 1995), pp. 2–8 (1995)
22. Pour, A.N.: Number Theory and Related Algorithms in Cryptography, Master's thesis, Japan Advanced Institute of Science and Technology, pp. 37–43 (2002)
23. Song, B., Kim, K.: Comparison of Existing Key Establishment Protocols. In: Information Security and Cryptography, pp. 1–13 (2000)
24. Symington, S., Farrell, S., Weiss, H., Lovell, P.: Bundle Security Protocol Specification. Request for Comments, RFC 6257, <http://datatracker.ietf.org/doc/rfc6257>
25. Van Besien, W.: Dynamic, Non-Interactive Key Management for Bundle Protocol. In: 5th ACM Workshop on Challenged Networks (CHANTS 2010), Illinois (2010)
26. Wood, L., Eddy, W.M., Holiday, P.: A bundle of problems. In: Aerospace Conference, pp. 1–14 (2009)

OFELIA – A Secure Mobile Attribute Aggregation Infrastructure for User-Centric Identity Management

Alexandre B. Augusto and Manuel Eduardo Correia

Center for Research in Advanced Computing Systems (CRACS-INESC LA),
Department of Computer Science, Faculty of Science, University of Porto, Portugal
{aaugusto,mcc}@dcc.fc.up.pt

Abstract. Personal mobile devices with real practical computational power and Internet connectivity are currently widespread throughout all levels of society. This is so much so that the most popular of these devices, the smart phone, in all its varied ubiquitous manifestations is nowadays the de facto personal mobile computing platform, be it for civil or even military applications. In parallel with these developments, Internet application providers like Google and Facebook are developing and deploying an ever increasing set of personal services that are being aggregated and structured over personal user accounts were an ever increasing set of personal private sensitive attributes is being *massively aggregated*. In this paper we describe OFELIA (Open Federated Environment for Leveraging of Identity and Authorization), a framework for user centric identity management that provides an identity/authorization versatile infrastructure that does not depend upon the massive aggregation of users identity attributes to offer a versatile set of identity services. In OFELIA personal attributes are distributed among and protected by several otherwise unrelated AAs (Attribute Authorities). Only the user mobile device knows how to aggregate these scattered AAs identity attributes back into some useful identifiable entity identity. Moreover by recurring to an IdB (Identity Broker), acting as a privacy enhancing blind caching-proxy, in OFELIA the identity attributes location in the Internet is hidden from the RP/SP (Relying Party, Service Provider) that wants to have temporary access to the users personal data. The mobile device thus becomes the means by which the user can asynchronously exercise discretionary access control over their most sensitive dynamic identity attributes in a simple but highly transparent way.

Keywords: Secure Digital Identity management, User centricity, Mobile Identity Wallet, XMPP, OpenID Connect, Attribute aggregation, Access control.

1 Introduction

Due to the massive organic growth of the Internet, with its unaccountable number of unrelated services, users personal data is currently completely scattered

all over the network. This is the direct result of the current need to create different user accounts (identity personas [11]) for the numerous Internet services that are being run by different operators. However this fragmentation of identity data can in some way be seen as a positive feature, because this means that no single system is capable of completely identifying a person identity attributes, in other words, user identity data on the Internet is naturally decentralized and this is a very useful tendency we should explore to improve upon the users privacy.

The interest on users digital identity has been increasing dramatically over the recent years due to its highly strategic commercial value for the market [21]. Internet application providers, companies like Google, Facebook and even Microsoft, are currently under a fierce competition over the hearts and minds of users for their personal data. Their main purpose is to create enormous monopolized centralized databases of user identity attributes as they allow them to produce highly accurate user profiles that they can then monetize very efficiently for marketing purposes. These global companies harvest and aggregate personal data in such a large scale that, lest it is put under some kind of control, it will very soon represent a major global threat to personal security and privacy the like of which the world has never seen.

Moreover interoperable Internet applications flourish in the presence of standardized and simple to use Identity, Authentication and Authorization services. This is manifested on the push Google, Facebook and other major players have been given lately to open identity and authorization protocols like OpenID [19] and OAuth [14]. These are employed as standardized mechanisms to build single sign-on systems and attribute sharing based on valet keys [13], which are nowadays essential to keep and follow the user navigating within the same service provider set of managed services. At the moment OpenID Connect [18] is under development, as a single solution for aggregating both Identity and Authorization into one single open standard.

However to share or give access to highly sensitive data [22] like bank accounts, electronic health record or the current geographic position to monopolized identity providers, nowadays constitutes a highly risky proposition. Once a user shares this kind of data he immediately loses control over it, not to mention that if the IdP suffers an attack, millions of highly detailed personal attributes can be immediately compromised. Personal data is also subject to change, and depending upon its nature it can quickly become stale. With a centralized and "distant" identity provider it can become quite difficult to manage the staleness of massive amounts of personal data. For these reasons we thus strongly believe that user data should be kept and managed as much as possible close to its origin by its owner, its *Authoritative Source*, and should only be made accessible with its owner explicit authorization. The disclosed data should also only be readable by the original requester, therefore a monopolized centralized intermediary Identity Provider (IdP) should not be trusted with the requested attributes. These characteristics of identity data lead us to propose the

development of a fully decentralized privacy and user oriented model for identity management. The necessity for user digital data aggregation from many different authoritative sources in a secure and user centric way [17] is our major motivation behind project OFELIA¹ (Open Federated Environment for the Leveraging of Identity and Authorisation).

In this project we are developing an identity infrastructure that is tackling digital identity related problems like: how much information service providers (RP/SP) and IdPs should have access to ? who should aggregate the user data? what authorizations and proofs are needed to request personal data? how a RP/SP can be sure that who provided the data is really its Authoritative Source? These are all problems we need to solve if one wants to provide a highly distributed user identity management based on the aggregation of scattered attribute authorities.

To a better comprehension of these problems and the possible solutions, we proceeded with a research about the already existing identity attribute aggregation models:

- *Identity relay* [16]: The SP trust in a single master federated IdP that is responsible to request all attributes to the SP, these attributes are returned directly to SP, in other words SP is responsible to aggregate the attributes. This model is like Identity proxying but with a reduced level of trust on master IdP.
- *Identity proxying or chaining* [9]: The service provider(SP) fully trust in a single master federated IdP that is responsible to request and aggregate all requested attributes before send it back to SP in other words the master IdP can request attributes from others IdPs that are part of its federation. This model have no control about how many IdPs will be requested to fulfill a request and was typified by myVOCS.
- *RP/SP mediated attribute aggregation* [2]: Based on SP-IdP federated model, SP redirect the user to each IdP, obliging the users a high level of interaction and is responsible to attribute aggregation.
- *Identity Federation model* [3]: Based on federated network, after user authentication a secret is generated and shared with each requested federated IdP by a user agent, the first contacted IdP provides SP the details of others IdP allowing the SP request the needed attributes. IdPs can create wrong assumptions about the attributes that others IdP issues.
- *Linking Service* [5]: In this model only the user knows about all his IdPs, a service called linking service is responsible to hold minimal information to allow SPs to obtain their queries from others IdPs. After a user authenticates, the IdP offers the possibility of attribute aggregation and if the user accept it the information to access the linking service is shared with SP, the aggregation of attributes can be done by the liking service itself or at SP.

¹ OFELIA-PTDC/EIA-EIA/104328/2008

- *Client mediated assertion* [16]: Based on an intelligent user agent that guide the user to the different IdPs, obliging the users a high level of interaction, the user agent is responsible for the attribute aggregation and the delivery to SP.

It is also currently widely accepted [4] that user centricity, in particular user authorization, is not only advisable but essential for attribute assertions to be considered reliable. Digital identity attributes should also be digitally signed by their respective Authoritative Sources and the end-user interactions related to identity management should also be kept as simple and sporadic as possible. Unfortunately there is no agreed upon final conclusion on the literature about the place where attributes should be kept aggregated. This is manifested by the high diversity of attribute aggregation models existing on the literature [4].

In this paper we propose a dynamic user centric identity attributes aggregation model with persistent user managed authorization mechanisms where the user smart-phone acts as the authorization and attribute management node in other words we intend to deploy user smart-phones as secure digital wallets with a full list of the user associated authoritative sources of his identity attributes. We strongly believe that it is essential to set the user as the unique authorization and revocation agent and the best way to materialize our vision is by employing smart phones because these devices are nowadays ubiquitous, have more then adequate processing CPU power to run modern operating systems, are being deployed with full Internet access, are accompanied by fully matured development systems and constantly follow their owners everywhere as the de facto personal mobile device.

The rest of the paper is organized as follows. In Section 2 we review the system architecture, describing each node, their functionality and how data flows between the different actors involved. In Section 4 we describe a representative usage case scenario that helps us to better understand the interplay of the different actors involved in OFELIA attributes authorizations and exchanges, its applicability and the its main advantages. In Section 5 we describe what has already been implemented, present some preliminary conclusions for the work we have developed thus far for OFELIA and delineate our plans for the immediate future.

2 Architecture

In this section we describe the main components of the OFELIA architecture and discuss the main reasons behind some of the options and compromises we had to make to realize our vision. We also take some time to describe the conceptual model for attribute aggregation and its most relevant aspects like the protocols and services we have employed to devise the OFELIA secure communication infrastructure. We are currently developing and testing four different components, one API and library components for the RP/SP, another for Attribute Authorities, an implementation for an IdB [12] and an android

OFELIA app, implementing the Digital identity Wallet authorization broker to aggregate AAs and authorize, manage and revoke access to their identity attributes. Figure 1 illustrates the relationship between the main OFELIA components and the type of communication that can occur between them in a simplified way.

In what follows we provide a more detailed description of the functional role played in OFELIA by each one of these OFELIA architectural components.

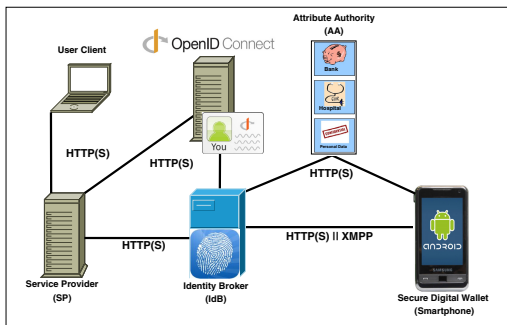


Fig. 1. OFELIA nodes relationship

2.1 OpenID Connect Identity Services

The OpenID Connect protocol is a simple identity layer built on top of the OAuth 2.0 protocol. It allows SPs to verify the identity of their end users by taking advantage of the authentication services provided by an associated OAuth service. This protocol is also capable of providing basic profile information about the end user by providing the web application developer with an identity/authentication API based on RESTful web services [18]. OpenID Connect allows users to sign into multiple different web applications with a single account, in Single Sign On (SSO) mode and at the same time control which of the user identity attributes can be shared with each one of these web applications.

In OFELIA we employ OpenID Connect as an authenticator and the provider of the OFELIA bootstrapping information required by the RP/SP to enroll into OFELIA. The essential information needed to bootstrap a RP/SP into OFELIA, for a particular user, consists of two identity attributes provided by OpenID Connect to the RP/SP, the Identity Broker Internet domain name and the user public key.

2.2 The RP/SP (Relying Party/Service Provider)

In OFELIA, a RP/SP is a web application that requires users attributes that are being managed and protected by the OFELIA identity infrastructure. We are currently developing an API and OFELIA software library components to allow for a much more simple integration of current existing web application into the OFELIA infrastructure.

The OFELIA software for the RP/SP provides functionalities for X509 certificate management, supports OpenID Connect authentication and is capable to asynchronously request and store OFELIA user attributes. To this effect it must also be capable of managing OFELIA's authorization tokens, user identifiers, expiration dates, decryption of attribute values and recognize AA identifiers

according to the OFELIA's specifications. It must also allow for the caching of conditional authorizations tokens provided by OFELIA, which must then be presented each time the RP/SP wants to renew an access to the associated OFELIA identity attribute.

2.3 Attribute Authorities

In OFELIA, *Attribute Authorities(AAs)* are network entities responsible for the security and management of the data owner identity attributes. The user mobile phone needs to be enrolled into each one of the users AA, in order to determine which personal attributes are being held and maintained by each one of these AAs. The mobile phone can then act as an authorization broker where access for each one of these attributes can then be announced and further negotiated with each one of the participating RP/SP.

The OFELIA framework for AAs provides appropriate security mechanisms for authentication and authorization to ensure the appropriate level of access control necessary to protect these assets from unauthorized access, and provides the SPs with the means to negotiate with the IdBs and mobile phone the authorization needed to be able to access the resources being protected by each one of the user registered AA. This type of framework allows a simple and fast integration of already existing AAs into OFELIA's infrastructure.

Each participating AA must be in the possession of a public key pair whose legitimacy can be attested by a valid OFELIA's PKI X509 certificate containing the AA's OFELIA identity. The also AA stores, for each one the OFELIA user identity attributes, their respective currently valid authorization tokens and for each one these tokens their expiration dates and the related RP/SP and IdB involved in that particular user authorization.

2.4 The Identity Broker

In OFELIA, the Identity Broker(IdB) acts like a privacy enhancing blind caching-proxy for identity attributes that hides from the SP the real network location of the AA responsible for that data. We need to keep in mind the importance of catering for the situations where the RP/SP cannot be fully trusted and it is therefore important to hide the AA real network location behind a trusted IdB. Moreover for privacy and security reasons, in OFELIA the IdB does not know the content of personal attributes it is proxying because they are encrypted with the asking RP/SP public key by the custodian AA before they are delivered to the IdB to be sent to the requesting RP/SP.

In OFELIA we aim for a trusting equilibrium where the RP/SP does not need to know the location of the AAs and the IdB does not need to know the nature and value of the personal attributes he his proxying for the RP/SPs. For authentication purposes and to prevent men in the middle attacks it is mandatory for the IdB to be in the possession of a public key pair whose legitimacy can be attested by a valid OFELIA's PKI X509 certificate containing the AA's OFELIA identity.

2.5 The Smart-Phone as a Secure Digital Wallet

In OFELIA we are employing android smart phones as highly decentralized personal access authorization management devices for identity management, empowering the user with the creation a management of the access control policies the finds most adequate for his own personal data. This means that user is no longer obliged to comply with the abusive identity management policies implicitly normally in place at major sites where the user is made to share or give full control of his data to network entities he does not fully know or does not fully trust, as happens with the majority of current Internet applications. OFELIA also brings some advantages in security due to the full "hidden" decentralization it imposes on the storage of identity attributes.

All mechanisms related to authorization token creation, token revocation, attribute access authorization and the enrollment into AAs and IdBs is conducted by OFELIA application installed on the smart-phone. More details about tokens authorization and AA and IdB enrollment process are discussed on [3.1](#) and [3.2](#).

The smart phone OFELIA application is the critical component of the users digital identity and should thus be always reachable over the Internet. Unfortunately this is not always possible. Network aware smart phone application are highly demanding on terms of phone battery usage and therefore cannot be always left running. In OFELIA we circumvent this problem by having the IdB to send a SMS message requesting the mobile phone to reconnect to OFELIA, every time the IdB needs to communicate with the phone but cannot reach it via the usual OFELIA channels, namely XMPP messaging. The phone has one SMS handler service installed on the phone that on receiving OFELIA reconnect SMS messages, launches the appropriate application thus reconnecting the phone back into OFELIA. After a certain period of inactivity the OFELIA application terminate to save on phone battery.

2.6 The XMPP Messaging Protocol

The XMPP messaging protocol is an open technology for real-time communication that uses the eXtensible Markup Language (XML) as a base format for exchanging information encapsulated into small pieces of XML [20](#). XMPP provides a complete standard set of services like authentication, asynchronous one-to-one messaging and other very useful messaging oriented services [7](#).

Arguably, in the cellular mobile world an implicit direct Internet communication with a personal device is generally not possible due to the shortage of public IPs addresses faced by Internet service providers. In the near future, IPv6 is supposed to have solved this problem, however we believe that the mobile Telecommunications operators (Mobicomms) will not allow for directly addressable mobile devices from the Internet due to their less flexible business plans and business culture that regards the mobile devices, smart phones in particular, as a strict consumer device, not as a provider of services. Towards this end, XMPP

messaging is proving to be an almost ideal communication infrastructure for OFELIA to circumvent these communication restrictions because of its ability to efficiently operate over HTTP by the means of the BOSH(Bidirectional-streams Over Synchronous HTTP) [15] protocol where two non directly addressable devices, located on private closed intranets and with minimal Internet access, can locate each other over the Internet and then freely exchange messages between themselves in a reliable and safe way.

2.7 OFELIA Secure Access Authorization Tokens

An OFELIA authorization token can be seen as something that the RP/SP has, that gives temporary access to some identity attribute and can be easily validated by an AA. The tokens are also very hard to falsify and take the form of a small base64 encoded XML excerpt, containing elements for a large pseudo-random number [6], and a simple statement describing the authorization validity restrictions applying to this particular authorization. This statement can express for example temporal restrictions. This XML is then digitally signed by the users phone OFELIA private key and the resulting XML document is then encoded into a base64 string which constitutes the OFELIA authorization token. The token is then installed by the phone into the AA responsible for the requested identity attribute. A copy is also sent by the phone to the IdB that then forwards it to the requesting RP/SP. These authorization tokens provide a more flexible security mechanism for smart-phone users to provide RP/SPs with a restricted more controlled access to their AAs identity attributes without having to share more permanent and hard to manage credentials.

2.8 The OFELIA TRUST Infrastructure

One of the critical components of OFELIA is the management of trust among the participating components. This role is played by a PKI infrastructure responsible for the management of the security certificates that constitute the core of the privacy, trust and authentication infrastructure we need to put in place to secure the OFELIA architecture.

To establish a stronger and therefore more trustworthy identity/authentication between the different OFELIA actors, RP/SP, AAs, IdBs and the personal smart phone, we rely on the existence of a standard compliant PKI that signs the X509 certificates we employ as id wallets for each one the OFELIA participating actors. Due to to the critical role played by the personal smart phone, that acts as the core authorization broker, in OFELIA we are taking advantage [8] of micro-SD mobile security cards [10] to better protect the mobile private keys associated with X509 certificates issued by the OFELIA PKI to the mobile devices. The smart phone OFELIA X509 certificate can also be doubly signed by the government issued electronic identity (eID) smart-card of the user's country to further ascertain his real civil identity.

3 Entities Enrollment and Communication Schemes

In OFELIA the mobile device must be enrolled into the each one of the aggregated AAs and into the IdB. The mobile device enrollment into each one of the user Attribute Authorities constitutes the main attribute aggregation mechanism employed by OFELIA. The mobile phone must also enroll into the IdB so it can then be announce and manage the list of attributes names and respective types that can then be made available to the requesting RP/SPs. These are maintained within the AAs aggregation sets that are being controlled by the user mobile device. This list is dynamic and must thus be updated each time the mobile phone is enrolled or unrolled from an AA, thus increasing or decreasing the number of attributes announced by the IdB for that particular digital identity that is being managed by the user mobile device

3.1 Attribute Authority Enrollment

The enrolment process should be as painless and automatic as possible for the users, and for that we can rely on the services provided by the OFELIA's AA framework infrastructure (already mentioned on subsection 2.3) and QR-codes [11].

In the OFELIA's AA framework, after being authenticated, the user is provided with the option to link his digital wallet (mobile phone). The set of parameters that must be provided to the mobile phone to achieve this linkage can be transmitted by the means of a web session screen QR-code that provides the mobile device with the necessary URL locations, the AA X509 certificate and the access token needed to officialise this connection in a secure way. To link his mobile device (digital wallet) into the AA, the user employs his OFELIA mobile application to scans the AA web session QR-code that provides the application with all the parameters it needs to conclude the enrolment process. Figure 2 illustrates this process and provides for a more detailed explanation of the AA enrolment process.

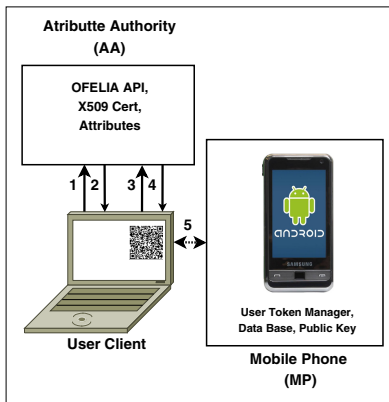


Fig. 2. AA enrollment flow

1. User requests authorization by sending the necessary credential.
2. AA Answers by granting authorization if credential are accepted.
3. User request a full access token.
4. AA answers with an access token and the AA information encrypted with the users public key, aggregated and encoded as a QR code.
5. The User uses a QR code scanner to register the AA on his digital wallet (mobile phone).

3.2 Identity Broker Enrollment

In order to establish an OFELIA authorization flow the user must have his mobile device (aggregating digital wallet) enrolled into the OFELIA IdB.

A similar process happens with the enrollment into the IdB as has already been described for the enrollment into an AA. The user logs in/authenticates into the IdB by OpenID Connect which provides the IdB with the XMPP identity and the public key of the mobile device. The user is then presented at his PC screen with a QR-code that can then be scanned by the mobile device and contains the information the smart phone needs to automatically enroll into the IdB, again via the appropriate invocation of enrollment web-services made available by the IdB, and be thus semi-automatically associated with the user OpenID identity. The IdB also provides the user with a web interface where he can list a history of the RP/SP attribute requests that have been made for that particular OFELIA identity, the enrollment process ends after the mobile OFELIA application sends an attribute name list of all attributes linked on mobile. This list is classified into generic groups like: Banks, Hospitals, Sports and etc. This enrollment process is illustrated on figure 3.

After the enrollment process has been completed, the user interacts with the mobile OFELIA application to decide upon and determine the restrictions that should be associated with each access requests being made by RP/SP web applications. He can also use the OFELIA application to revoke previously given and still valid authorizations.

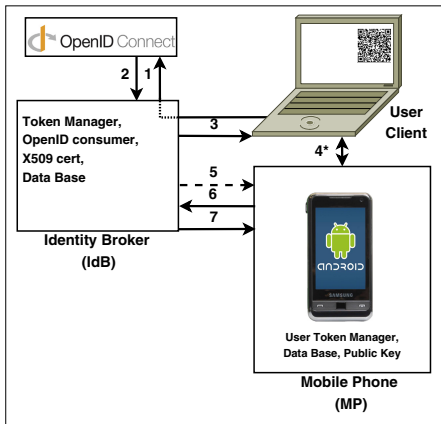


Fig. 3. IdB enrollment flow

1. User authenticates at IdB via Openid Connect allowing IdB to request users jabber Id and public key.
2. Openid Connect answers to IdB with the requested data.
3. IdB sends back to user browser a Q-R code holding IdB information: Certificate, users identification and IdB addresses (jabber and web).
4. User using a Q-R code reader pre-register the IdB on his digital wallet. (mobile phone)
5. IdB sends via XMPP a signed challenge encrypted with user mobile public key.
6. Mobile phone answers the challenge to IdB and send the list of attribute names holden by itself.
7. IdB confirms the registration.

3.3 Service Provider Enrollment

Every time the user decides to use a new SP an enrollment process is triggered in order to allow data exchange. This process is a bit longer than the others since all nodes have to act.

The user logs in/authenticates into the SP by his OpenID Connect account which provides the IdB link, then the user is redirected to the IdB with a generic request list from the SP, now the user has to interact and decides the attributes he will give access based on the SPs generic list. After authorization is given, the IdB sends via XMPP a request to confirm the authorization to the mobile OFELIA APP that must be confirmed by the user mobile thus triggering an authorization token creation phase. Now the mobile OFELIA APP has to create signed access tokens for the respective requested AAs, sending the tokens to them and to the SP by using the IdB, in other words encrypting the tokens with SP certificate and requesting IdB to delivery it. This scenario is exemplified on figure 4.

Now the SP can request attributes from IdB until authorization given by the user is valid.

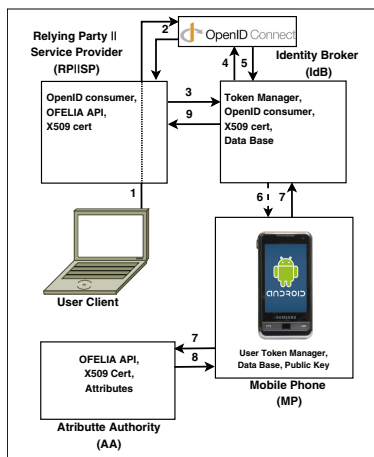


Fig. 4. RP/SP enrollment flow

1. User authenticates itself at RP via Openid Connect allowing RP to request users public key and IdB address.
2. Openid Connect answers to RP with the requested data.
3. RP requests a registration to the IdB providing his certificate, OpenID request link and a details of the service with a list of requested data cyphered with user public key.
4. IdB tries the OpenID request link.
5. If the answer is a reply attack tentative the IdB will preregister the RP generating a identifier token.
6. IdB sends via XMPP to the MP a signed request message with the encrypted data request plus RP details(identifier token, certificate, details of service and address).
7. If the user authorizes, an access token is generated and sent to IdB encrypted with RP public key and to AA with RP details
8. IdB validate RP registration and send to RP the encrypted access token.

4 Usage Case Scenario

For a credible illustrative OFELIA aggregation scenario imagine a chain retailer supermarket acting as a SP and for example a credit card and gas company acting as AAs. Now lets assume the user is online shopping at the chain retailer and upon completion of his purchase, if he can prove that he has a specific credit card and is a regular customer of a certain gas company, the chain retailer gives him an immediate special discount on car accessories.

At the moment of purchase after with user already authenticated via OpenID Connect the supermarket, acting as a SP will request the IdB for proof of credit card membership and gas service for that already logged particular user. This triggers an authorization request made by the IdB that is displayed at the user mobile phone, to authorize the relevant AAs to disclose this information. On the user discretion, he then uses his mobile phone to authorize both AAs to emit a certification (signed by the AAs public certificates). These authorizations take the form of digitally signed authorization tokens that are registered on the respective AAs and delivered to the IdB but encrypted to the SP with an entropy salt, that then sends them to the asking chain retailer SP.

The SP, now in possession of these digitally signed tokens, can then present them to the IdB encrypted to the AA plus the salt each time he wants to get evidence the user is still a valid customer of the credit card and gas company. These tokens together with the consultation requests are then signed and relayed by the IdB into the appropriate AAs, which upon analyzing the validity of the accompanying authorization tokens deliver the requesting information back plus a new entropy salt to the IdB, digitally signed by the AAs and encrypted to the SP. The encryption step is important because for privacy and security reasons the IdB should not know the value of the identity attributes, otherwise the entity responsible for the IdB would be in a position of doing massive data aggregation with their users data, and that aggregation by itself would become a much more prized target for attacks. This constitutes two of the main reasons for OFELIA to have been developed in the first place, i.e. *to provide an identity/authorization versatile infrastructure that does not depend upon the massive aggregation of users identity attributes.*

Finally the IdB relays the requested encrypted information to the SP that can verify its integrity and validity by decrypting the attributes values and verifying the validity of its digital signature letting the supermarket apply the special discount on car accessories.

5 Conclusions

Nowadays we sit at a crossroads where there is a real need for users to gain back some level control about their personal data and be given the means to only disclose their most sensitive identity attributes when they need to use a network service that really requires access to this type of sensitive data. This should also only happen for a limited period of time and be kept under strict revocation control by the data legitimate owner. OFELIA infrastructure is thus an user centric empowering infrastructure where it is possible to securely dynamically manage the aggregation of identity attributes from different Authorization Authorities into a single user centric digital identity whose authorizations can be managed in a novel versatile way involving for example temporal constraints by the arbitrage of the user mobile phone.

OFELIA also possesses innovative mechanisms to protect users privacy by preventing the massive aggregation of users data into a single place. We have

taken special care to prevent the disclosure of attributes values at the IdB precisely to prevent the massive disclosure of user data lest the IdB be compromised. In OFELIA if an attacker compromises the IdB he will not have disclosed the user attributes values that should therefore continue to remain safe in a privacy aware away.

We are currently extending OFELIA with mobile phone to mobile phone communication mechanisms parametrized by QR-codes to cater for side channel authorization requests in the case where some OFELIA user, enrolled in a SP and acting as some predefined role wants to ask to some other user, permission to access some of his OFELIA managed personal attributes.

Acknowledgments. This work is funded by the ERDF through the Programme COMPETE and by the Portuguese Government through FCT - Foundation for Science and Technology, project OFELIA ref. PTDC/EIA-EIA/104328/2008 and is being conducted with the institutional support provided by DCC/FCUP and the facilities and research environment gracefully provided by the CRACS (Center for Research in Advanced Computing Systems) research unit, an INESC LA associate of the Faculty of Science, University of Porto.

References

1. Baden, R., Bender, A., Spring, N., Bhattacharjee, B., Starin, D.: Persona: an online social network with user-defined privacy. *SIGCOMM Comput. Commun. Rev.* 39, 135–146 (2009)
2. Cantor, S.: Shibboleth architecture, protocols and profiles (September 2005), <http://www.mediafire.com/?8bswqc4y47sqygw> (verified on January 13, 2012)
3. Chadwick, D.: Authorisation using attributes from multiple authorities. In: 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE 2006, pp. 326–331. IEEE (2006)
4. Chadwick, D., Inman, G., and Klingenstein, N. Authorisation using attributes from multiple authorities—a study of requirements. *European Institute for E-Learning (EIFEL)*, 366 (2007)
5. Chadwick, D.W., Inman, G., Klingenstein, N.: A conceptual model for attribute aggregation. *Future Generation Computer Systems* 26(7), 1043–1052 (2010)
6. Eastlake III, D., Crocker, S., Schiller, J.: Randomness recommendations for security (2005), <https://ietf.org/rfc/rfc4086.txt> (verified on February 14, 2012)
7. Saint-Andre, P. (ed.): Extensible messaging and presence protocol (xmpp):core. RFC 3920, IETF (July 2004)
8. For Android, P.S. Secure element evaluation kit for the android platform - the 'smartcard api' (2011), <http://tinyurl.com/seek4android> (verified on January 10, 2012)
9. Gemmill, J., Robinson, J.-P., Scavo, T., Bangalore, P.: Cross-domain authorization for federated virtual organizations using the myvocs collaboration environment. *Concurr. Comput.: Pract. Exper.* 21, 509–532 (2009)
10. GmbH, G.. D. S. F.S. Mobile security card ve 2.0 (2011), <http://tinyurl.com/mobseccard> (verified on January 10, 2012)

11. Huang, H.: Reversible data hiding with histogram-based difference expansion for qr code applications. *IEEE Transactions on Consumer Electronics* 57(2), 779–787 (2011)
12. Haaker, T., Smit, S., Vester, J., Shepherd, K., Ito, N., Guelbahar, M., Zoric, J.: Business models for networked media services. In: *Proceedings of the Seventh European Conference on European Interactive Television Conference, EuroITV 2009*, pp. 53–56. ACM, New York (2009)
13. Hammer-Lahav, E.: *Introducing oauth 2.0* (2010)
14. Hammer-Lahav, E.: *The oauth 1.0 protocol (rfc5849)* (April 2010), <http://tools.ietf.org/html/rfc5849> (verified on April 14, 2011)
15. Ian Paterson, P. S.-A.: *Xep-0206: Xmpp over bosh* (July 2010), <http://bit.ly/xep0206> (verified on April 14, 2011)
16. Inman, G., Chadwick, D.: A privacy preserving attribute aggregation model for federated identity managements systems. *Serbian Publication InfoReview joins UP-ENET, the Network of CEPIS Societies Journals and Magazines* 21 (2010)
17. Jsang, A., Pope, S.: User-centric identity management. In: *Proceedings of AusCERT 2005, Brisbane, Australia* (May 2005)
18. Sakimura, N., et al.: *Openid connect standard 1.0*, <http://tinyurl.com/openidc> (verified on January 13, 2012)
19. Recordon, D., Reed, D.: *Openid 2.0: a platform for user-centric identity management*. In: *Proceedings of the Second ACM Workshop on Digital Identity Management, DIM 2006*, pp. 11–16. ACM, New York (2006)
20. Saint-André, P., Smith, K., Tronçon, R.: *XMPP: the definitive guide. Definitive Guide Series*. O'Reilly (2009)
21. Schwartz, P. M.: *Property, Privacy, and Personal Data*. SSRN eLibrary
22. Song, D., Bruza, P.: Towards context sensitive information inference. *Journal of the American Society for Information Science and Technology*, IETF 54, 321–334 (2003)

Smart OpenID: A Smart Card Based OpenID Protocol

Andreas Leicher¹, Andreas U. Schmidt¹, and Yogendra Shah²

¹ Novalyst IT AG,
Robert-Bosch-Str. 38, 61184 Karben, Germany
{andreas.leicher, andreas.schmidt}@novalyst.de
<http://www.novalyst.de>

² InterDigital Communications, LLC.,
781 Third Avenue, King of Prussia, Pennsylvania, 19406 USA
yogendra.shah@interdigital.com
<http://www.interdigital.com>

Abstract. OpenID is a lightweight, easy to implement and deploy approach to Single Sign-On (SSO) and Identity Management (IdM), and has great potential for large scale user adoption especially for mobile applications. At the same time, Mobile Network Operators are increasingly interested in leveraging their existing infrastructure and assets for SSO and IdM. In this paper, we present the concept of Smart OpenID, an enhancement to OpenID which moves part of the OpenID authentication server functionality to the smart card of the user's device. This seamless, OpenID-conformant protocol allows for scaling security properties, and generally improves the security of OpenID by avoiding the need to send user credentials over the Internet and thus avoid phishing attacks. We also describe our implementation of the Smart OpenID protocol based on an Android phone, which interacts with OpenID-enabled web services.

Keywords: OpenID, Identity Management, Single Sign-On, Authentication, Smart Cards, GBA.

1 Introduction

One of the challenges in the growing use of online services is the management of digital user identities [1]. The issues arising from poorly implemented identity management (IdM) include identity theft, phishing, fraud and lack of privacy. Most services implement proprietary IdM systems, where a password based mechanism is the most widely used method for user authentication. Different solutions, such as Liberty Alliance [2], CardSpace [3,4], Higgins [5] have been proposed to address the issues arising from the extensive use of username/password authentication. However, most of these protocols and architectures have not seen a high adoption rate by end users and services in the consumer market.

In addition, users are becoming more concerned about their privacy and are less willing to provide personal identity information. This results in the requirement that while providing a convenient and comfortable access to services, the

IdM system has to provide security and privacy at the same time. The current distribution of user personal data, leads to different problems such as inconsistency of data, loss of control as well as multiple authentication and sign-on methods for the services the user wants to access. In our contribution we discuss a novel approach which combines the security of smart cards, the authentication systems of mobile network operators and an open IdM protocol, namely OpenID, in order to address these issues.

1.1 Role of Mobile Network Operators

With the evolution of new technology, new market entrants and changed customer expectations, Mobile Network Operators (MNOs) have to face a change in their traditional business model [6]. The potential for MNOs to exploit and leverage the large amount of user data they possess has been studied as part of the EU FP7 project PrimeLife [7]. The *IdM Enabler* concept has been developed to allow an economic valuation of IdM business models.

Building on their customer relationships and an existing infrastructure for authentication and communication, MNOs are a prime candidate to monetize this data by providing IdM services to third-party service providers [8,9]. They are in possession of *IdM data assets*, i.e., they can provide user attributes such as address, name, etc. and at the same time they have deployed the necessary *IdM Functional Capabilities*, i.e., the technical capabilities to provide authentication, authorization, accounting, attribute management and policy functions.

1.2 OpenID

OpenID [10] was developed with the goal to provide a lightweight, Single Sign-On experience to users across a wide variety of online services. The OpenID protocol allows users to sign on to different services with a single identifier, where the authentication itself is performed by the OpenID provider (OP). OpenID hence eliminates the need to create separate user accounts for the services, and thus tackles the main issues with classical password based authentications, namely phishing, the reuse of passwords on multiple service sites or the use of "Post-Its" to remember passwords [11,12].

OpenID uses identifiers in the form of an URL provided by an OP. The OP authenticates the user and validates the user's credentials on behalf of the services, which are referred to as Relying Parties (RP).

To sign in to a RP, the user provides his OpenID identifier to the RP, from which the RP can extract the necessary information about the location of the OP. The OP and RP then establish an association, i.e., they exchange a shared secret which is used to sign any future communication between the OP and the RP which can be identified by an association handle. Then, the RP redirects the user's browser to the OP login page. This redirection message, called OpenID authentication request [10], includes the user identifier and the association handle. The user authenticates at the OP, using the method provided by the OP, which most commonly is via username and password.

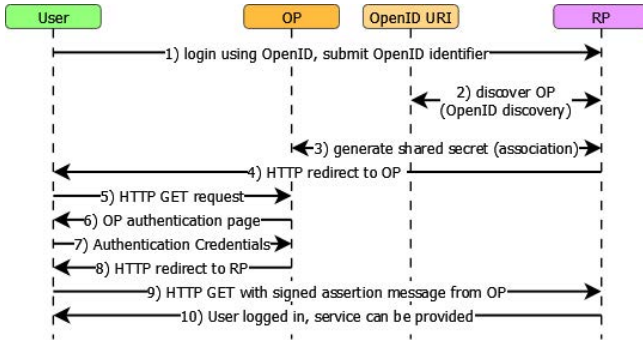


Fig. 1. Overview of the OpenID Protocol Flow

However, OpenID itself does not specify the authentication method, which has led to the integration of different authentication mechanisms, such as smart card based SSL certificates [13], TPM based authentication [14], or 3G network authentication mechanisms such as EAP-SIM [15] or GBA [16].

After successful user authentication, the OP redirects the user's browser back to the RP, including a signed assertion on whether the user authentication succeeded or not. The RP can then validate the authentication response by checking the signature on the signed assertion using the association secret. If the check is successful, the service can be provided to the user.

1.3 Smart Cards

Smart cards are portable and tamper-resistant computing devices which are able to securely store and process information. An attacker would need to be in possession of the smart card and also would need additional knowledge of the smart card hardware and software to probe for information. Smart cards are hence often used for secure data storage and authentication. The biggest market for smart cards is the mobile communication industry. Mobile phones have a Universal Integrated Circuit Card (UICC), which is a smart card that identifies the user. The UICC is able to perform authentication algorithms and provides cryptographic functions for encryption and decryption [17]. Using technology like Java Card, it is possible to write Java based code which is then loaded in the form of applets onto the smart card. Communication between a host application and an applet on the smart card uses Application Protocol Data Units (APDUs) in a command/response protocol [18]. Additional APIs, e.g. OpenMobileAPI [19], enable applications running on a mobile phone to access UICC functions.

2 Related Work

As OpenID receives a lot of attention from industry and research, different aspects of OpenID have been discussed in the literature. Most research work is

centered around the integration of different authentication mechanisms into the OpenID protocol. The intent of this section is to discuss the work which is closest to our proposal and highlight how we can improve on existing solutions. Additionally there has been research on weaknesses of the OpenID protocol, which we aim to mitigate by our solution.

2.1 OpenID 2.0 Security

A study on the security aspects of OpenID [20], highlights the danger of phishing as a major concern in OpenID. By tricking an user into giving away their OpenID authentication credentials, e.g. by setting up a fake OP, an attacker can get access to all OpenID enabled services in the name of the legitimate user. In order to do this, the attacker does not have to attack the OP directly but can set up a malicious RP which then redirects the user to the fake OP. As OPs act as Identity Providers for multiple users an attacker can use this technique to easily collect credentials for a large amount of users.

Another attack highlighted in [20] is the danger of Cross-Site-Request-Forgery (CSRF) attacks which silently logon the user to a service of the attacker's choice once the user has logged in into another RP and then allows the attacker to perform actions in the user's name. The CSRF attack relies on the fact that the OP and not the RP decides on the login security policy and that the OP does not show the login process to the user if a CSRF attack happens.

With the use of a smart card we introduce a multi-factor authentication, i.e., in order to use the smart card capabilities of our solution, the user is requested to provide an additional authentication factor such as a PIN code. As the attacker would need to be in possession of the smart card and the PIN code, phishing attacks become more difficult. Our solution is designed such that the user will be notified if a login process is taking place, preventing the issue of silent logon by CSRF attacks.

2.2 3GPP OpenID/GBA

The Generic Bootstrapping Architecture (GBA) [21] is intended to extend the security infrastructure of MNOs to applications and services on the internet. GBA provides a method for users to authenticate to services which implement a network application function (NAF) The NAF connects to the Bootstrapping Server function (BSF) of the MNO in order to bootstrap service authentication keys from the subscriber keys stored in the Home Subscriber Server (HSS).

GBA [22, p. 23-47] consists of two phases, where the first phase is a bootstrapping procedure and the second phase is an authentication phase with the NAF using the bootstrapped keys. In the case of UMTS networks the GBA keys are obtained by running the Authentication and Key Agreement (AKA) protocol [23] between the UE and the HSS with the BSF as intermediary. At the end of the bootstrapping, the BSF and UE obtain a session key K_s and a transaction identifier $B-TID$. The UE can now use this secret with the application specific NAF for authentication and securing the communication. A NAF specific key,

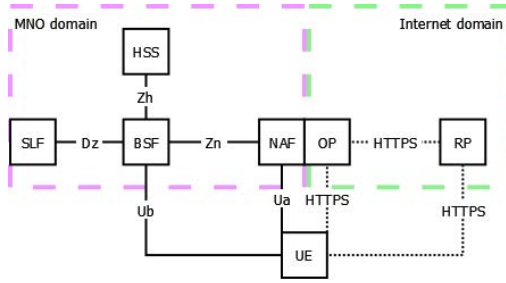


Fig. 2. Overview for an integrated OpenID/GBA Architecture according to [16]

Ks_NAF is derived from Ks in the UE and the NAF gets the same Ks_NAF from the BSF. This establishes a shared secret between the UE and the NAF. 3GPP has investigated the option for interoperability of GBA and OpenID [16,24,25], providing GBA based authentication at the OP.

The OpenID/GBA protocol follows the regular OpenID protocol as shown in figure 1 replacing steps 6 and 7 with GBA authentication. The solution relies on the availability of GBA functions in the network and on the device. While the network needs to implement the BSF and NAF functions, the device needs to implement a GBA module to communicate with the browser, the NAF and the UICC. The architecture of OpenID/GBA is shown in figure 2. A similar approach for integration of EAP-SIM with OpenID has been taken by [15].

2.3 SSL Certificate Based OpenID Authentication

In [13,26] a smart card based solution for OpenID authentication is presented, where a smart card stores a SSL certificate which may be used for authentication with the OP. The smart card carrying the key material and certificates is attached to a PC with a USB dongle. This solution requires an additional public key infrastructure and also requires the use of an additional USB key device. This approach is not as elegant as an approach which can leverage the already existing authentication infrastructure of MNOs.

2.4 Liberty Alliance Advanced Client

Liberty Alliance (LA) introduced in [27] a new entity, called advanced client. At the core of an advanced client is the trusted module (TM), which is an extension of the network IdP. Local applications and service providers can delegate authentication of the user to the TM instead of the network IdP. The TM is considered to run in a trusted and secure environment of the device. In [28] the deployment of a TM onto a UICC using 3GPP Over The Air (OTA) management operations [29,30] is discussed.

3 The Smart OpenID Protocol

In order to enable an efficient use of the existing and deployed security infrastructure of MNOs and at the same time provide seamless access to service providers, we introduce a new entity for our smart card based OpenID protocol, called local OP. The local OP, similar to the advanced client from LA [27], acts as a delegate of the network OP. The main role of the local OP is to authenticate the end user and issue the OpenID assertions to the end user's browser. The local OP therefore has an HTTP interface, which allows seamless communication with the device's browser without any modification to the browser itself. Using the capabilities provided by the UICC, the local OP can securely store secrets and perform crypto operations, e.g. creating signatures. The local OP may be implemented as a combination of a user level application, which provides the HTTP interface towards the browser and an applet on the UICC which handles all cryptographic operations and storage of keys. It is important to note that our proposed protocol does not impose any changes to the OpenID protocol, so existing RPs don't need to be modified. Due to restrictions from the mobile network, the local OP cannot be reached by RPs for the discovery and association steps of the OpenID protocol (steps 2, 3 in section 1.2, figure 1), i.e. communication to the local OP is restricted to the device only. Therefore, we introduce the OP support function (OPSF), operated by the MNO and reachable via public internet which facilitates these steps.

For the local OP to act as a delegate for the network OPSF, there needs to be a trust anchor. This trust is created by a shared secret S between the local OP running on the smart card and the OPSF. On the device, the secret is stored in the secure storage of the UICC and can only be used by the local OP. The shared secret can be established with different means, ranging from a preinstalled secret at time of personalisation of the UICC, by binding it to the network authentication or by using the MNO's OTA capabilities [29,30]. When the user logs on to a RP using his identifier, the RP performs discovery of the OPSF and requests to establish an association. Since the RP cannot easily communicate directly with the local OP on the mobile device, the OPSF is used in this step for the discovery and association process. The OPSF looks up the shared secret S for the identifier and creates a random association handle `asc_hdl`, which is an ASCII string of at most 255 characters. The OPSF then uses a key derivation function (KDF) with input S and `asc_hdl` to create an association secret S_a , which will be valid for this OpenID session. In the association response [10, sec. 8.2], the OP returns the `asc_hdl` and the S_a to the RP, where S_a is encrypted using a Diffie-Hellman secret established between the RP and OPSF. The RP then redirects the user to the OP. Instead of following the redirect to the network OP, the browser is actually redirected to the local OP, which can be done via a modification to the local DNS lookup or by a browser plugin. The local OP then first authenticates the user, requesting for example a password or PIN code. This authentication is done locally, in contrast to the regular OpenID protocol, where user authentication credentials are sent over the internet to the OP. Local authentication not only protects from phishing or man in the middle attacks

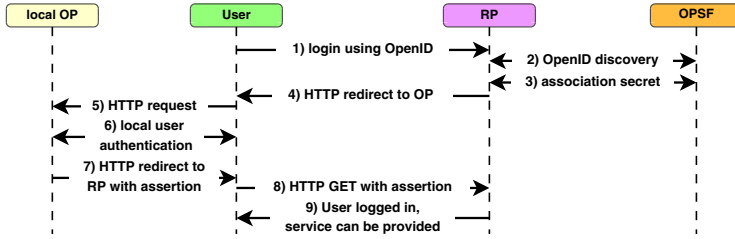


Fig. 3. Protocol Flow for the Smart OpenID protocol

on the internet, but also introduces a two-factor authentication and allows to employ advanced authentication mechanisms, such as biometrics. Since the local OP runs on the device, it can either use a regular webpage or use the user interface of the phone's operating system to provide a more seamless experience to the user. After the user has successfully authenticated towards the local OP, e.g. by entering a PIN code or password, the local OP checks if the identifier is in the list of allowed identifiers for this user.

If so, the local OP loads the shared secret S and together with the `asc_hdI` received from the redirected request, applies the same KDF as the OPSF to derive S_a . Using S_a , the local OP calculates the needed assertion signature and creates the redirect message containing the OpenID parameters and the signed assertion. The browser is then redirected to the RP, which can in turn verify the assertion using the S_a , as exchanged in the association with the OPSF. No additional communication is needed between the RP and OPSF.

3.1 Additional Privacy Using Identifier Select Mode

In the generic description we have assumed that there is one shared secret S per local OP. This allows a single local OP to provide multiple identifiers for one user, e.g. for personal or business use. Since the security relevant operations of the local OP are in the smart card and cannot be modified by the user, the MNO has control over the identifiers that can be used by the local OP. The local OP will only create assertions for the identifiers which are enabled by the MNO. The user could for example create or buy identifiers which are then installed in the local OP if he needs separate identifiers for different purposes.

OpenID supports the so called OP-driven identifier selection [10, sec. 10], where the user only supplies the URL of its OP to the RP, e.g. `id.example`. The RP will then associate with the OP using the OP identifier `id.example` by setting `openid.identity` to `http://specs.openid.net/auth/2.0/identifier_select`. The user's browser is then again redirected to the local OP as described above. However, in this case, the local OP recognizes that identifier selection is used and displays a list of possible identifiers for the user to select between. This list is controlled by the MNO, and the user cannot select an identifier which is not in the list. After selection of an identifier, the local OP creates the signed assertion for

this identifier and redirects the browser back to the RP. The RP verifies the assertion and provides the service. As the RP does not need to communicate with the OPSF, the actual identifier used is not revealed to the OPSF and allows the creation of a privacy sensitive variant of the OpenID protocol. With a normal OpenID protocol run, the OP will always get to know the identifier at the time of user authentication. Hence, a higher level of privacy can be achieved as long as RP and OPSF do not collude.

While the use of the identifier select mode seems to have some restrictions, it can also be used for a slightly different purpose. To create unlinkability of user identifiers for the RP, a so called random id can be enabled in the local OP. Therefore the OPSF establishes a shared secret S_r with all local OP entities which have signed up for this service. The OPSF can provide a different OP URL, e.g. `r.id.example` which is then used in the identifier select mode. When the user is redirected to the local OP, the local OP allows the user to create a pseudo random identifier string, e.g. resulting in an identifier `a6gh8.r.id.example` and issue an assertion for this identifier. Since the identifiers can be created different for each login, they cannot be linked by the RP.

3.2 Attribute Assertions Using Identifier Select Mode

An additional use of the identifier select mode is the combination with attributes. Upon entering a contract with the MNO, the user has provided certain identity attributes, such as name, street address, age, etc.. As an example we consider age verification, where the user has to prove that he is over 18 to access the service of the RP. The RP, e.g. an online casino, is legally bound to verify the age of its customers. Implementing an age verification system can be costly for the RP and the RP therefore relies on a statement from an entity that he trusts, i.e. the MNO. The user wishes to only reveal sufficient identity data to enable access to the service also does not want to let the MNO know that he is into online gambling. The OPSF creates group keys G_i for attribute i and shares the group key with all local OPs that are members of that group, i.e. the user's local OP has a key G_{age} installed to enable the age attribute to be provided. The user now only enters the URL of his OP at the RP, which in turn engages in an OpenID authentication using identifier select mode with the OP identifier `age.id.example`. The OPSF then derives an association secret using G_{age} . When the local OP asks the user to select an identifier, the user can select the identifier 'over18', and the local OP then creates the assertion using a key derived from the association handle and G_{age} . The RP can now be assured that the user is over 18, because the assertion is coming with the MNO trust anchor. The user did not have to enter personal data to provide proof of age to the RP and at the same time did not reveal to the MNO the access to the online casino.

3.3 Implementation

We have implemented the local OP as an Android application on a Nexus One phone and successfully demonstrated OpenID logins to different existing RPs.

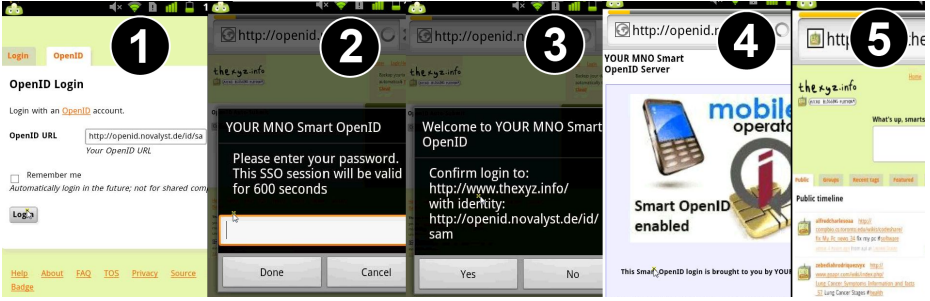


Fig. 4. Screenshots of the demo implementation, showing a login to thexyz.info

For the OPSF and local OP we used the Python based OpenID libraries [31] and SL4A [32] on the Android phone. In order to demonstrate the variants described in sections 3.1 and 3.2, we have also implemented a RP demo service. Our demo implementation shows that the protocol modifications are compliant to the standards and existing RPs don't need modifications to work with the local OP. The demo is software based and does not yet include access to the UICC, which will be added in the future. Figure 4 shows screenshots of a login process. The user enters his identifier (1) at the RP, and is then redirected to his local OP which displays a local authentication prompt, where the user is asked to provide his secret PIN code (2). After the user enters his code and confirms the login (3), a branding screen from the MNO can be displayed (4) and the service from the RP is provided (5). The user interface of the local OP is that of normal Android application creating a more seamless user experience.

3.4 Binding to Network Authentication

Instead of creating the shared secret between the local OP and the OPSF at the time of personalisation of the UICC, it is also possible to bind the use of the secret to a successful network authentication. This creates an additional level of security as the network authentication has to be successful in order for the local OP to engage in an OpenID authentication. Such a binding can for example be achieved with the GBA protocol [21]. The OPSF can be co-located with a NAF, similar to the OP/NAF combination in [16]. The local OP then runs GBA with the OPSF/NAF to get a Ks_NAF key which will then be used to derive the shared secret for the local OP and OPSF to be used with Smart OpenID.

As an alternative, there can be a combination of our proposal and the OpenID/GBA protocol [16]. When first logging in to a new RP, a full OpenID run using GBA is performed, resulting in the device and OPSF getting a Ks_NAF . Both entities then derive a RP specific K_RP key from Ks_NAF . Any subsequent login to that RP will then follow our Smart OpenID protocol and derive the association secret from the K_RP and have the local OP sign and issue the assertion. This method of binding has previously been described in [33]. A fringe benefit of this approach is a seamless and automated login and access to the RP

using the strong authentication security of the MNO. Alternative embodiments leveraging the MNO provisioned credentials such as OpenID with EAP-SIM or EAP-AKA to achieve a similar result are also feasible.

3.5 Analysis of Smart OpenID

Our solution does not require any changes to the OpenID protocol standards and remains compliant with existing RPs, which means that it can be directly deployed and enables access to a large variety of OpenID RP services. Some vulnerabilities of the standard OpenID protocol are addressed by the OpenID Security Best Practices [34]. Phishing of user credentials for an OpenID identifier should be of special concern, as OpenID enables access to all RP services. OpenID replaces multiple insecure passwords or a single password used across multiple services with one credential. Our solution increases the security by introducing a strong two-factor authentication, which is bound to the device and network authentication. Using a single point of authentication, namely the local OP, the user can easily and securely log in to different services, without having to remember different passwords. The user only has to remember his PIN code for the Smart OpenID application, which he has to provide for every authentication. This multi-factor authentication, based on possession of the card and knowledge of the PIN code, provides security from misuse in the case of a stolen or lost phone. In addition, using OTA mechanisms, the MNO can provide a portal for users, to remotely delete the smart card applet and thus prevent usage of stolen cards. The same portal could then allow users to register their new smart card with their existing identifier. Due to the local authentication, using the PIN code, credentials are not sent over the internet, preventing phishing and MitM attacks on the internet. It is possible to use varying grades of multi-factor local authentication, e.g. biometrics, to further increase the security.

The local OP provides not only a secure storage of credentials, but due to its function as an assertion issuing entity, also enables attribute based access control with some level of privacy for the user. The concept allows MNOs to easily provide IdM as a service to RPs, monetizing user data that they already possess. The MNOs leverage their existing infrastructure, user data and trust to provide attributes to RPs who can in turn reduce their costs in verifying user data. The local OP, based on the UICC, further enables multi-factor authentication for OpenID. As presented, Smart OpenID can create privacy towards the network and still issue trustworthy assertions on a user's identity or attributes to RPs.

In our experiments, we successfully logged in to various RPs on the internet. We simulated additional delays that might be introduced by the processing speed of a smartcard. The operations on the smart card which are needed for Smart OpenID can be limited to the crypto operations, so that the user does not experience a delay in the authentication. Since communication to the remote server in normal OpenID is replaced by faster local communication, the user even perceives less delay using the local OP. The smart cards currently used in mobile networks are equipped with anti-temper devices and using cryptographic protection which makes cloning of the card very difficult. Given the fact that an

attacker needs physical access in order to attack the card's contents, the user has enough time to react and report the loss to his MNO to block further usage of the card. As OpenID addresses the issues of the password dilemma, the presented combination with a strong, smart card based, local authentication provides an efficient protection from attacks, and creates a secure and seamless solution for SSO. Our current research is focussed on mobile scenarios, and hence uses the UICC for the local OP. Smart OpenID can be extended to other smart cards or security tokens, such as secure micro SD cards, which are all based on the Javacard platform. The split-terminal scenario described in [16] can serve as a blueprint for the extension of the concept to other devices, such as laptops.

4 Conclusions

OpenID as a lightweight protocol for IdM is increasingly being adopted by the industry. We have shown that by introducing an additional entity, the local OP, to existing smart card security, we can create a more secure OpenID implementation without requiring changes to the protocol. In addition, we have demonstrated that the use of the identifier select in combination with the local OP creates a higher level of privacy. This concept can be seen as an *IdM Enabler* as described in [9], allowing OpenID to be used by MNOs to offer new application services, which need elevated security and rely on trusted identity information. We have implemented the local OP and the OPSF and successfully demonstrated login to different RP websites without any modification to their OpenID implementation. As UICCs are in every mobile phone, they are a prime asset of MNOs which can provide the security for the Smart OpenID protocol.

References

1. Windley, P.: Digital Identity. O'Reilly Media, Inc. (2005)
2. Liberty Alliance Project. Web page at (2002), <http://www.projectliberty.org>
3. Chappell, D., et al.: Introducing windows cardspace. MSDN (April 2006)
4. Bertocci, V., Serack, G., Baker, C.: Understanding windows cardspace. Addison-Wesley Professional (2007)
5. Higgins Personal Data Service, <http://www.eclipse.org/higgins/>
6. Telco 2.0: Telco 2.0 Manifesto - Business Model Innovation for the Digital Economy, <http://www.stlpartners.com/manifesto.php>
7. Camenisch, J., Fischer-Huebner, S., Rannenberg, K.: Privacy and Identity Management for Life. Springer (2011)
8. Koschinat, S., Bal, G., Rannenberg, K.: Economic Valuation of Identity Management Enablers. PrimeLife Deliverable D6.1.2 (May 2011)
9. Koschinat, S., Bal, G., Weber, C., Rannenberg, K.: Privacy by sustainable identity management enablers. Privacy and Identity Management for Life, 431–452 (2011)
10. OpenID.net: OpenID Specifications, <http://openid.net/developers/specs/>
11. Uruena, M., Busquiel, C.: Analysis of a Privacy Vulnerability in the OpenID Authentication Protocol. In: IEEE Multimedia Communications, Services and Security (MCSS 2010), Krakow, Poland (2010)

12. van Thanh, D., Jonvik, T., Feng, B., Van Thuan, D., Jorstad, I.: Simple strong authentication for internet applications using mobile phones. In: IEEE GLOBECOM Global Telecommunications Conference 2008 (2008)
13. Urien, P.: Convergent identity: Seamless OpenID services for 3G dongles using SSL enabled USIM smart cards. In: Consumer Communications and Networking Conference (CCNC), pp. 830–831. IEEE (2011)
14. Leicher, A., Schmidt, A.U., Shah, Y., Cha, I.: Trusted Computing enhanced OpenID. In: 2010 International Conference for Internet Technology and Secured Transactions (ICITST), pp. 1–8 (2010)
15. Jorstad, I., Johansen, T., Bakken, E., Eliasson, C., Fiedler, M., et al.: Releasing the potential of openid & sim. In: 13th International Conference on Intelligence in Next Generation Networks, ICIN 2009, pp. 1–6. IEEE (2009)
16. 3GPP: Identity management and 3GPP security interworking; Identity management and Generic Authentication Architecture (GAA) interworking. TR 33.924, 3GPP (June 2011)
17. Chen, Z.: Java Card Technology for Smart Cards. Prentice Hall (2000)
18. ISO : ISO 7816-4: Identification cards - Integrated circuit cards - Organisation, security and commands for interchange (2005)
19. SIM Alliance: OpenMobile API Specification v2.0.2 (2011), <http://www.simalliance.org>
20. Tsyркlevich, E., Tsyркlevich, V.: Single Sign-On for the Internet: A Security Story. In: BlackHat Conference Las Vegas 2007 (2007)
21. 3GPP: 3G Security; Generic Authentication Architecture (GAA); System description. TR 33.919, 3GPP (June 2010)
22. Holtmanns, S., Niemi, V., Ginzboorg, P., Laitinen, P., Asokan, N.: Cellular Authentication for Mobile and Internet Services. Wiley (2009)
23. 3GPP: 3G security; Security architecture. TS 33.102, 3rd Generation Partnership Project (3GPP) (December 2010)
24. Weik, P., Wahle, S.: Towards a generic identity enabler for telco networks. In: Proc. 12th Internat. Conf. on Intelligence in Networks (ICIN 2008), Bordeaux, pp. 20–23 (2008)
25. Ahmed, A.S.: A User Friendly and Secure OpenID Solution for Smart Phone Platforms. Master’s thesis, Aalto University, School of Science and Technology, Faculty of Information and Natural Sciences (2010)
26. Urien, P.: An OpenID provider based on SSL smart cards. In: 7th IEEE Consumer Communications and Networking Conference, CCNC (2010)
27. Liberty Alliance: ID-WSF Advanced Client 1.0 Specifications. Technical report, (2007)
28. Liberty Alliance: ID-WSF Advanced Client Implementation and Deployment guidelines for SIM/UICC Card environment. Technical report (2007)
29. 3GPP: Remote APDU Structure for (U)SIM Toolkit applications. TS 31.115, 3GPP (December 2009)
30. 3GPP: Remote APDU Structure for (Universal) Subscriber Identity Module (U)SIM Toolkit applications. TS 31.116, 3GPP (December 2009)
31. Janrain: Python OpenID libraries, <http://www.janrain.com/openid-enabled>
32. Scripting Layer for Android, <http://code.google.com/p/android-scripting/>
33. Schmidt, A.U., Leicher, A., Shah, Y., Cha, I.: Efficient Application SSO for Evolved Mobile Networks. In: Proceedings of the Wireless World Research Forum Meeting 25 (WWRF 25), London, UK (2010)
34. OpenID Foundation: OpenID security best practices, <http://wiki.openid.net/OpenID-Security-Best-Practices>

Peer to Peer Botnet Detection Based on Flow Intervals

David Zhao¹, Issa Traore¹, Ali Ghorbani², Bassam Sayed¹, Sherif Saad¹, and Wei Lu³

¹ Department of Electrical and Computer Engineering
University of Victoria

davidzhao@ieee.org, {itraore,bassam,shsaad}@ece.uvic.ca

² University of New Brunswick

ghorbani@unb.ca

³ Keene State College

wlu@keene.edu

Abstract. Botnets are becoming the predominant threat on the Internet today and is the primary vector for carrying out attacks against organizations and individuals. Botnets have been used in a variety of cybercrime, from click-fraud to DDOS attacks to the generation of spam. In this paper we propose an approach to detect botnet activity by classifying network traffic behavior using machine learning classification techniques. We study the feasibility of detecting botnet activity without having seen a complete network flow by classifying behavior based on time intervals and we examine the performance of two popular classification techniques with respect to this data. Using existing datasets, we show experimentally that it is possible to identify the presence of botnet activity with high accuracy even with very small time windows, though there are some limitations to the approach based on the selection of attributes.

Keywords: Botnet, Network Intrusion Detection, Traffic Behavior Analysis, Network Flows.

1 Introduction

A bot is an autonomously operating software agent which may be controlled by a remote operator (the botmaster) to perform malicious tasks typically installed onto a victim's computer without the owner's consent or knowledge. Bots allow a remote operator to control the system and command it to perform specific, typically malicious tasks. Some of the tasks performed by a botnet include distributed denial of service (DDOS), mass spam, click fraud, as well as password cracking via distributed computing and other forms of cybercrime.

Command and control (C&C) is the key identifying characteristic of a botnet, and as such there is a variety of methods used by bots to form this network structure. Command and control channels must allow a botmaster to issue orders to individual bots in an efficient manner while at the same time avoiding being detected by computer security measures. Additionally, command and control channels would ideally want to be decentralized so that individual members are difficult to detect even if the C&C channel is compromised, allowing for resiliency in the network. As with

any problem, these three traits are frequently at odds with each other and botmasters and bots designers must make a tradeoff between stealthiness, decentralization and efficiency.

One of the most popular methods for implementing botnet command and control is by using the Internet Relay Chat (IRC) protocol [1]. IRC based C&C channels are highly efficient due to the ease of implementation as well as the capability of forming large networks very quickly due to the simplicity of the network architecture. Their weakness lies in their highly centralized nature: a compromise of a botnet C&C server may compromise the location of all bots connected to that server. Additionally, monitoring of network traffic may easily reveal the messages being passed from the server to individual clients, and much research has been done on botnet detection based on the analysis of these message contents.

C&C schemes based on HTTP traffic is another popular method for botnets. As a well-known protocol, HTTP based botnet C&C attempts to be stealthy by hijacking a legitimate communications channel in order to bypass traditional firewall based security and packets are often encrypted to avoid detection based on deep packet analysis. However, HTTP based C&C schemes still suffer from the issue of centralization, and it is possible to exploit such centralized behavior in their detection.

A more recent development in botnet C&C technology utilizes peer to peer (p2p) networks and protocols to form the communications network for bots. In p2p schemes, individual bots act as both client and server, producing a network architecture without a centralized point which may be incapacitated. The network is resilient in that when nodes are taken offline, these gaps may be closed automatically, allowing for the network to continue to operate under the attacker's control. [2] [3].

Feily et al. [4] divides the life-cycle of a botnet into five distinct phases: initial infection, secondary injection, connection, malicious command and control, and update and maintenance. In the initial infection phase, an attacker exploits a known vulnerability for a target system and infects the victim machine, granting additional capabilities to the attacker on the target system. In the secondary injection phase, the attacker uses his newly acquired access to execute additional scripts or programs which then fetch a malicious binary from a known location. Once the binary has been installed, the victim computer executes the malicious code and becomes a bot. In the connection phase, the bot attempts to establish a connection to the command and control server through a variety of methods, joining the botnet officially once this connection has been established. The maintenance phase is the last phase of the botnet lifecycle, bots are commanded to update their binaries, typically to defend against new attacks or to improve their functionality.

Leonard et al. divides a botnet's lifecycle into four phases: formation, C&C, attack and post-attack [5]. The attack phase is noted as a phase in the botnet lifecycle when the bot is actively performing malicious activities based on received instructions, while the post-attack phase is similar to the maintenance phase described in [4].

Many existing botnet detection techniques rely on detecting bot activity during the attack phase or during the initial infection / secondary injection phase. Typical detectors are based on traditional intrusion detection techniques, focusing on identifying botnets based on existing signatures of attacks by examining the behavior of underlying malicious activities.

In our work, we propose a method to detect the presence of peer to peer botnets not only during the attack phase, but also in the command and control phase. We examine the network behavior of a botnet at the level of the TCP/UDP flow, splitting it into multiple time windows and extracting from them a set of attributes which are then used to classify malicious (botnet) or non-malicious traffic using machine learning classification techniques. In particular, we compare the performance of the Bayesian Network classifier and a decision tree classifier using reduced error pruning (REPTree).

There are several advantages to detecting botnets based on their network flow characteristics. As a bot must communicate with the rest of the network during all phases after secondary injection, our approach may be used to detect the presence of a bot during a significant part of its life. Furthermore, detection based on network traffic characteristics is immune to encryption algorithms which may counter other approaches such as packet inspection and is computationally cheaper than those techniques. Additionally, by splitting individual flows into characteristic time windows, we may be able to detect bot activity quickly, before it has finished its tasks during the C&C or attack phases.

We organize the remainder of this paper in the following way: In Section 2, we provide an overview of existing botnet detection approaches and techniques. Section 3 provides our motivation and approach for the detection of botnets. In Section 4, we evaluate our approach using existing experimental datasets and compare the effectiveness of the two classification algorithms mentioned above which we have selected based on their performance and characteristics. Our concluding remarks and discussion of future work is provided in Section 5.

2 Related Work

A large collection of literature exists for the detection of botnets though interest towards the detection of peer to peer botnets has only recently emerged. Furthermore, botnet detection approaches using flow analysis techniques have only emerged in the last few years [6] and of these most examine flows in their entirety instead of smaller time intervals. Faily et al. classify botnet detection systems into four general types, signature-based detection, anomaly-based detection, DNS-based detection and mining-based detection [4]. Our focus will be on mining and anomaly based detection due to their increasing popularity.

Gu et al. proposed successively two botnet detection frameworks named BotHunter [7] and BotMiner [8].

BotHunter [7] is a system for botnet detection which correlates alarms from the Snort intrusion detection system with bot activities. Specifically, BotHunter exploits the fact that all bots share a common set of underlying actions as part of their lifecycle: scanning, infection, binary download, C&C and outbound scanning. BotHunter monitors a network and captures activity related to port scanning, outbound scanning and performs some payload analysis and malware activity detection based on Snort rules, and then uses a correlation engine to generate a score for the probability that a bot has infected the network. Like many behavior correlation techniques, BotHunter works best when a bot has gone through all phases of its

lifecycle, from initial exploit to outbound scan. BotHunter is also vulnerable to encrypted command and control channels that cannot be detected using payload analysis.

BotMiner [8] relies on the group behavior of individual bots within a botnet for its detection. It exploits the underlying uniformity of behavior of botnets and detects them by attempting to observe and cluster similar behavior being performed simultaneously on multiple machines on a network. BotMiner performs ‘C-plane’ clustering to first group network traffic behaviors which share similarities. Flows with known safe signatures (such as for some popular protocols) are filtered out of their list to improve performance. Once similar flows have been identified, BotMiner uses a second ‘A-Plane’ clustering technique which groups flows by the type of activities they represent using anomaly detection via Snort. By examining both the A-Plane and C-Plane, BotMiner correlates hosts which exhibit both similar network characteristics as well as malicious activity and in doing so identify the presence of a botnet as well as members of the network. Experimentally, BotMiner was able to achieve detection accuracies of 99% on several popular bot variants with a false positive rate around 1%.

Yu et al. proposed a data mining based approach for botnet detection based on the incremental discrete Fourier transform, achieving detection rates of 99% with a false positive rate of 14% [9]. In their work, the authors capture network flows and convert these flows into a feature stream consisting of attributes such as duration of flow, packets exchanged etc. The authors then group these feature streams using a clustering approach and use the discrete Fourier transform to improve performance by reducing the size of the problem via computing the Euclidean distance of the first few coefficients of the transform. By observing that individual bots within the same botnet tend to exhibit similar flow patterns, pairs of flows with high similarities and corresponding hosts may then be flagged as suspicious, and a traditional rule based detection technique may be used to test the validity of the suspicion.

Zeidanloo et al. proposed a botnet detection approach based on the monitoring of network traffic characteristics in a similar way to BotMiner. In their work, a three stages process of filtering, malicious activity detection and traffic monitoring is used to group bots by their group behavior. The approach divides the concept of flows into time periods of six hours and clusters these flow intervals with known malicious activity. The effects of different flow interval durations were not presented, and the accuracy of the approach is unknown.

All of the above mentioned group behavior clustering approaches requires that malicious activity be performed by the bots before detection may occur and therefore are unsuitable for early detection during the C&C phase of a bot’s lifecycle. Additionally, similarity and group behavior detection strategies relies on the presence of multiple bots within the monitored network and are unreliable or non-functional if only a single infected machine is present on the network.

Livadas et al. proposed a flow based detection approach for the detection of the C&C traffic of IRC-based botnets, using several classifiers to group flow behavior [10]. Their approach generates a set of attributes from IRC botnet flows and classifies these flows using a variety of machine learning techniques. Using a Bayesian network classifier, they achieved a false negative rate between 10% to 20% and a false positive rate of 30% to 40% though their results may have been negatively affected by poor labeling criterion of data. They showed using their approach that there exists a

difference between botnet IRC chat behavior and normal IRC chat behavior and that it was possible to use a classifier to separate flows into these categories.

Wang et al. presented a detection approach of peer to peer based botnets (specifically the peer to peer Storm variants using the Kademlia based protocol) by observing the stability of control flows in initial time intervals of 10 minutes [11]. They developed an algorithm which measures the stability of flows and exploits the property that bots exhibit similar behavior in their command search and perform these tasks independently of each other and frequently. This differs from the usage of the protocol by a normal user which may fluctuate greatly with user behavior. They show that by varying parameters in their algorithm, they were able to classify 98% of Storm C&C data as 'stable', though a large percentage of non-malicious peer to peer traffic were also classified as such (with a false positive rate of 30%). Our own approach is similar to this research, though we seek to significantly increase our detection accuracy by introducing new attributes and by utilizing a machine learning algorithm.

3 Approach

3.1 Foundation

Early works in botnet detection are predominantly based on payload analysis methods which inspect the contents of TCP and UDP packets for malicious signatures. Payload inspection typically demonstrates very high identification accuracy when compared with other approaches but suffer from several limitations that are increasingly reducing its usefulness. Payload inspection techniques are typically resource intensive operations that require the parsing of large amounts of packet data and are generally slow. Additionally, new bots frequently utilize encryption and other methods to obfuscate their communication and defeat packet inspection techniques. Furthermore, the violation of privacy is also a concern in the payload analysis-based detection scheme.

A more recent technique, traffic analysis, seeks to alleviate some of the problems with payload inspection. Traffic analysis exploits the idea that bots within a botnet typically demonstrate uniformity of traffic behavior, present unique communications behavior, and that these behaviors may be characterized and classified using a set of attributes which distinguishes them from non-malicious traffic and techniques. Traffic analysis does not depend on the content of the packets and is therefore unaffected by encryption and there exists dedicated hardware which may extract this information with high performance without significantly impacting the network.

Typical traffic analysis based detection systems examine network traffic between two hosts in its entirety. While this approach is feasible for offline detection, it is not useful for the detection of botnet behavior in real time. A network flow between two hosts may run for a few seconds to several days, and in many instances it is desirable to discover botnet activity as soon as possible.

In this paper, we present a detection technique based on traffic analysis which allows us to identify botnet activity in real time by examining the characteristics of these flows in small time windows. We exploit some properties of botnet traffic in order to perform this detection with high confidence even when other non-malicious traffic is present on the network.

The uniformity of botnet communications and botnet behavior is well known and has been exploited by various architectures towards their detection [8] [9] [12] [13] [14]. Most of these techniques exploit this uniformity by monitoring the traffic behavior of a number of machines, and then identifying machines which are part of a botnet when they begin to simultaneously perform similar malicious actions. Other methods include observing the command and response characteristics of bots; in the BotSniffer architecture, Gu et al. detect individual bots by drawing a spatial-temporal correlation in the responses of bots to a specific command [13]. With this idea, we make the assumption that should there exist a unique signature for the flow behavior of a single bot, we can use this unique signature to detect many bots which are part of the same botnet.

Several studies have shown that it is possible to detect certain classes of network traffic simply by observing their traffic patterns. Jun et al. proposed a technique of detecting peer to peer traffic based on a set of network flow attributes [15]. While the research does not focus on computer security but instead traffic classification, they nevertheless show that it is possible to detect various classes of peer to peer applications (eMule, Kazaa, Gnutella) based on their unique flow attributes. We also observe that bots utilizing implementations of the Overnet / Kademia p2p protocol as well as unique p2p implementations as those seen on the Waledac bot exhibit unique and specific message exchange characteristics, particularly when first joining their p2p networks [2] [16].

For our technique, we will analyze specifically the network flow characteristics of traffic on a network. For the purposes of our framework, we define a flow as a collection of packets being exchanged between two unique IP addresses using a pair of ports and utilizing one of several Layer 4 protocols. We observe the characteristics of a given flow by examining its traffic in a given time window T and make two observations about the size of the time window. First, if a time window is too small, we may fail to capture unique traffic characteristics that only become apparent over a longer period of time, and we may also introduce errors as the behavior of a flow may change over time. If a time window is too large, we cannot make a decision in our classification until the window has been met, which means that our time to detection will increase to an undesirably long period. Ultimately, the selection of a time window size will be based on a compromise between detection accuracy and speed.

In order to classify the flow characteristics, we compute a set of attributes for each time window which encodes relevant information about the behavior of the flow during that time window. The selection of our set of attributes is based on the observations we have made above, combined with our intuition of botnet messaging activities.

Operation of our detection framework consists of two phases. In the training phase, we provide our detectors with a set of known malicious and non-malicious data attribute vectors in order to train our classifiers in the identification of the two classes of data. Once complete, the system is placed in the detection phase, where it actively observes the network traffic and classifies the attribute vectors generated from active flows. When a set of attribute vectors has been classified as 'malicious' in the live data, the flows in question may be flagged as suspicious.

3.2 Attribute Selection

An attribute is some characteristic of a flow or a collection of flows in a given time window T which may be represented as a numeric or nominal value. Table 1 lists the set of 12 attributes we have selected for the purposes of our evaluation. Some attributes, such as the source and destination IP addresses and ports of a flow, may be extracted directly from the TCP / UDP headers, while others, such as the average length of packets exchanged in the time interval, require additional processing and computation. These attributes are then used as part of an attribute vector which captures the characteristics of a single flow for a given time interval.

Table 1. Selected network flow attributes

Attribute	Description
SrcIp	Flow source IP address
SrcPort	Flow source port address
DstIp	Flow destination IP address
DstPort	Flow destination port address
Protocol	Transport layer protocol or 'mixed'
APL	Average payload packet length for time interval.
PV	Variance of payload packet length for time interval.
PX	Number of packets exchanged for time interval.
PPS	Number of packets exchanged per second in time interval T
FPS	The size of the first packet in the flow.
TBP	The average time between packets in time interval.
NR	The number of reconnects for a flow
FPH	Number of flows from this address over the total number of flows generated per hour.

We selected our set of attributes based on the behavior of various well known protocols as well as the behavior of known botnets such as Storm, Nugache and Waledac. For example, we note that unlike normal peer to peer usage, bot communication may exhibit a more uniform behavior whereupon the bot queries for updates or instructions on the network continuously, resulting in many uniform sized, small packets which continuously occur. Another observation we may make is that for many protocols, the initial exchange of packets when a client joins a network tends to be unique and follows well defined behavior; this knowledge may allow us to assist in classification by capturing the characteristics of the initial packet exchange and carrying this information forward to subsequent time intervals for that flow. For instance, the first packet size attribute is obtained immediately when the initial flow has been established and is carried on to future time windows to assist in classification.

It should be noted that while included in our attribute list, the source and destination IP and port numbers for a flow may not be a very good attribute if the training data comes from a different network and uses different IP values. Typically we would like to use attributes which are universal to any network in order to provide for a more portable signature.

One final consideration for the selection of attributes is to provide some resistance to potential evasion techniques for bots. While no known bots today exhibit this evasion strategy, it is feasible that flow perturbation techniques could be used by a bot in an attempt to evade our analysis. A bot may, for example, inject random packets into its C&C communications in order to throw off correlations based on packet size. In order to mitigate some of these techniques, we measure the number of flows generated by a single address, and compare it with the number of total flows generated in some time period (in this case, an hour). This metric allows us to exploit the fact that most bots will generate more flows than typical non-malicious applications as it queries its C&C channels for tasks and carry out those tasks. We also measure the number of connections and reconnections a flow has made over time in case the bot attempts to randomly connect and disconnect to defeat our connection based metric. Like any service, it is desirable for a bot to be connected to its command and control channel as much as possible, and therefore any random disconnects a bot performs in order to defeat detection will naturally provide some mitigation against the bot's activities. Finally, it is possible to generate white lists of known IP addresses and services which help eliminate potential benign programs which may exhibit similar connection behavior to better isolate malicious applications. None of our proposed strategies are foolproof, but they serve to increase implementation complexity for the botmaster as well as provide natural detriments to the efficient operation of a botnet.

3.3 Classifier Selection

Many machine learning (ML) classification techniques exist which all attempt to cluster and classify data based on attribute sets. For purposes of our work, we would like to select classification techniques with a high performance in order to support real time detection goals while at the same time exhibiting high detection accuracy.

We have selected two popular classification techniques for our evaluation based on the above criteria, a Bayesian network classifier, and a decision tree classifier.

Bayesian networks (BN) are directed acyclic graphs where each node represents a domain variable, or in our case, a flow attribute, and each edge between nodes represents the probability of dependency. Given values assigned to other nodes, a BN may compute the conditional probability of one node occurring given values assigned to other nodes. These networks have been used in the classification of a variety of data, including network traffic data. Like most graphical classification techniques, Bayesian networks are easily visualized.

Decision tree based classifiers are a well known classification technique which exhibits desirably low computational complexity. In a decision tree, interior nodes represent input attributes with edges extending from them which correspond to

possible values of the attributes. These edges eventually lead to a leaf node which represents an output variable (in our case, whether a flow is malicious or non-malicious). Classification of an attribute vector simply consists of traversing the path from root to leaf by following edges which correspond to the appropriate values of the attributes in the attribute vector. Decision trees are learned via a partitioning process where the source attribute set is split into subsets based on a value test. This partitioning halts after a user-defined stopping criteria has been reached. For our evaluation, we select a decision tree using the Reduced Error Pruning algorithm (REPTree). This algorithm helps improve the detection accuracy of a decision tree with respect to noisy data, and reduces the size of the tree to decrease the complexity of classification.

4 Evaluation

4.1 Evaluation Dataset

There are considerable difficulties in obtaining real world datasets of botnet malicious activity. Many publicly available datasets consist of information collected from honeypots which may not reflect real-world usages. In a typical honeynet configuration, a honeypot is a machine dedicated for the collection of malicious data and typically is not used for other normal activities. In such a case, it is atypical to see non-malicious traffic within a honeypot network trace except in the smallest quantities, and such non-malicious data rarely reflect real world usage scenarios.

In order to evaluate our system, we attempt to generate a set of network traffic traces which contain both malicious and non-malicious traffic, including traffic from standard usage of popular networked applications. Malicious and non-malicious traffic are intermixed in a way that both types of traffic occur during the same time periods, and we label this data in order to evaluate the accuracy of our methods.

For this work, we obtained and used two separate datasets containing malicious traffic from the French chapter of the honeynet project involving the Storm and Waledac botnets, respectively [17]. Waledac is currently one of the most prevalent P2P botnets and is widely considered as the successor of the Storm botnet with a more decentralized communication protocol. Unlike Storm, which uses Overnet as a communication channel, Waledac utilizes HTTP communication and a fast-flux based DNS network exclusively.

To represent non-malicious, everyday usage traffic, we incorporated two different datasets, one from the Traffic Lab at Ericsson Research in Hungary [18] and the other from the Lawrence Berkeley National Laboratory (LBNL) [19]. The Ericsson Lab dataset contains a large number of general traffic from a variety of applications, including HTTP web browsing behavior, World of Warcraft gaming packets, and packets from popular bittorrent clients such as Azureus.

The LBNL is a research institute with a medium-sized enterprise network. The LBNL trace data consists of five datasets labeled $D_0 \dots D_4$; Table 2 provides general information for each of the datasets.

Table 2. LBNL datasets general information

	D ₀	D ₁	D ₂	D ₃	D ₄
Date	Oct 4, 04	Dec 15, 04	Dec 16, 04	Jan 6, 05	Jan 7, 05
Duration	10 min	1 hour	1 hour	1 hour	1 hour
Subnets	22	22	22	18	18
Hosts	2,531	2,102	2,088	1,561	1,558
Packets	18M	65M	28M	22M	28M

The recording of the LBNL network trace happened over three months period, from October 2004 to January 2005 covering 22 subnets. The dataset contains trace data for a variety of traffic which spans from web and email to backup and streaming media. This variety of traffic serves as a good example of day-to-day use of enterprise networks.

In order to produce an experimental dataset with both malicious and non-malicious traffic, we merged the two malicious datasets and the Erikson (non-malicious) dataset into a single individual trace file via a specific process depicted by Figure 2. First we mapped the IP addresses of the infected machines to two of the machines providing the background traffic. Second, we replayed all of the trace files using the TcpReplay tool on the same network interface card in order to homogenize the network behavior exhibited by all three datasets; this replayed data is then captured via wireshark for evaluation.

The final evaluation data produced by this process was further merged with all five LBNL datasets to provide one extra subnet to indeed simulate a real enterprise size

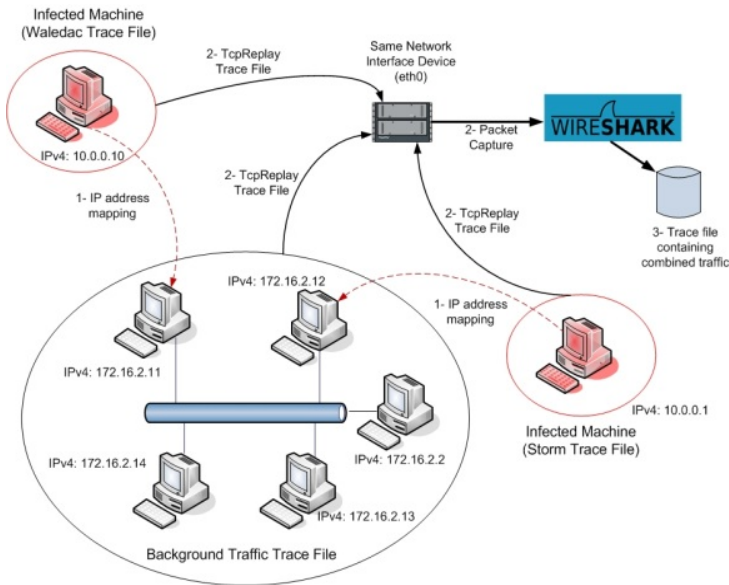


Fig. 1. Dataset merging process

network with thousands of hosts. The resulted evaluation dataset contains 22 subnets from the LBNL with non-malicious traffic and one additional subnet as illustrated in Figure 2 with both malicious and non-malicious traffic originating from the same machines.

4.2 Evaluation Results

We implemented our framework in Java and utilized the popular Weka machine learning framework and libraries for our classification algorithms [20]. Our program extracts from a given pcap file all flow information and then parses the flows into relevant attribute vectors for use in classification.

In all, there were a total of 1,672,575 network flows in our test set. The duration of the flows vary greatly, with some lasting less than a second and a few lasting more than a week. Of these flows, 97,043 (about 5.8%) were malicious, and the remainder non-malicious. From these flows, we generated 111,818 malicious attribute vectors, and 2,203,807 non-malicious attribute vectors. Each feature vector represents a 300 second time window in which at least 1 packet was exchanged. We consider malicious flow attribute vectors a vector which is extracted from a flow associated with the Storm or Waledec botnet data, and we considered all other attribute vectors as non-malicious, including peer to peer applications such as Bittorrent, Skype and e-Donkey.

To evaluate detection accuracy, we used the 10-fold cross-validation technique to partition our dataset into 10 random subsets, of which 1 is used for evaluation and 9 others are used for training. This process is repeated until all 10 subsets have been used as the testing set exactly once, while the remaining 9 folds are used for training. This technique helps us guard against Type III errors and gives us a better idea of how our algorithm may perform in practice outside of our evaluation data. The true and false positive rates of both the Bayesian Network and decision tree classifiers are listed in Table 2 and 3 respectively. The resulting detection values are an average of the results of the ten runs.

Table 3. Detection rate of Bayesian Network Classifier (attribute vectors identified)

Detection rate using Bayesian Network classifier (T = 300s)		
	True positive	False positive
Malicious	97.7%	1.4%
Non-Malicious	98.6%	2.3%

Table 4. Detection rate of REPTree classifier (attribute vectors identified)

Detection rate using decision tree classifier (REPTree) (T = 300s)		
	True positive	False positive
Malicious	98.3%	0.01%
Non-Malicious	99.9%	1.7%

As can be seen in the above tables, both the Bayesian Network classifier and the decision tree produced very high (above 90%) detection rates with a very low false positive rate. Between the two classifiers, the decision tree was more accurate, classifying 99% of all attribute vectors correctly while incorrectly identifying only 1% of all attribute vectors. These results indicate that there are indeed unique characteristics of the evaluation botnets when compared to everyday p2p and non-p2p traffic.

In terms of speed, the decision tree classifier was slightly slower than the Bayesian Network classifier, requiring 76.9 seconds for the full evaluation as compared to 56.1 seconds for the Bayesian Network.

In order to get some idea of the key discriminating attributes in our dataset, we use a correlation based attribute evaluator (CFS feature set evaluation) with best first search to generate a list of attributes with the highest discriminatory power while at the same time exhibiting low correlation with other attributes in the set. The algorithm generated a subset of four attributes, listed in Table 5, to be used as an optimized subset that may improve our performance without producing a large reduction in accuracy.

Table 5. Attribute subset from CFS subset evaluation

Feature	Description
PV	Variance of payload packet length for time interval.
PX	Number of packets exchanged for time interval.
FPS	The size of the first packet in the flow.
FPH	# flows per address / total flows

Tables 6 and 7 list the results of classification using only the above attribute subset. We can see that by reducing the number of attributes to three, the accuracy of the Bayesian network and REPTree classifiers both decreased slightly due to an increased false positive rate. Neither classifier with the reduced attribute set performed as well as the REPTree classifier with the full attribute set, though both very closely matched the best case's accuracy. In terms of performance, reducing the number of attributes allowed the Bayesian network classifier to classify all attribute vectors in 76% of the original time, while the REPTree classifier took 33% of the time. Table 8 shows the actual times for classifying all attribute vectors by the classifiers on both the full attribute set and the reduced set.

Table 6. Detection rate of Bayesian Network Classifier with reduced subset

Detection rate using Bayesian Network classifier (T = 300s)		
	True positive	False positive
Malicious	92.3%	1.9%
Non-Malicious	98.1%	7.7%

Table 7. Detection rate of REPTree classifier with reduced subset

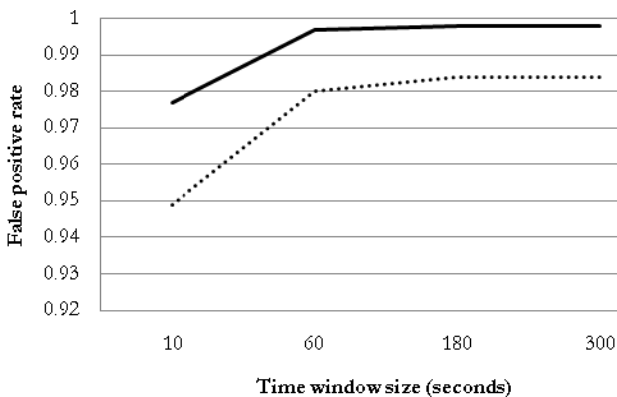
Detection rate using decision tree classifier (REPTree) (T =300s)		
	True positive	False positive
Malicious	98.1%	2.1%
Non-Malicious	97.9%	1.9%

Table 8. Classifier performance (average training time)

Performance of classifiers	
Classifier	Time (seconds)
BayesNet	40.6
REPTree	29.4
BayesNet (subset)	11.22
REPTree (subset)	8.58

With the above results, we may conclude that both the Bayesian network classifier and the decision tree classifier are suitable for the building of a botnet detection framework based on flow attributes. We further observe that while maximum accuracy may be achieved with a sizable attribute vector, we may be able to detect unique characteristics in bot traffic based simply on their packet exchange behavior, in particular the variations in their flow behavior compared to standard network traffic and the first packet exchanged in any new flows created by such malicious traffic.

Finally, we examine the effects of varying the size of our time window on the accuracy of detection. Figures 3 and 4 show the effects of the time window size on the true and false positive rates for malicious flow detection. The best results were obtained when $T = 180$ seconds for both the REPTree and BayesNet classifiers, though very good results were obtained for all four time window sizes (10, 60, 180 and 300 seconds).

**Fig. 2.** Effects of time window size on true positive rate for REPTree (*solid line*) and BayesNet (*dotted line*) classifiers.

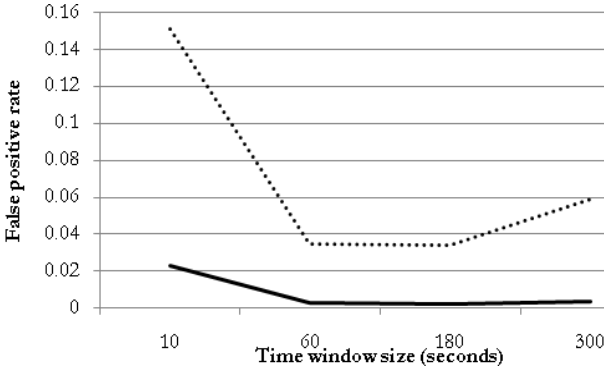


Fig. 3. Effects of time window size on false positive rate for REPTree (*solid line*) and BayesNet (*dotted line*) classifiers.

While both the performance and accuracy of our classifiers are satisfactory for real time detection, it must be noted that the system may be sensitive to new behaviors from bots implementing highly varied protocols. In order to guard against such a threat, a method for online training and continuous refinement of the classifiers must still be implemented.

4.3 Comparison with BotHunter

BotHunter [7] is one of the few botnet detection tools relevant to our work that is openly available. BotHunter mainly consist of a correlation engine that ties together alerts generated by Snort [21]. The tool consists of two custom plug-ins to snort called SLADE and SCADE. The SLADE plug-in mainly detects payload anomalies while the SCADE plug-in detects in/out bound scanning of the network. In addition to the two plug-ins, the tool includes a rule-set that is specifically designed to detect malicious traffic related to botnet activities such as Egg downloads and C&C traffic. The correlation engine ties all the alerts together and generates a report with the infection if any.

We used a recent version namely BotHunter version 1.6.0 and we tried to run it with our dataset. In practice, BotHunter is designed to run in live capture mode. However, it provides a script named runsnort.csh which allows running the tool offline against an existing dataset and generating an alert log file.

After running BotHunter against our dataset, the generated alerts indicated that there is a spambot in the dataset. More specifically, three alerts with “Priority 1” reporting the presence of botnet traffic were generated, however; the three alerts were all pointing to the same IP address. This IP address corresponds to a machine that was infected with Waledac botnet. BotHunter failed to detect the other machine that was infected with Storm botnet. In all, of the 97,043 unique malicious flows in the system, BotHunter was able to detect only a very small 56 flows. It is important to note that BotHunter itself does not work on the basis of flows, and the infected machine it detected is responsible for 17,006 malicious flows, (17% of the malicious traffic). It is important to also mention that BotHunter did not report any false positives.

While BotHunter does not expect or require that all phases of a bot lifecycle to be present in order to perform its detection, the fact that our dataset was missing the initial infection stages of the bot may have contributed to its poor detection performance.

5 Conclusion

In this paper we proposed a system for detecting bot activity in both the command and control and attack phases based on the observation of its network flow characteristics for specific time intervals. We emphasize the detection in the command and control phase because we would like to detect the presence of a bot before any malicious activities can be performed, and we use the concept of time intervals to limit the duration we would have to observe any particular flow before we may raise our suspicions about the nature of the traffic. We showed that using a decision tree classifier, we were able to successfully detect botnet activity with high accuracy by simply observing small portions of a full network flow, allowing us to detect and respond to botnet activity in real time. By comparing the true and false positive rates of our detector at various time window sizes, we have determined that a duration of 180 seconds provided the best accuracy of detection while a time window of 10 seconds was still able to produce an effective detector with a true positive rate of over 90% and a false positive rate under 5%.

There are limitations to our current approach that we hope to resolve in our future work. First, we recognize that our detection technique is based on the availability of existing malicious data and that in order for a detector to be truly robust we must develop a mechanism to evolve the classifiers to adapt to new threats. We are also aware that it is possible for a malicious botnet designer to obfuscate the network flow behavior of a bot in order to evade detection, even if such evasion would come at the expense of the effectiveness of a bot. To address these concerns, we are looking into the development of hybrid detectors which utilizes evolving classifiers along with our current approach, and defeat obfuscation by incorporating group behavior correlation.

References

1. Abu, R.M., et al.: A Multifaceted Approach to Understanding the Botnet Phenomenon. In: Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement. ACM, New York (2006) ISBN:1-59593-561-4
2. Grizzard, J.B., et al.: Peer-to-Peer Botnets: Overview and Case Study. In: Proceedings of the First Conference on First Workshop on Hot Topics in Understanding Botnets. USENIX Association, Berkeley (2007)
3. Holz, T., et al.: Measurements and Mitigation of Peer-to-Peer-based Botnets: A Case Study on Storm Worm. In: Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats. USENIX Association, Berkeley (2008)
4. Faily, M., Shahrestani, A., Ramadass, S.: A Survey of Botnet and Botnet Detection. In: Third International Conference on Emerging Security Information, Systems and Technologies (2009)

5. Leonard, J., Shouhuai, X., Sandhu, R.: A Framework for Understanding Botnets. In: International Workshop on Advances in Information Security. Fukuoka Institute of Technology, Fukuoka (2009)
6. Sperotto, A., et al.: An Overview of IP Flow-Based Intrusion Detection. *IEEE Communications Surveys & Tutorial* 12(3) (2010)
7. Gu, G., et al.: BotHunter: Detecting Malware Infection Through IDS-Driven Dialog Correlation. In: Proceedings of the 16th USENIX Security Symposium, pp. 167–182 (2007)
8. Gu, G., et al.: BotMiner: clustering analysis of network traffic for protocol- and structure-independent botnet detection. In: Proceedings of the 17th Conference on Security Symposium, San Jose (2008)
9. Yu, X., et al.: Online Botnet Detection Based on Incremental Discrete Fourier Transform. *Journal of Networks* 5(5) (2010)
10. Livadas, C., et al.: Using Machine Learning Techniques to Identify Botnet Traffic. In: 2nd IEEE LCN Workshop on Network Security, pp. 967–974 (2006)
11. Wang, B., et al.: Measuring Peer-to-Peer Botnets Using Control Flow Stability. In: International Conference on Availability, Reliability and Security, Fukuoka, p. 663 (2009), 978-1-4244-3572-2
12. Al-Duwairi, B., Al-Ebbini, L.: BotDigger: A Fuzzy Inference System for Botnet Detection. In: The Fifth International Conference on Internet and Applications and Services, Barcelona (2010)
13. Gu, G., Zhang, J., Lee, W.: BotSniffer: Detecting botnet command and control channels in network traffic. In: Proceedings of the 15th Annual Network and Distributed System Security Symposium (2008)
14. Ricardo, V.-S., José, B.C.: Bayesian Bot Detection Based on DNS Traffic Similarity. In: Proceedings of the 2009 ACM Symposium on Applied Computing, pp. 2035–2041. ACM, Honolulu (2009), 978-1-60558-166-8
15. Jun, L., et al.: P2P Traffic Identification Technique. In: International Conference on Computational Intelligence and Security, Harbin, pp. 37–41 (2007), 0-7695-3072-9
16. Sinclair, G., Nunnery, C., Kang, B.B.: The Waledac Protocol: The How and Why. In: Proceeding of 4th IEEE International Conference on Malicious and Unwanted Software (2009)
17. The HoneyNet Project, French Chapter | The HoneyNet Project. The HoneyNet Project, <http://www.honeynet.org/chapters/france>
18. Szabó, G., Orincsay, D., Malomsoky, S., Szabó, I.: On the Validation of Traffic Classification Algorithms. In: Claypool, M., Uhlig, S. (eds.) PAM 2008. LNCS, vol. 4979, pp. 72–81. Springer, Heidelberg (2008)
19. Lawrence Berkeley National Laboratory and ICSI., LBNL/ICSI Enterprise Tracing Project. LBNL Enterprise Trace Repository (2005), <http://www.icir.org/enterprise-tracing>
20. Witten, I.H., et al.: Weka: Practical Machine Learning Tools and Techniques (1999)
21. Roesch, M.: Snort - lightweight intrusion detection for networks. In: Proceedings of USENIX LISA 1999 (1999)

Towards a Universal Data Provenance Framework Using Dynamic Instrumentation

Eleni Gessiou¹, Vasilis Pappas², Elias Athanasopoulos²,
Angelos D. Keromytis², and Sotiris Ioannidis³

¹ Computer Science & Engineering Department
Polytechnic Institute of NYU
gessiou@cis.poly.edu

² Department of Computer Science
Columbia University

{vpappas,elathan,angelos}@cs.columbia.edu

³ Institute of Computer Science
Foundation for Research and Technology - Hellas
sotiris@ics.forth.gr

Abstract. The advantage of collecting data provenance information has driven research on how to extend or modify applications and systems in order to provide it, or the creation of architectures that are built from the ground up with provenance capabilities. In this paper we propose a universal data provenance framework, using dynamic instrumentation, which gathers data provenance information for real-world applications without any code modifications. Our framework simplifies the task of finding the right points to instrument, which can be cumbersome in large and complex systems. We have built a proof-of-concept implementation of the framework on top of DTrace. Moreover, we evaluated its functionality by using it for three different scenarios: file-system operations, database transactions and web browser HTTP requests. Based on our experiences we believe that it is possible to provide data provenance, transparently, to any layer of the software stack.

1 Introduction

All systems occasionally store their activity in log files, a process we usually refer to as logging. A log file can contain error messages, warnings, and important information for a user debugging a problem or investigating the condition of a running process. However, traditional logging is very limited. It lacks of semantic and critical information related to the internal state of an object. For example, a log file can hardly represent the state change for a database executing a particular query. Thus, we seek of tools and technologies that can monitor an object and collect intrinsic information associated with its internal state. More precisely, instead of having a collection of log files containing warning and error messages we are interested in the data provenance information.

Data provenance – describing how an object came to be in its present state – is an area that has drawn much research attention lately. There is a number of

schemes that have been proposed on how to apply it in practice and in widely used applications. Some representative provenance schemes have been proposed for file systems [18], databases [6], web applications [13,17,22], cloud computing [19], smart phone operating systems [8], and web browsers [16]. One could argue that applying data provenance at a low level, e.g. at the system call-level, would be sufficient. However, that is not always the case. Applying data provenance to different layers of the software stack can provide different levels of information. In the database case, for example, it would be much more efficient to apply it at the query-level, where all the data, along with any meta-information (like ownership) is available, than at the system call-level, where one would get very limited information like I/O operations on file blocks. Also, in the web browser case, data provenance provided solely at the system-call level will be unable to capture and record higher-level information. This can be the URLs the browser renders, in the case the communication channel is over HTTPS. On the other hand, the function-call level, as provided by OpenSSL [23], has access to the encrypted data stream used by an HTTPS connection.

By looking at these examples it becomes clear that depending on the case, data provenance must be provided at the layer closest to the information we would like to capture and document. This may prove to be a very challenging task, especially when dealing with large and complex software applications such as web browsers and databases. For example, Firefox 4 consists of more than 5M lines of code, spread in almost 40,000 files. In the case of MySQL 5.5, there are 1.2M lines of code, in almost 3,000 files.

In this paper, instead of exploring how we can extend an application to provide data provenance, we take an alternate route. We propose, and build, a universal data provenance framework using dynamic instrumentation. The main goal of our framework is to provide an easy way to prototype any data provenance scheme, no matter the size and complexity of the system it is applied on. We argue that lightening the implementation burden of such data provenance applications would lead research on this field forward, allowing researchers to test and evaluate their ideas more easily. Our framework is built on dynamic instrumentation, and especially on DTrace. Its key feature is that it can greatly assist the user in discovering paths in the system that interesting data pass through. For example, all URLs and search keywords that flow inside a browser. After the discovery phase, the user can dynamically instrument these points to record provenance about this transit data. Moreover, in cases where the source code of a system is available, our framework could be used for simply discovering points of interest in the system and evaluating a data provenance application. Then, prototyping a low-overhead source-code level implementation of the data provenance application at production level is trivial, as the user knows exactly which points to instrument in the system.

The choice of dynamic instrumentation for such cases seems ideal as it provides several advantages. First of all, it requires no changes to the source code of the system that data provenance capabilities are going to be added in. Second, it can be easily enabled or disabled, even at runtime in some cases, as we shall see

later on. Finally, it removes the requirement of having the source code, although having it could be helpful during the discovery phase. On the other hand, the main disadvantage of dynamic instrumentation is its runtime overhead. However, this is not an issue in our case as we perform system-call and function-level, but not instruction-level instrumentation, which can be extremely expensive.

Contribution. The contributions of this paper are the following.

- We deliver a generic instrumentation framework for acquiring information at the system-call and function-call level. The collected information can reveal data provenance information.
- We design techniques for easily discovering the important points, which are associated with particular state changes of a process. For example, our framework can automatically discover the functions that process SQL queries in SQLite.
- We provide three case studies with real world systems: (a) a file system, (b) a database (SQLite), and (c) a web browser (Safari).

2 Background

Tracing is the process of observing the execution of a program for collecting useful information of diagnostic and systemic nature. Various techniques have been developed for supporting this facility throughout the development cycle as well as after the deployment of a system. Instrumentation is one such technique that allows someone to augment the execution of a program with new, user-provided, code that aids in collecting data for analyzing the behavior of a system.

DTrace is a dynamic instrumentation utility that focuses on production systems. It allows the instrumentation of both user-level as well as kernel-level code in a unified and safe manner and has absolutely zero performance cost when disabled. Initially, DTrace was developed for Sun’s Solaris 10, but it has been integrated also into Apple’s Mac OS X/Darwin (since 10.5/9) [3], FreeBSD (since 7.1) [9], as well as into other microkernel designs such as QNX (i.e., the Unix-like, real-time OS for embedded systems) [20]. Moreover, Oracle Linux latest version included a port of DTrace for Linux [14].

Since the primary focus of DTrace is production systems, it was designed around two key properties: (a) zero performance cost when disabled and (b) absolute system safety when enabled. Its dynamic nature allows to be injected on demand into virtually every place of a running system without suffering from the performance burden of static “disabled probes”. Systems that support static instrumentation typically induce some disabled probes overhead. Dynamic instrumentation allows truly zero cost, since the probes are dynamically attached and detached on demand, and hence, they are “absent” when the instrumentation is disabled. User-provided instrumentation code, also known as analysis code is written in a high-level language, named D, that is subjected to a set of run-time checks for guaranteed safety.

D is C-like but it also resembles AWK [2] a lot in terms of structure. It has support for all ANSI C operators, it allows access to user- and kernel-level

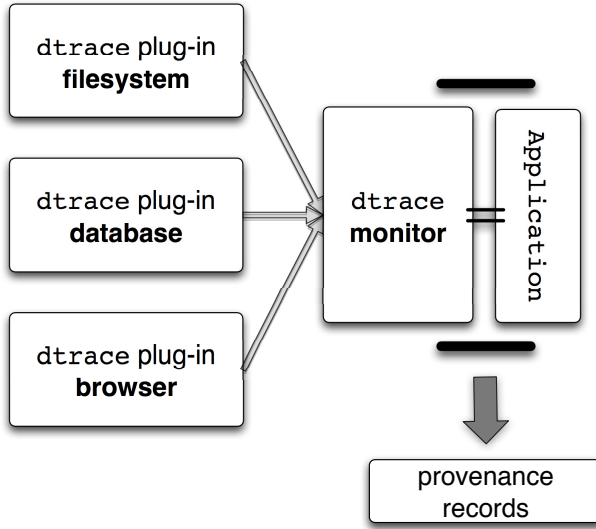


Fig. 1. A complex scenario where provenance information from different sources is combined

variables and data structures, and offers dynamic user-defined variables, structs, unions, and associative arrays. The scoping rules of the language, its intrinsic data types, as well the program structure are explained in great detail in [5].

The core part of DTrace lies inside the OS kernel and includes all the necessary facilities for providing an infrastructure for dynamic, arbitrary, tracing. User-level processes become DTrace consumers by communicating with that in-kernel component and enabling instrumentation. However, the DTrace framework does not perform any instrumentation of the system. This functionality is provided by the providers; kernel-level parts, typically loadable modules, that communicate with the core engine using a well-defined API. Providers declare to DTrace the points that can potentially instrument by providing a callback function. All in all, DTrace provides merely a skeleton for supporting future instrumentation methodologies. Nonetheless, it comes with a set of ready-to-use providers that have no observable overhead when disabled.

3 Design

The goal of the framework is to extract and gather provenance information related to the activity of complex systems, such as a web browser or a database. The collection of information is carried out by monitoring a process at the system-call and function-call level. This task requires the knowledge of all valid entry points, where data flows inside the running process. We refer to all system and function calls, which process critical information as valid entry points.

For example, a modern web browser, such as Mozilla Firefox, has a function that processes each input URL. This function is critical, since each processed URL denotes a semantic transition in the state of the web browser during its life cycle. The framework is generic enough for allowing information gathering from multiple valid entry points. The core engine of the framework is based on DTrace, which gives all low-level primitives for dynamic instrumentation of running processes. Furthermore, we provide two basic features for reducing the complexity associated with identifying *interesting* valid data entries. First, we provide a highly configurable logging component and, second, we provide *assisted discovery*. We now analyze both of these features in detail.

3.1 Configurable Logging

The logging component delivers the core functionality of our framework as it is responsible for generating provenance information. By default, the component monitors all system calls of all running process. For each monitored system call the following information is collected: (a) system-call arguments, (b) return value, (c) user id, (d) process name, (e) process id and (f) timestamp. The component provides an interface where a user can configure which processes and which system calls are monitored. This component can also be configured to log library function calls, by specifying the library's name along with the name of the function. For example, a security researcher who wants to monitor all symmetric cryptographic operations of a running program, can specify a monitoring set of valid entry points composed by: `EVP_EncryptInit_ex`, `EVP_EncryptUpdate_ex`, and `EVP_EncryptFinal_ex`. All these three functions can be found in the OpenSSL [23] library which is used as the *de facto* standard for cryptographic operations.

Another example is the following. Suppose that we want to log all the `write()` system calls issued by the text editor `vi`. In that case, the following script is generated by our framework (note that the predefined variable `arg0` holds the return value in the second probe):

```
syscall::write:entry
/execname == "vim"/
{
    self->arg1 = arg1;
}

syscall::write:return
/execname == "vim" && self->arg1/
{
    printf("%Y %s (%d) uid %d write: %s -> %d",
           walltimestamp, execname, pid, uid,
           copyinstr(self->arg1), arg0);
    self->arg1 = 0;
}
```

3.2 Assisted Discovery

Complex systems, such as web browsers and databases, are composed by a huge code base. While these systems run, there are many different transitions that can significantly alter their state. For example, each query processed by a database can trigger a group of transitions that will drive the database in a completely new state. It is challenging to identify all valid entry points, which initiate all these state transitions. A key feature of our framework is that it can greatly reduce the search space of such points. This is achieved by first instrumenting all the functions and checking for a specific value that the user has given as input in the argument list of the monitored system. If found, the name of the function, the call stack, or both, are logged. Then, the user can decide which of the collected functions qualify as valid entry points. The subset of functions narrowed down by our technique is significantly reduced compared to the overall size of the system in the average case.

Example: We now present a real-world example for a better understanding of how assisted discovery works. Suppose that we have a new application for data provenance in a database system. To evaluate its efficiency, we first need to log all the queries. Normally, we have two options. We can either download the source of the database and find the appropriate places in the code for logging the queries, or, we can proxy the inputs (queries) of the database system using an external software component. The first approach can be very difficult, depending on the size and complexity of the system, or completely infeasible, in case the source code is not available. The second one would be difficult to enable/disable as it would require restarting the database system. In addition, a proxy can hardly provide any information associated with internal states of the system. For example, consider a SQL query which is triggered by a user-generated query and it is initialized and executed internally in the database for optimization purposes. This query cannot be captured by the external proxy, since it is generated internally inside the database. On the other hand, our framework can easily discover the appropriate points for monitoring and logging any information we need transparently and on demand. This is achieved by instrumenting all the functions within the database system and search for a special argument value, at runtime. This argument value should be a query string, like ‘`SELECT assisted FROM discovery;`’, and the output of our framework will be a set of function names that had this character sequence as an argument.

4 Case Studies

To demonstrate the effectiveness of our framework, we evaluated it in three scenarios, namely file-system, database and web browser data provenance. The reason behind choosing these applications is because they have been already studied in the literature [18,6,16]. The case studies are presented in order of increasing complexity for our framework and, interestingly, the most complex one revealed some limitations that we consider in Section 5.

4.1 File-System

Data provenance in the context of file-systems was introduced in PASS [18]. To gather provenance information, PASS was implemented in the Linux kernel. Here, we show how we can gather similar information using our framework, without having to dig and alter the code of the kernel. We define the set of system calls that create or change the current state of data, as provenance in a file-system. This set includes the following systems calls: `creat`, `write`, `chmod`, `chown` and `unlink`. We also include the `open` system call for completeness. The output includes a timestamp (indicating the exact moment that the system entered a system call), the executable's name, the process id and the user id that makes the system call. Also, in case that the system call's arguments contain useful information, like the location of file, or the file's permissions, we log this information too.

An example of the information that is recorded by our framework when a user edits a file using the `TextEdit` application on Mac OS X is shown bellow:

```
2012 Jan 7 23:44:52 TextEdit (23624) uid 501 open: ../paper.txt -> 15
2012 Jan 7 23:44:52 TextEdit (23624) uid 501 write: 15, My paper -> 9
2012 Jan 7 23:44:52 TextEdit (23624) uid 501 close: 15 -> 0
```

This scenario shows that using the powerful logging capabilities of our framework we can gather system call level information about the file-system by simply specifying the set of the file-system related system calls, without performing any changes in the operating system itself.

4.2 Database (SQLite)

The second scenario we consider is in the context of databases and demonstrates the assisted discovery feature of our framework. Our goal is to locate the appropriate point (i.e., function) to log queries in SQLite [1]. Initially, we created a simple table with a few columns and added some test data. Then, our framework instrumented all the functions in `libsqlite3.dylib`¹ at runtime.

Then, while assisted discovery was enabled, we performed a few different types of queries (insertions, deletions, updates, etc.). Each time the given query string appears in the argument list of an instrumented function, assisted discovery logs it. An example of these logs is the following:

```
sqlite3 (27426) libsqlite3.dylib:sqlite3_prepare_v2 'create table ..'
```

Our framework logged several functions, but, just by looking at their names, it was obvious that `sqlite3_prepare_v2` was the most appropriate function to instrument in order to log all the queries. Then, given the syntax of each SQL command, we parsed them in order to keep track of any changes made to the database. Generally, the provenance information about the database entries include the creation (`create`) and deletion (`drop`) of tables and their contents

¹ `.dylib` is the filename extension for dynamically loaded libraries in Mas OS X.

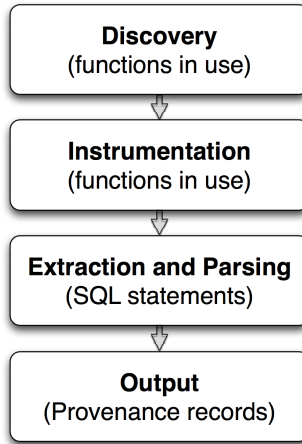


Fig. 2. The steps extracting provenance information from the SQLite database system using assisted discovery

(`inserts`) with corresponding timestamps. Also, the process name and id combined with the user id are logged for every SQL command that is executed. The procedure described above is also depicted in Figure 2.

We also calculated the performance overhead of logging provenance information using dynamic instrumentation versus logging the same information by altering the source code. We modified the `sqlite3_prepare_v2` function to record every SQL query made. Then, we measured the total time needed to perform one million `insert` operations in both versions of SQLite, the non-modified but dynamically instrumented and the modified one. On average, the performance overhead of dynamic instrumentation in this case was around 0.8%. Thus, for simple operations, like logging, there is no much difference performance-wise.

4.3 Web Browser (Safari)

Another example for applying the idea of provenance is that of the browser. We treat the notion of data provenance in a browser as it has been proposed in [16]. The main idea is the correlation between a search term and the URL that the user finally follows from the search engine. In other words, a search term is the provenance of the URL that the user asks for.

We implement this scenario by tracking the system calls that are called, such as `read`, `write`, `send` and `recv` using our framework. Although sufficient as a proof-of-concept implementation, we have to note that system call level tracking produces redundant information. It becomes too hard to separate the useful information for provenance from “noise”. Moreover, the system call level instrumentation cannot be used when data are encoded, encrypted, etc. Finally,

assisted discovery failed to locate functions, in a higher level than system calls, where the search keywords and the URLs could be logged. We discuss more about this in Section 5.

5 Discussion and Future Directions

The last usage scenario, extracting provenance information from a web browser, revealed some interesting limitations of our framework against very large and complex systems. These limitations are mostly due to the restricted instrumentation actions of DTrace. Combining a more powerful tool in our framework, like the dynamic binary instrumentation of Pin [15], seems sufficient to overcome these limitations, but in some extra cost. Also, tools implemented on top of Pin, like libdft [12] which provides byte-level data flow tracking, could potentially add more features to our framework.

Although Pin does not come built-in in any operating system, it can be installed in many of them, like Microsoft Windows, Linux and Mac OS. As DTrace, Pin can also be used to instrument arbitrary functions and system calls, but not on demand. The program to be instrumented has to be executed through Pin. The main advantage of Pin over DTrace though is that there is no limit in what an instrumentation actions can be. Although that may not seem very important, there are at least two cases that this feature extends the functionality of our framework. First off, we can better track the data in the assisted discovery phase when it is encoded, or encapsulated inside a complex type. As an example, suppose that we want to track a string value in a large software system that happens to use a custom string class and we know nothing about its internal representation. By using Pin during the assisted discovery phase though we can either convert them to simple character sequences or use any specific compare function they provide. The second advantage of unconstrained instrumentation actions is the ability to alter the state and the data of the program.

To better understand and demonstrate the extra functionality that such a tool adds to our framework, let us consider again the example of implementing data provenance in a browser. More precisely, suppose that in order to implement a hypothetical data provenance scheme we want to log all the URLs that a user navigated to in the Safari web browser. Safari uses the WebKit open source HTML rendering engine, where URLs are represented by `class KURL`. To successfully track a user-input URL during the assisted discovery phase, the instrumentation action has to compare `KURL` objects with the input string. Then, after we locate the appropriate function to instrument in order to log all the URLs, our framework will convert the `KURL` object to a simple string and then log it along with a timestamp, etc.

6 Related Work

Data provenance has been a very active field of research lately, focusing both on theoretical and practical issues. We discuss below the works most related to our dynamic data provenance framework.

Most of the recent work has been on data provenance in database systems [4]. Also, more theoretical aspects of the provenance notions (where, why and how) have been studied [6]. In our work we focus on the practical aspects of data provenance, on any software application and not simply databases.

The authors of [16] describe how a browser enabled with provenance capabilities can improve user experience. They provide several scenarios for history search, web search and download management, that a browser with provenance can offer. In Section 4.3 we show an example of how this kind of data provenance application could be achieved using our framework, without any modification to the actual web browser.

Story Book is a system that adds provenance in different systems [21]. It runs in user space and treats provenance events as a generic event log. It collects application-specific provenance and supports a file system and a database. PASS [18] supports provenance at the system level and is a layer grafted in a file system. It gathers provenance for all file activities by inspecting system calls. Again, an example of how this could be implemented using our framework is given later on. As an advantage, our approach requires no modifications to the file system, or any other layer for that matter.

Another system that combines both static and dynamic analysis to trace provenance is Garm [7]. While our framework is transparent, Garm has to rewrite the binary when the binary executed. The main advantage of our approach over Garm's is that our framework can be applied to any level of the system, capturing even very high-level information. On the other hand, Garm only operates on low level data operations.

Recently, the notion of provenance has also been introduced in cloud infrastructures [19]. Our framework could be successfully applied in such environments as well, as it is execution environment agnostic.

Transient provenance [10] keeps information about emigrant data, like files that were moved, who moved them, and when they were moved, aiming at facilitating the administrators in case of a leak. The authors propose the creation of ghost objects so as to be able to record when a file is leaked from the central system, and moreover by whom, by logging the user ID. This way, after an information leakage happens, the suspects, the timeline and the objects of leakage can be reduced up to the people, timestamps and files, recorded by the ghost objects, giving administrators a valuable hint.

iLeak [11] is a system proposed for detecting inadvertent information leaks. Although not related to data provenance, iLeak is related to our work as it is also implemented on top of DTrace. Any sensitive data within a system have to be initially marked with the appropriate tags. Then, iLeak constantly monitors for their leakage by dynamically instrumenting the system's communication channels (sockets, SSL connections, etc.) using DTrace.

7 Conclusion

We designed and implemented a data provenance framework that can be used for rapid prototyping of any data provenance scheme. Our framework is built on dynamic instrumentation and provides an easy way to locate the right points to instrument, as well as a configurable logging component.

Our experience from implementing three real-world examples of data provenance applications, already proposed in the literature, demonstrated the easiness and practicality of our framework. Moreover, some limitations that were revealed under very large and complex systems open room for improving our framework by integrating more powerful dynamic instrumentation tools.

Acknowledgements. The project was being co-financed by the European Regional Development Fund (ERDF) and national funds, was a part of the Operational Programme “Competitiveness & Entrepreneurship” (OPCE II), Measure “COOPERATION” (Action I). This work was also supported in part by DARPA Contract FA8650-11-C-7190, NSF Grant CNS-09-14312, FP7-PEOPLE-2010-IOF project XHUNTER and Marie Curie Actions - Reintegration Grants project PASS. Any opinions, findings, conclusions or recommendations expressed herein are those of the authors, and do not necessarily reflect those of the US Government, DARPA, or the NSF.

References

1. Sqlite, <http://www.sqlite.org/>
2. Aho, A.V., Kernighan, B.W., Weinberger, P.J.: The AWK Programming Language. Addison-Wesley (1988)
3. Apple. dtrace(1) Mac OS X Manual Page, <http://developer.apple.com/mac/library/documentation/Darwin/Reference/ManPages/man1/dtrace.1.html>
4. Buneman, P., Tan, W.-C.: Provenance in databases. In: Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data, SIGMOD 2007, pp. 1171–1173. ACM, New York (2007)
5. Cantrill, B.M., Shapiro, M.W., Leventhal, A.H.: Dynamic instrumentation of production systems. In: Proceedings of the USENIX Annual Technical Conference (ATC), pp. 15–28 (2004)
6. Cheney, J., Chiticariu, L., Tan, W.-C.: Provenance in Databases: Why, How, and Where. Foundations and Trends in Databases 1(4), 379–474 (2007)
7. Demsky, B.: Garm: cross application data provenance and policy enforcement. In: Proceedings of the 4th USENIX Conference on Hot Topics in Security, HotSec 2009, p. 10. USENIX Association, Berkeley (2009)
8. Dietz, M., Shekhar, S., Pisetsky, Y., Shu, A., Wallach, D.S.: Quire: Lightweight provenance for smart phone operating systems. In: Proceedings of the 20th USENIX Security Symposium, San Francisco, CA (August 2011)
9. FreeBSD. DTrace – FreeBSD Wiki, <http://wiki.freebsd.org/DTrace>
10. Jones, S., Strong, C., Long, D.D.E., Miller, E.L.: Tracking emigrant data via transient provenance. In: Proceedings of the 3rd USENIX Workshop on the Theory and Practice of Provenance (TaPP 2011), Heraklion, Greece (June 2011)

11. Kemerlis, V.P., Pappas, V., Portokalidis, G., Keromytis, A.D.: iLeak: A lightweight system for detecting inadvertent information leaks. In: Proceedings of the 6th European Conference on Computer Network Defense (EC2ND), Berlin, Germany, pp. 21–28 (October 2010)
12. Kemerlis, V.P., Portokalidis, G., Jee, K., Keromytis, A.D.: libdft: Practical dynamic data flow tracking for commodity systems. In: Proceedings of the 8th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE), London, UK (March 2012)
13. Lakshmanan, G.T., Curbera, F., Freire, J., Sheth, A.: Guest editors' introduction: Provenance in web applications. *IEEE Internet Computing* 15(1), 17–21 (2011)
14. Linux, O.: Trying out dtrace, http://blogs.oracle.com/wim/entry/trying_out_dtrace
15. Luk, C.-K., Cohn, R., Muth, R., Patil, H., Klauser, A., Lowney, G., Wallace, S., Reddi, V.J., Hazelwood, K.: Pin: building customized program analysis tools with dynamic instrumentation. In: Proceedings of the 2005 ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2005, pp. 190–200. ACM, New York (2005)
16. Margo, D.W., Seltzer, M.: The case for browser provenance. In: Proceedings of the First Workshop on on Theory and Practice of Provenance, pp. 9:1–9:5. USENIX Association, Berkeley (2009)
17. Michaelis, J.R., McGuinness, D.L.: Towards Provenance Aware Comment Tracking for Web Applications. In: McGuinness, D.L., Michaelis, J.R., Moreau, L. (eds.) IPAW 2010. LNCS, vol. 6378, pp. 265–273. Springer, Heidelberg (2010)
18. Muniswamy-Reddy, K.-K., Holland, D.A., Braun, U., Seltzer, M.: Provenance-aware storage systems. In: Proceedings of the Annual Conference on USENIX 2006 Annual Technical Conference, p. 4. USENIX Association, Berkeley (2006)
19. Muniswamy-Reddy, K.-K., Macko, P., Seltzer, M.: Provenance for the cloud. In: Proceedings of the 8th USENIX Conference on File and Storage Technologies, FAST 2010, pp. 14–15. USENIX Association, Berkeley (2010)
20. QNX. The community portal for qnx software developers, <http://community.qnx.com/sf/projects/dtrace/>
21. Spillane, R., Sears, R., Yalamanchili, C., Gaikwad, S., Chinni, M., Zadok, E.: Story book: an efficient extensible provenance framework. In: Proceedings of the First Workshop on Theory and Practice of Provenance, pp. 11:1–11:10. USENIX Association, Berkeley (2009)
22. Theoharis, Y., Fundulaki, I., Karvounarakis, G., Christophides, V.: On provenance of queries on semantic web data. *IEEE Internet Computing* 15, 31–39 (2011)
23. Viega, J., Messier, M., Chandra, P.: Network security with OpenSSL. O'Reilly Media (2002)

Improving Flask Implementation Using Hardware Assisted In-VM Isolation

Baozeng Ding^{1,2}, Fufeng Yao^{1,2}, Yanjun Wu¹, and Yeping He¹

¹ Institute of Software, Chinese Academy of Sciences, Beijing 100190, China

² Graduate University, Chinese Academy of Sciences, Beijing 100049, China

{sploving1,yffbrave,yanjun.wu}@gmail.com, yeping@iscas.ac.cn

Abstract. The Flask architecture, which mainly contains object manager (OM) and security server (SS), is widely used to support flexible security policies in operating system. In nature, OM and SS should be isolated from each other to separate decision from enforcement. However, current implementation of Flask, such as SELinux and SEBSD, puts both OM and SS in the same address space. If one component is subverted, the whole system will be exposed to the attacker. In this paper, we present hardware assisted in-VM isolation to improve the security of the Flask implementation. The key of our approach is the separation of SS from other parts of guest OS by constructing hardware assisted page tables at the hypervisor level. In this way SS can execute in a strongly isolated address space with respect to its associated guest OS, and therefore can provide a trustworthy and centralized repository for policy and decision-making. Our experiment shows that our method introduces moderate performance overhead.

Keywords: Flask Architecture, Virtualization, Security, In-VM Isolation, Extended Page Tables.

1 Introduction

Since operating system kernels occupy a privileged position in the software stack of computer systems, exploiting kernel-level vulnerability gives attackers completely control of systems. So protecting the integrity of operating system is very important. The Flask architecture [1], which mainly contains object manager (OM) and security sever (SS), is widely used to support flexible security policies in operating system. Various security modules such as SELinux [2], SEBSD [3], have been implemented using the Flask architecture. Since such modules play an important role in the system security, it is important to protect them from being attacked. In nature, OM and SS should be isolated from each other to separate decision from enforcement. However, current implementation of Flask, places both OM and SS in the same address space. If one component is subverted, the security module will be broken and the system will be exposed to the attacker.

As far as we know, there is no effective ways to protect the Flask implementation. Kernel-level integrity protection of Flask is not enough, as these protection

mechanisms reside in the same privilege level as the operating system kernel. They could be potentially bypassed or subverted. Virtualization gives us a chance to improve the security of Flask implementation. As the virtual machine monitor (VMM), also known as the hypervisor, exists on the lower level than the kernel, it could be used to detect and prevent the attacks against the Flask. Livewire [4], VMwatcher [5], Lares [6], Patagonix [7] raised to protect the guest kernel integrity utilizing virtualization. These approaches leverage a separate trusted VM to monitor the untrusted VM. Although these out-of-VM approaches have good security effect, they are suitable only for monitoring the events that occur infrequently during system running because switching between VMs causes large overhead. So it's not appropriate for protecting Flask architecture which is used to actively monitor those events occurring frequently.

SIM [8] is an In-VM monitoring framework that puts security tools in the same VM. It utilizes shadow page tables (SPT) mechanism [23] to create a separate virtual address space called SIM address space in the guest VM. It's a one-side view address space and exists in parallel to the system address spaces used by guest kernel. That means the code in the kernel address space cannot visit SIM address space directly but the code in the SIM address space can visit the kernel code. Security monitor is placed in the SIM address space so that it can be isolated from the untrusted kernel, and therefore is protected. It also leverages a feature of Intel VT to reduce the overhead when switching between the system address space and the SIM address space. This way gives SIM the performance benefits and the same security level as out-of-VM method. However, the SPT mechanism that SIM uses will result in expensive VM exits and context switches when synchronizing the shadow page tables with guest page tables [19]. Moreover, the source code structure of the shadow page tables is quite complex and not easy to maintain [19].

In this paper, we utilize Intel Extended Page Table (EPT) technology [21] to design a hardware assisted in-VM framework to isolate SS from the kernel. Isolated SS provides a trustworthy and centralized repository for policy and decision-making. EPT is an Intel hardware virtualization technology that is used to support the virtualization of physical memory. It controls the translation from guest physical addresses to the host physical address. The extended page table is shared by all the process in the guest VM and we call it system EPT which constructs system address space. In order to construct an isolated address space for SS, we create an independent EPT called SS EPT. This EPT maps all the physical addresses that belong to the guest VM, which constructs security server address space. Then we modify the origin system EPT to eliminate the mapping for SS code and data. In this way, the code in the kernel cannot access SS code and data directly. It has to enter security server address space by switching from system EPT to SS EPT to call SS function. We use UnixBench [11] to evaluate our prototype. The result shows the overhead is moderate. As AMD Nested Page Tables (NPT) [24] technology is similar to EPT, our work can be extended to the system with an AMD CPU. To the best of our knowledge, our work is a first step towards constructing in-VM isolation using hardware assisted page tables to improve Flask implementation.

2 Background

2.1 The Flask Architecture Overview

Flask [1] is an operating system security architecture that aims at supporting diverse security policies, as shown in Figure 1. OM is used to enforce security policy decisions returned by SS. Access Vector Cache module (AVC), is used to store the access decisions provided by SS for future use so that it can reduce the performance overhead.

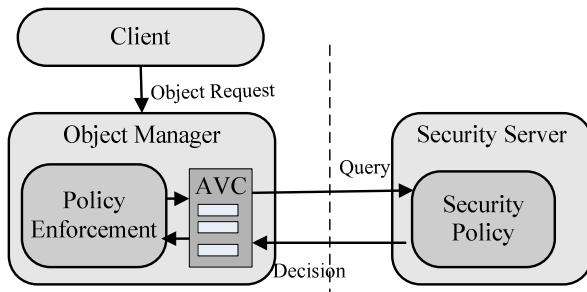


Fig. 1. The Flask architecture

When the system executes an operation, opening a file for example, OM will enforce security policy decision for this operation. OM queries AVC for the access decision. If AVC has stored such rules, it passes the result directly to OM. If no valid entries exist, AVC will send the query to SS. SS will look up the policy database, find the access decision and return it to AVC. AVC saves this decision, and then returns the result to OM.

2.2 EPT Overview

Without virtualization, memory management unit (MMU) is used to translate virtual addresses (VA) to physical addresses (PA) using page tables. But in a virtualized environment, the hypervisor takes full control of the host memory which is shared among virtual machines. So the guest VM can't see PA and guest page tables can't be used directly. In order to abstract the physical memory, the hypervisor presents guest physical addresses (GPA) to VM and maintains the mapping from GPA to host physical addresses (HPA). This gives the VM an illusion that it owns the actual physical memory.

EPT mechanism [21] is an Intel hardware virtualization technology that can be used to support the virtualization of physical memory. It is used to map GPA to HPA. Figure 2 shows the address translating process using EPT technology. First, the guest page tables pointed by CR3 register are used to translate VA to GPA. Then the EPT pointed by EPT base pointer is used to translate GPA to HPA. EPT has a structure of four level page tables. The top level is page map level 4 table, the second level is page directory pointer table, the third is page directory table and the last is page table.

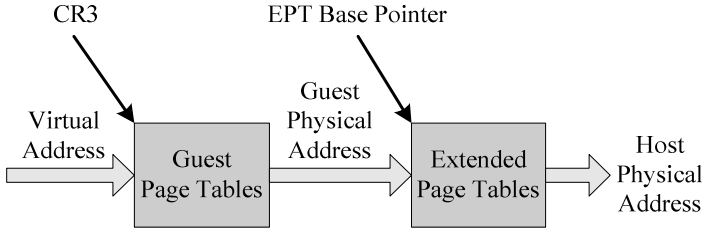


Fig. 2. Address translation using EPT

3 Assumption and Threat Model

Our framework is based on the following security assumptions. First, we assume that the operating system runs on a virtualization environment. Second, we assume that the underlying hypervisor is trusted. Recent hypervisor rootkits [13-14] make the hypervisor insecure. There are already some research such as HyperGuard [15], HyperCheck [16], HyperSentry [17] and HyperSafe [18] which aim at protecting the integrity of the hypervisor. In our prototype, we assume that the attacker cannot control the hypervisor to do malicious things. Especially the attacker cannot manipulate the EPT maintained by the hypervisor. Third, we assume OM hooks can be protected so that they cannot be removed or changed maliciously to bypass the security checks enforced by SS. Protecting these hooks are out of the scope for our work. It can be done using other research work like HookSafe [12]. In this way, we can guarantee the security operation cannot be bypassed. Finally, the boot of guest VM is trusted and the guest is clean before we enable our setting.

Our threat model allows an attacker to gain access to the kernel address space and to carry out the following type of attack:

1. SS code attack:

The attacker can modify SS code or inject malicious code to make SS not functional.

2. SS data attack:

The attacker may carry out control flow attack targeting SS stack or heap data. For example, stack buffer overflow or heap overflow attack. Also, non-control data attack is possible. It is pointers out that non-control data, like global variable, can be exploited to launch attack [22]. For example, a Boolean variable in SS that used to determine whether authentication is passed or not can be flipped to circumvent the authentication. Moreover, the attacker may take control of AVC entries and security policy data which may cause the Flask architecture not work as expected.

4 Overview of Our Approach

In order to make SS trustworthy to provide a centralized repository for policy and decision making, we isolate SS from kernel, including OM and other parts of the kernel, using hardware assisted in-VM isolation technique, as shown in Figure 3.

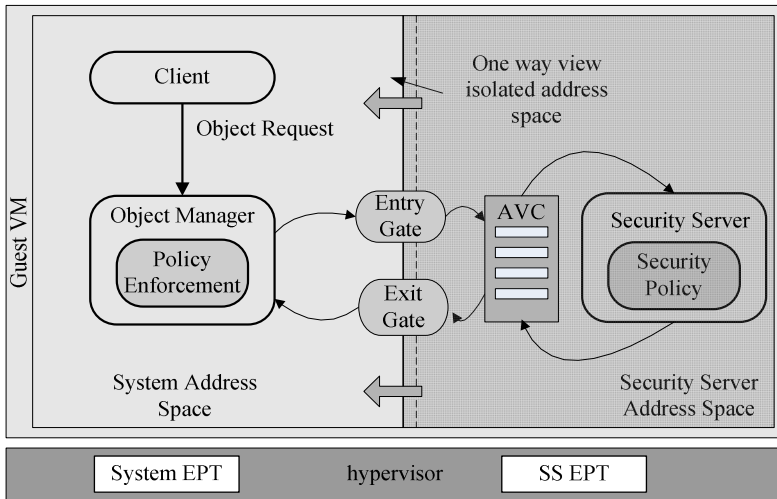


Fig. 3. Hardware assisted in-VM isolated Flask framework

We introduce a security server address space that is independent from and invisible to the system address space used by the operating system. We place SS together with AVC in it, as AVC is subverted, it makes no sense to isolate and protect SS. For clarity, when we refer to SS in the following section, it includes SS and AVC together. We create security server address space by constructing an independent EPT, called SS EPT, at the hypervisor level. This EPT is a synchronized copy of system EPT and is only used by SS for address translation. At the same time we remove the mapping to SS in the original system EPT. In this way, the code in the system address space cannot view and access SS code and data directly.

The entry and exit gates are used to switch address spaces by loading different EPTs. When the kernel calls SS function, it uses the entry gates to enter security server address space by switching from system EPT to SS EPT. When the call is finished, it uses the exit gates to return to system address space by switching from SS EPT to system EPT. The entry and exit gates also switch the stacks as SS need have its own to prevent attacks against stack data. The entry gates perform the transfer out of system stack into security server stack. The exit gates perform the switch from security server stack to system stack. The entry and exit gates are the only regions mapped into both address spaces so that a transfer between system address space and security server address space can only happen through code contained in these regions. They are write-protected so that their contents cannot be modified by any code in the kernel.

5 Implementation

In our prototype, we use Xen as the hypervisor and SELinux in the HVM DomU with 2.6.32 Linux kernel as the implementation of the flask architecture.

First, we construct an independent address space for SS by creating a separate EPT. Then we create entry and exit gates to achieve the switching between two EPTs. At last, we load a kernel module to take all the settings into effect.

5.1 Isolated Address Space Construction

Generally, SELinux which includes OM and SS is compiled into the kernel. SS code and kernel code may be sharing one physical page. SS data is similar. In order to isolate SS from other parts of the kernel, we need modify the Linux kernel linker script file to guarantee SS code and data are page aligned. So they are not mixed with code and data from the kernel. Take x86 architecture for example, when the kernel is compiled, a link script file called “vmlinux.lds” is generated according to the source file “vmlinux.lds.S”. It’s used to control the kernel linking. Here we modify its source file so that the changes will take effect every time vmlinux.lds is generated. In vmlinux.lds.S, we make SS code and data page align and add four indicators to record the start and end address of SS code and data. After the kernel building is completed, we can lookup System.map table to find the values of indicators to see whether the change is applied successfully.

After that, in order to construct an EPT that is only used by SS to execute address translation, we add a hypercall named “ept_create”. It is invoked only once. After guest VM is booted up, we load a kernel module to notify the hypervisor to invoke this function. It completes the following two tasks.

First, this hypercall creates a new EPT. For clarity, we would refer the EPT used for guest VM as system EPT, and refer the EPT we create for SS as SS EPT. This hypercall uses the system EPT as a template. Since EPT has a structure of four level page tables, we need allocate the physical pages for them and set their entries. First of all, the hypercall allocates a physical page as the new top level page table for SS EPT. Then it checks the top level page table entries of system EPT. For each valid entry that points to a second level page table of system EPT, the top level page table of SS EPT will be set the corresponding entry to point to a new page allocated as a second level page table of SS EPT. As for the access permission of the entry, the hypercall will copy it from the top level page table entry of system EPT. We do the similar operations for the second and the third page table of SS EPT. The last level page table is dealt with differently. As for the page table entries, we assign them exactly the same value as those in system EPT. When all is done, we get an independent EPT which has its own top level, second level, third level and last level page tables. But it maps the same physical memory as system EPT.

Second, the hypercall changes access permissions in system EPT. In order to isolate SS from the guest OS, SS code and data can only be seen and used in the security server address space. So the hypercall removes the mapping for them from system EPT. The hypercall “ept_create” takes four parameters when it’s called. Two of them represent start and end of guest frame number of SS code section. The other two are for SS data section. For each guest frame number of code section, we find the entry of page table in system EPT and make it invalid. For SS data section, we do the same setting.

5.2 Entry Gate and Exit Gate Implementation

We use entry and exit gates to switch the address spaces. First, in order to implement the switching between system EPT and SS EPT, we need to define another hypercall named “ept_switch”. The main task of entry and exit gates is issuing this hypercall to switch EPTs. This hypercall is designed to take an integer as parameter to indicate switching direction. If we want to enter SS to execute code, the physical address of the top level page table of SS EPT will be written to a control field in virtual machine control structure. If we want to exit SS and return to the kernel, the physical address of the top level page table of system EPT will be written to that field. Both the above physical addresses are stored in a structure of the hypervisor, so they cannot be obtained and operated by attackers.

In order to prevent the execution from diverting to somewhere else due to interrupt, we disable interrupts at the beginning of entry gates and enable interrupts at the end of exit gates. Moreover, we also need switch the stacks in the entry and exit gates. In other words, the security server address space has its own stack. The entry gates perform the transfer out of the stack in the system address space into the stack in the security server address space. The exit gates do the opposite thing. In this way, SS stack is not viewed and accessed by the kernel code, and therefore our framework can defend attacks against stack data.

There is still one detail we need to deal with. After SELinux completes the initialization, the entry and exit gates should begin to work. But we don’t have the SS EPT created at the moment, so the entry and exit gates actually do nothing before the “ept_create” hypercall is issued. We define a global integer variable H in the kernel to illustrate this condition. Before SS EPT is constructed, we assign zero to H . The entry and exit gates will read this value and do nothing. After creating SS EPT, we assign one to H . The code in both gates will perceive the change and begin to work.

5.3 Loading a Module to Enforce the Settings

The last step is to load a kernel module to take the entire configuration into effect. The module does the following two things.

First it will issue the “ept_create” hypercall to notify the hypervisor to create SS EPT. The four parameters of this function can be obtained through simple calculation. In section 5.1, we define four indicators, which are saved in the System.map file. They record the start and end address of SS code and data. The addresses are all virtual addresses. But the kernel lies in the direct mapping area, so we can get physical address by subtracting a fixed offset from virtual address. Then we can get guest frame numbers that the hypercall needs by shifting right the physical address a page size. When this hypercall is called successfully, SS EPT is created. Second, the module will set the global variable H , which is described in 5.2, to be one. The entry and exit gates check this variable and start to work.

Since the global variable H is in the system address space, attackers may change its value to be zero at run time. However, if it is set to be zero, the entry gates and exit gates will do nothing, and the guest VM can only use system EPT. In this way OM cannot find SS when it calls SS functions, as the system EPT has no mapping to them. So it returns a segmentation fault and terminates current process.

6 Evaluation

We have implemented our prototype on a DELL desktop with Intel Core i5-650 processor, which supports Intel VT technology. We used Xen 4.0.1 as the hypervisor. The host OS is 32-bit Ubuntu 10.04. So is the guest VM. It's booted up as HVM DomU with one VCPU and 512M RAM. We evaluate our work from the aspects of performance and security.

6.1 Performance Overhead

As many security hooks exist in the kernel and they are frequently called, we should pay attention to the performance overhead. In our experiments, we used UnixBench [11] to evaluate the system performance. UnixBench provides a basic reference of the performance of a Unix-like system. We tested the performance of the guest VM in two scenarios: one was with SELinux enabled natively and the other one was our prototype that used hardware assisted in-VM isolation.

First we ran UnixBench in DomU with SELinux enabled natively. It means SS and OM are not isolated from each other and both of them are in the kernel address space. We used the results as the base of our test. Then we ran UnixBench again using our prototype. SS is isolated from OM and it is in its own address space. As is shown in Figure 4, we divided the latter results by the formal ones and expressed the final results as percentages. Each result is the average of 5 trials. Most of them demonstrate that the overhead of our prototype is small except for the shell script test. This test made a process start eight concurrent copies of a shell script which did operations to a data file. SELinux would check process creation, shell script running and file access for each copy, so the performance went down due to frequent switching of EPTs.

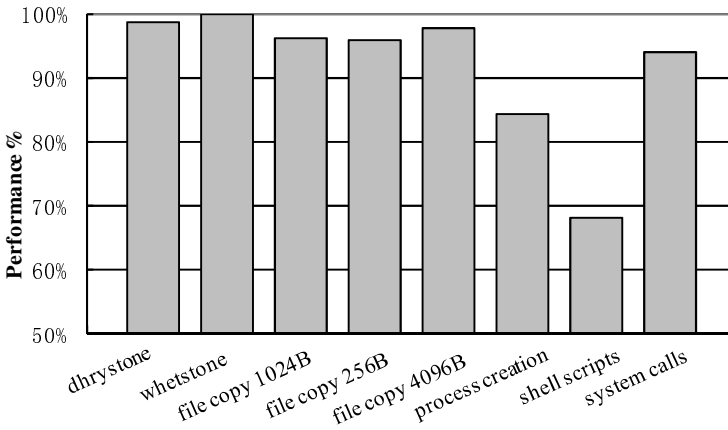


Fig. 4. Performance Overhead

Besides, as we put AVC in the security server address space to improve security, every time OM queries AVC for the access rules, it would switch to the security server space. This is a large part of performance overhead. We can improve the performance by moving AVC out of the security server address space but we need extra effort to protect it from being attacked.

6.2 Security Analysis

One important design consideration is to stop attacks against SS code and data. It is easy to protect SS code. For example, we can use write-protected technology or attestation technology to protect SS code. However, it is nontrivial to protect SS data. It includes dynamic data that may cause control flow attack. For instance, the heap overflow can be used to modify function pointers and stack buffer overflow can be used to change return addresses. Our approach can prevent such attack, as SS has its own address space, including its own heap and stack. Attackers may also launch non-control-data attack [22] to bypass the authentication. Since we isolate SS strongly from other parts of the kernel, the attacker who even obtains the kernel privilege cannot see SS non-control data. So our approach can prevent such attack too. As we put AVC and security policy data in the security server address space, it is also very difficult for the attacker to change them maliciously.

Since SS can load policy files into memory at runtime, attackers may breach the security of SS by loading an invalid policy. To prevent this we can maintain a white list which stores valid hash values of policy files. When SS loads a policy file, we can check whether the hash value of this file is in the list. If it is, the file is allowed to be loaded into the memory. Otherwise, SS refuses to load the policy.

7 Related Work

Our work is related to projects that use secure monitors to protect the integrity of OS kernels. Livewire [4] is a VMM-based prototype that aims at protecting the guest VM kernel code and critical data structures from being modified. VMwatcher [5] is a VMM-based malware detection approach, which moves the anti-malware facilities out of the monitored VM in order to achieve stronger tamper-resistance. Lares [6] is an architecture that places hooks inside the guest VM that can invoke a security application residing in a separated VM. Patagonix [7] is a hypervisor-based prototype that detects and identifies covertly executing binaries without making assumptions about the OS kernel. All these approaches are out-of-VM monitors which are inappropriate for Flask as it is a fine-grained secure architecture that the hooks are called frequently during system execution. They would incur heavy overhead due to frequent switching between the guest VMs and the hypervisor.

SIM [8] is an in-VM monitor which places the security tools in the same VM and uses the hardware virtualization technology to avoid the hypervisor involvement when switching address spaces. It is more efficient than the out- of-VM approaches, and also guarantees the security. However, the shadow paging mechanism it uses is

quite complex and not easy to maintain. So we design a hardware assisted in-VM isolated framework that takes advantage of Intel EPT technology to improve the Flask implementation.

8 Conclusions and Future Work

In this work, we improve the Flask implementation using hardware assisted in-VM isolation. We introduce a separated EPT in the hypervisor to construct an address space for SS so that we can isolate SS from other parts of the kernel. Since all the improving is done at the hypervisor level, the attacker cannot subvert them at the guest VM level. Our approach only works with a Type I VMM, which is implemented directly on the physical hardware, since we need the lower level VMM to protect guest VMs running on it. Our work is a first step towards creating in-VM isolation using hardware assisted page tables to improve Flask implementation. It can be applied to other security tools, such as integrity measurement agent, instruction detection tools and so on.

Our future work is to protect OM, especially the hooks used by OM to call AVC and SS functions. If they are bypassed, SS will have no effect at all. Only using write-protect method to protect them is not enough and we will utilize the research results of HookSafe [12] to protect the hooks from being hijacked.

Acknowledgments. Our work is supported by the National Natural Science Foundation of China under Grant No. 90818012, National Science and Technology Major Project No.2010ZX01036-001-002, 2010ZX01037-001-002, and the Knowledge Innovation Key Directional Program of Chinese Academy of Sciences under Grant No.KGCX2-YW-174, No.KGCX2-YW-125.

References

1. Spencer, R., Smalley, S., Loscocco, P., Hibler, M., Andersen, D., Lepreau, J.: The Flask security architecture: system support for diverse security policies. In: Proceedings of the 8th USENIX Security Symposium, Washington, DC, pp. 123–139 (1999)
2. Loscocco, P., Smalley, S.: Integrating flexible support for security policies into the linux operating system. In: Proceedings of the FREENIX Track: 2001 USENIX Annual Technical Conference (2001)
3. Vance, C., Watson, R.: Security Enhanced BSD. Network Associates Laboratories (2003)
4. Garfinkel, T., Rosenblum, M.: A virtual machine introspection based architecture for intrusion detection. In: Proceedings of the 10th Annual Network and Distributed Systems Security Symposium (2003)
5. Jiang, X., Wang, X., Xu, D.: Stealthy malware detection through vmm-based “out-of-the-box” semantic view reconstruction. In: Proceedings of the 14th ACM Conference on Computer and Communications Security, pp. 128–138 (2007)
6. Payne, B.D., Carbone, M., Sharif, M., Lee, W.: Lares: An architecture for secure active monitoring using virtualization. In: Proceedings of the 29th IEEE Symposium on Security and Privacy, pp. 233–247 (2008)

7. Litty, L., Lagar-Cavilla, H.A., Lie, D.: Hypervisor support for identifying covertly executing binaries. In: Proceedings of the 17th USENIX Security Symposium, pp. 243–258 (2008)
8. Sharif, M., Lee, W., Cui, W., Lanzi, A.: Secure in-VM monitoring using hardware virtualization. In: Proceedings of the 16th ACM Conference on Computer and Communications Security, pp. 477–487 (2009)
9. Intel Virtualization Technology, <http://www.intel.com/technology/virtualization>
10. Intel Inc. Intel 64 and IA-32 Architecture Software Developer's Manual Volume 3B: System Programming Guide, Part 1 (2006)
11. Tux.Org (1996), <http://www.tux.org/pub/tux/benchmarks/System/unixbench>
12. Wang, Z., Jiang, X., Cui, W., Ning, P.: Countering kernel rootkits with lightweight hook protection. In: Proceedings of the 16th ACM Conference on Computer and Communications Security, pp. 545–554 (2009)
13. The Blue Pill Project, <http://bluepillproject.org/>
14. King, S.T., Chen, P.M., Wang, Y.-M., Verbowski, C., Wang, H.J., Lorch, J.R.: SubVirt: implementing malware with virtual machines. In: Proceedings of the 2006 IEEE Symposium on Security and Privacy (2006)
15. Wojtczuk, R., Rutkowska, J.: Xen Owing trilogy. In: Black Hat Conference (2008)
16. Wang, J., Stavrou, A., Ghosh, A.K.: HyperCheck: A hardware-assisted integrity monitor. In: Proceedings of the 13th International Symposium on Recent Advances in Intrusion Detection (2010)
17. Azab, A.M., Ning, P., Wang, Z., Jiang, X., Zhang, X., Skalsky, N.C.: HyperSentry: enabling stealthy in-context measurement of hypervisor integrity. In: Proceedings of the 17th ACM Conference on Computer and Communications Security, pp. 38–49 (2010)
18. Wang, Z., Jiang, X.: HyperSafe, a lightweight approach to provide lifetime hypervisor control-flow integrity. In: Proceedings of the IEEE Symposium on Security and Privacy (2010)
19. Wang, X., Zang, J., Wang, Z., Luo, Y., Li, X.: Selective hardware/software memory virtualization. In: Proceedings of the 7th ACM Conference on Virtual Execution Environments, pp. 217–226 (2011)
20. Devine, S., Bugnion, E., Rosenblum, M.: Virtualization system including a virtual machine monitor for a computer with a segmented architecture. US Patent, 6397242 (1998)
21. Intel Inc. Intel 64 and IA-32 Architecture Software Developer's Manual Volume 3B: System Programming Guide, Part 2 (2007)
22. Shuo, C., Xu, J., Sezer, E.C., Gauriar, P., Iyer, R.K.: Non-control-data attacks are realistic threats. In: Proceedings of the 14th conference on USENIX Security Symposium (2005)
23. Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., Warfield, A.: Xen and the art of virtualization. In: Proceedings of the Symposium on Operating System Principles (2003)
24. Advanced Micro Devices. AMD64 Architecture Programmer's Manual Volume 2: System Programming, 3.12 edn. (2006)

HyperForce: Hypervisor-enFORced Execution of Security-Critical Code

Francesco Gadaleta, Nick Nikiforakis, Jan Tobias Mühlberg,
and Wouter Joosen

IBBT-DistriNet, KU Leuven, Celestijnenlaan 200A B-3001, Leuven, Belgium
{francesco.gadaleta,nick.nikiforakis,jantobias.muehlberg,
wouter.joosen}@cs.kuleuven.be

Abstract. The sustained popularity of the cloud and cloud-related services accelerate the evolution of virtualization-enabling technologies. Modern off-the-shelf computers are already equipped with specialized hardware that enables a hypervisor to manage the simultaneous execution of multiple operating systems. Researchers have proposed security mechanisms that operate within such a hypervisor to protect the *virtualized* operating systems from attacks. These mechanisms improve in security over previous techniques since the defense system is no longer part of an operating system’s attack surface. However, due to constant transitions between the hypervisor and the operating systems, these countermeasures typically incur a significant performance overhead.

In this paper we present HyperForce, a framework which allows the deployment of security-critical code in a way that significantly outperforms previous *in-hypervisor* systems while maintaining similar guarantees with respect to security and integrity. HyperForce is a hybrid system which combines the performance of an *in-guest* security mechanism with the security of in-hypervisor one. We evaluate our framework by using it to re-implement an invariance-based rootkit detection system and show the performance benefits of a HyperForce-utilizing countermeasure.

Keywords: virtualization, hypervisor, virtual devices, countermeasure.

1 Introduction

The “cloud” is probably the most used technological term of the last years. Its supporters present it as a complete change in the way that companies operate that will help them scale on-demand without the hardware-shackles of the past. CPU-time, hard-disk space, bandwidth and complete virtual infrastructure can be bought at a moment’s notice. Backups of data are synced to the cloud and in some extreme cases, all of a user’s data may reside there (Chromium OS). Its opponents treat it as privacy nightmare that will take away the user’s control over their own data and place it in the hands of corporations as well as a risk to the privacy, integrity and availability of user data [3,13]. Regardless however

on one's view of the cloud, one of the main technologies that makes the cloud-concept possible is virtualization.

Virtualization is the set of technologies that together allow for the existence of more than one running operating systems on-top of a single physical machine. While initially all of the needed mechanisms for virtualization were created in software, the sustained popularity of virtualization, lead to their implementation in hardware, providing the desired speed that was lacking in their software counterparts. Today both Intel¹ and AMD² support a set of instructions that are there with the sole purpose of facilitating virtualization. Apart from the use of virtualization as way to host different operating systems on one machine, virtualization can also be used to provide greater security guarantees for operating systems. Researchers have already proposed various system that use virtualization primitives that all fall in this category [5,6,7,8,15]. The chief difference between these systems that operate from within the virtualized system's hypervisor (*in-hypervisor*) and protection systems that operate from within the operating system (*in-guest*) is that the latter are part of the system's attack surface. For instance, an antivirus that operates from within the operating system that it supposedly protects, i.e. in-guest, could be deactivated or crippled by the attack itself. In-hypervisor security systems, in contrast, can utilize the isolation guarantees of virtualization to make sure that they will be active regardless of the state of system that they protect. Unfortunately these security benefits do not come for free. The constant transition from the virtualized operating system to the hypervisor that protects it (known as VMExit) and back (VMEntry), negatively affects the performance of the virtualized systems forcing one to choose between better security or better performance.

In this paper we present HyperForce, a framework that allows countermeasures for virtualized operating systems to be protected, with security and integrity comparable to the one provided by in-hypervisor systems but at the performance cost of in-guest systems. Our system follows a hybrid approach by maintaining the security-critical code within the guest but forcing its execution and protecting its instructions and data from the hypervisor. Using our framework, we re-implement Hello rootKitty [7], an in-hypervisor rootkit-detection system which uses the invariance of critical kernel objects as a way of identifying kernel compromises. We evaluate the implementation of Hello rootKitty using our framework and show that it significantly outperforms the original version while maintaining comparable security guarantees.

The rest of this paper is structured as follows. In Section 2 we present our motivation for the HyperForce framework followed by its design details. In Section 3 we evaluate the performance benefits of our framework by using it to re-implement the aforementioned rootkit-detection system. In Section 4 we discuss the related work and Section 5 concludes.

¹ <http://www.intel.com/technology/virtualization/technology.htm>

² <http://sites.amd.com/us/business/it-solutions/virtualization>

2 Design

In this section, we describe the needs that motivated us to create HyperForce and we provide the design and implementation details of our system.

2.1 Motivation

Designing a countermeasure that protects virtualized operating systems is considered a challenge not only because of the difficulty to modify the target system (due to the lack of sources or licenses) but also because a virtualized system is already affected by consistent overhead, by design. An important goal for any framework using virtualization as a security tool, is to guarantee the execution of critical code in the kernel-space of a virtualized operating system regardless of the state of the kernel, i.e. code that will run identically in both clean and compromised kernels. By critical code we refer to code that, in general, monitors the state of the system and that it is desirable, mainly from a security point of view, to maintain its execution. Examples of such code include the integrity check of sensitive kernel-level data structures that are usually abused by rootkits or the scanning of files and memory for known malware signatures. Given our assumption of a kernel-level attacker, it is also needed to ensure the integrity of the critical code to protect it from malicious modifications which might compromise its efficacy or completely disable its operations.

A straightforward way of achieving this goal is to implement and execute security-critical code within the hypervisor [7]. An alternative approach monitors the target system from a separate virtual machine. In fact, one of the most interesting features of virtualization technology is that it guarantees complete isolation between the hypervisor and any virtual machine running on top of it as well as isolation between multiple virtual machines running on top of the same physical machine. Unfortunately, both approaches are known to be affected by consistent performance overhead, making it hard to consider such solutions for production systems. The main goal of HyperForce is to keep a degree of security comparable to these completely isolated systems while significantly reducing their performance overhead.

2.2 Core Idea

The idea of HyperForce is to combine the best features of the *in-guest* and *in-hypervisor* defense systems into a hybrid solution which performs as an in-guest countermeasure while providing security comparable to in-hypervisor countermeasures. We achieve this by deploying the functional part of the countermeasure within the guest operating system while maintaining its integrity and enforcing its execution with the assistance of the hypervisor. Since the functional part of the security-critical code, i.e. its instructions and data, is running within the virtualized operating system, it also has native access to the resources of the virtualized operating system such as the memory, disk and API of the virtualized kernel. This provides a great performance benefit for code that

needs to access many memory locations within the virtualized operating system since it alleviates the costly need of introspection that in-hypervisor systems require, i.e. the discovery of the corresponding physical memory pages of the virtual memory pages of the guest and their remapping within the hypervisor or within another virtual machine.

Enforcement of Execution. Given an arbitrary piece of security-critical code, HyperForce needs to ensure its execution at regular time intervals. A complete reliance for its execution on the virtualized operating system, could potentially allow a kernel-level attacker to intervene and inhibit the code's execution through the modification of the appropriate kernel-level data-structures. For instance, an attacker could locate the function pointer pointing to the security-critical code and overwrite it with a pointer towards their own code.

From a high-level view, HyperForce changes the execution flow of the guest kernel whenever the installed monitoring code has to be executed and restores the original execution flow upon code termination. The advances of virtualization technology allows one to implement this transition in a multitude of ways. Our decision was influenced by our desire of minimizing the amount of instrumentation code in the hypervisor and of keeping performance overhead to a minimum.

In our framework, the security-critical code is encapsulated within a function that is loaded in the virtualized operating system in the form of Linux Kernel Module (LKM). This allows the code to have native access to all of the VM's native resources. HyperForce then uses the infrastructure of the virtualization platform, specifically the Virtual Machine Monitor (VMM), to create a virtual device. Virtual devices simulate real hardware devices, such as sound-cards and video cards, and are supported by all modern Virtual Machine Monitors. Once this virtual device is created and loaded in the virtualized operating system, HyperForce then registers the address of the security-critical code as an interrupt handler for the virtual device, as illustrated in Figure 11. The cooperation of the hypervisor and the trusted module, allows for the security-critical code to execute every time that the virtual device generates an interrupt.

Since the virtual device is fully controlled by the hypervisor, it is the hypervisor that decides when interrupts must be generated and not the virtualized operating system. Due to this fact, the possibly compromised kernel of the VM, cannot anticipate when the security-critical code will be executed since the logic behind it is hidden from it through the virtualization-guaranteed isolation between hypervisor and VM. This fact stops any attackers' efforts to evade detection by mimicking a non-compromised operating system just before the execution of the critical-code and restoring their malicious activities after it.

Integrity of Code. In the previous paragraphs we described the loading of security-critical code within the hypervisor and the use of virtual devices to ensure the execution of that code. Since the code is loaded in the VM as a LKM, it executes with the privileges of the virtualized kernel. While this is

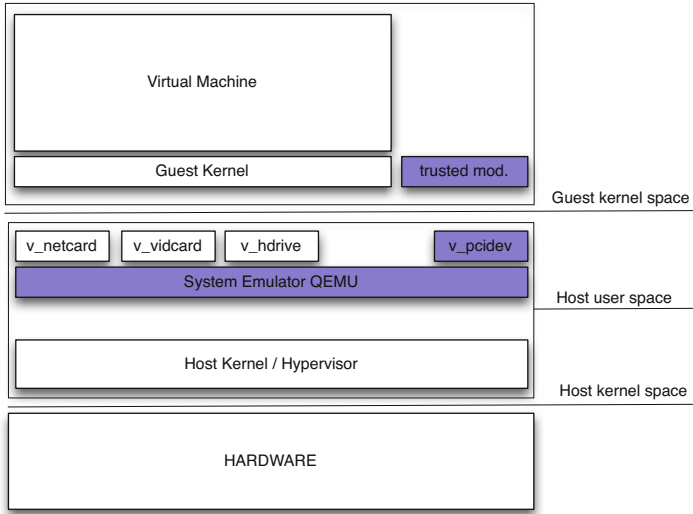


Fig. 1. Schema of HyperForce. Highlighted components indicate parts of the system that need instrumentation/modification.

desired, it also opens up the code to attacks, e.g. modifications of its code and data, from an attacker who is in control of the virtualized kernel. Traditionally, the module could not be protected from the rest of the kernel since they both operate within the same protection ring, namely Ring 0. Due to virtualization however, the hypervisor has more power than the virtualized operating system's kernel (signified as Ring -1) and can thus protect any resources from the virtualized kernel, including memory pages. HyperForce takes advantage of this fact, and write-protects the memory pages holding the instructions and data of the security-critical code. In order to allow the code to make changes to its data, HyperForce can unlock the memory pages before it triggers an interrupt of its virtual device and lock them back immediately after the code's execution. In order to ensure that an attacker cannot avoid the execution of the interrupt handler containing the security-critical code, HyperForce also write-protects the memory page holding the Interrupt Descriptor Table (IDT) of the protected VM. Lastly, HyperForce protects the Interrupt Descriptor Table Register (IDTR) that contains the address of the IDT, as a regular invariant critical kernel object.

3 Evaluation

We implemented HyperForce in KVM, an extension of the Linux kernel with hypervisor capabilities. KVM is formed by a system emulator, QEMU, that runs as regular process in user space and a kernel-space device driver that uses virtualization-enabled processors. In order to show the improvements provided

by our framework we chose to re-implement and measure a pure *in-hypervisor* countermeasure, namely *Hello rootKitty* [7].

Hello rootKitty is a lightweight invariance-enforcing framework that mitigates the problem of kernel-level rootkits. It represents a typical in-hypervisor monitoring system that checks the integrity of invariant guest-kernel objects from the hypervisor. A periodical mapping of guest-kernel memory into hypervisor space is followed by computation of its hash and checks against a set of precomputed values. Such a countermeasure often deals with a high number of kernel objects and performance overhead can easily make the guest system unusable. To minimize the amount of time spent by additional code, only a subset of these objects is checked whenever control returns to hypervisor (VMExit). Thus a certain number of VMExit events is needed to check the entire list of protected objects. This relaxation will have a cost in terms of detection time needed to check the entire list of objects.

The original version of *Hello rootKitty* was implemented in BitVisor [19], a tiny hypervisor designed for mediating I/O access from a single guest operating system. In order to be able to fairly compare it with our implemented version using HyperForce, we also re-implemented the original *Hello rootKitty* in KVM. A schema of *Hello rootKitty* implemented in KVM is provided in Fig. 2. It can be observed that while the HyperForce framework requires only the system emulator to be modified (Fig. 1), the implementation of the in-hypervisor *Hello rootKitty* needs instrumentation code to be added to the host kernel. In both cases the trusted module needs to be added to the guest kernel.

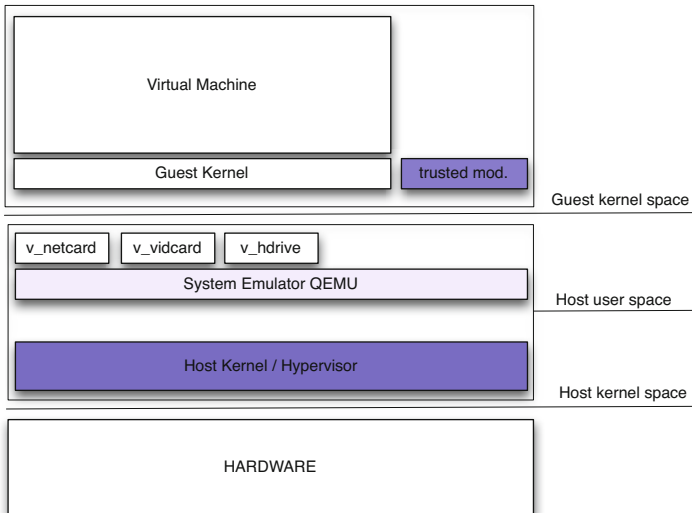


Fig. 2. Schema of *Hello rootKitty* implemented in Linux KVM. Highlighted components indicate parts of the system that need instrumentation/modification.

Table 1. Macro benchmarks (*in-host* OS and *in-guest* OS) evaluating *Hello rootKitty* implemented with and without HyperForce

(a) In-host measurements			(b) In-guest measurements		
	iperf [Gb/s]	overhead		iperf [Gb/s]	bunzip [sec]
HRK	6.36	-	native KVM	5.97	32.04
HF(HRK)	6.29	+1.1%	HRK	5.26 (+12%)	33.73 (+5%)
			HF(HRK)	5.71 (+4.3%)	32.88 (+2.5%)

We collected results of macro and micro-benchmarks from the guest and from the host machine and discuss them in Section 3.1 and Section 3.2. In order to provide reliable results, all tests have been repeated 10 times and averaged. Experiments have been performed on Intel Core 2 Duo 2 Ghz processor with 4GB of RAM.

3.1 Macro-benchmarks

We run two macro benchmarks, *iperf* that measures TCP and UDP bandwidth performance and *bunzip* of a Linux kernel source code. The original in-hypervisor version of *Hello rootKitty* is denoted as “HRK” while the version using the HyperForce framework is denoted as “HF(HRK)”.

While the in-hypervisor approach, due to the slower context switching, has a slightly better throughput of network performance in the host machine Table 1(a), benchmarks in the guest machine show a considerably better performance with HyperForce. *iperf* and *bunzip* have also been executed on a native KVM system and compared against the same system running in-hypervisor *Hello rootKitty* and then HF(HRK). The performance overhead of our approach is about half of the in-hypervisor *Hello rootKitty*, as shown in Table 1(b). *Hello rootKitty* implemented using HyperForce performs with 4.3% overhead compared to a native KVM guest while in-hypervisor *Hello rootKitty* shows 12% overhead. The second column reports overhead of *bunzip* measured in seconds. HF(HRK) outperforms the in-hypervisor *Hello rootKitty*, showing an overhead of only 2.5% compared to the native KVM guest.

3.2 Micro-benchmarks

Micro benchmarks show a more detailed picture of the two approaches. We use *LMbench* [12]³ to measure the overhead of operating system specific events such as context switch, memory mapping latency, page fault, signal handling and fork. Within the host machine, HyperForce shows substantial improvement against the alternative *Hello rootKitty*. In Table 2 we report only the tests where this improvement is consistent. In all other tests the in-hypervisor *Hello rootKitty* and the *Hello rootKitty* using HyperForce show negligible performance overhead.

³ We use version 3 of LMbench as available at <http://lmbench.sourceforge.net/>

Table 2. Overhead of *Hello rootKitty* using the HyperForce framework (HF(HRK)) is measured against in-hypervisor *Hello rootKitty* (HRK) with LMBench micro-benchmarks within the host machine. Operations are measured in microseconds.

	ctx switch	mmap lat	page flt	mem lat
HRK	2.020	6148	1.57	114.7
HF(HRK)	1.48	4950	1.46	101.7
speedup	+26%	+19%	+7%	+11%

Table 3. Overhead of HF(HRK) is measured against in-hypervisor *Hello rootKitty* with LMBench micro-benchmarks within the guest machine. Operations are measured in microseconds.

	null call	null IO	open/close	sig inst	sig handl	fork proc	exec proc	ctx switch
HRK	0.30	0.32	2.32	0.74	5.37	1923	4087	5.58
HF(HRK)	0.14	0.21	2.10	0.45	2.60	1788	3984	5.00
Speedup	+53%	+34%	+10%	+39%	+51%	+8%	+2.5%	+10%

The picture in the guest machine shows a similar trend in which HF(HRK) outperforms the original in-hypervisor *Hello rootKitty* in every test (Table 3).

To interpret the results shown in Table 3, one has to know that *Hello rootKitty* performs integrity checks whenever the guest kernel writes to a control register (MOV_CR* event). When virtual addressing is enabled, the upper 20 bits of control register 3 (CR3) become the page directory base register, which is used to locate the page directory and page table of the current process. Thus, on every context switch or system call invocation, CR3 is modified. Trapping these events strategically contributes to *Hello rootKitty*’s short attack detection time. Yet, the integrity checks performed by *Hello rootKitty* increase the latency of context switches and system calls.

In contrast, our implementation of *Hello rootKitty* in HyperForce employs interrupt events to trigger in-guest integrity checks. This eliminates overheads with respect to switching execution context and address mapping between the hypervisor and the guest OS, while the remaining computational overhead affects guest operations more evenly. As can be seen in Table 3, the above changes imply significant speedups on system call invocations (53%) and context switches (10%). Although LMBench is often considered as insufficient for evaluating system performance [10], our example shows that the benchmark suite can be used to neatly distinguish the actual speedup on system call invocations (“null call”) from the impact on a particular system call execution (e.g. “open/close”).

One may think that our approach to trigger security checks through interrupts in HyperForce reduces the security of the protected system compared to the original in-hypervisor *Hello rootKitty*: in the latter case an attacker increases their chance of being detected with every system call raised. However for a total

of 15,000 protected kernel objects, the worst-case detection time reported in [7] is 6 seconds. *Hello rootKitty* in HyperForce improves on that by checking the same amount of kernel objects in 4 seconds. While *Hello rootKitty* relies on the activity of the system as a trigger that checks the integrity of protected objects, *Hello rootKitty* in HyperForce performs the checking independently of system activity every 4 seconds.

In summary, our implementation of *Hello rootKitty* in HyperForce significantly reduces computational overhead while reducing the worst-case detection time for potentially malicious manipulations of invariant kernel objects. Our results indicate that the HyperForce framework could be used to re-implement other in-hypervisor applications, enhancing their performance and maintaining their effectiveness.

4 Related Work

In this section we review related work in the domain of kernel code integrity assurance. For a discussion of literature related to rootkit detection we refer the reader to [7].

Hardware-Based Execution Flow Integrity. Means of guaranteeing the integrity of a running operating system that employ dedicated hardware devices to monitor the physical memory of a computer system have been proposed in [1] and [14]. In order to perform integrity checks, both systems make use of PCI hardware that directly accesses the computer’s memory at a negligible performance overhead. Yet, the need for dedicated hardware may hinder widespread deployment of these techniques.

Hypervisor-Based Execution Flow Integrity. A tiny hypervisor that protects legacy OSs by ensuring that only validated code can be executed in kernel mode, is SecVisor [17]. A similar system, NICKLE [16], shadows physical memory to store authenticated guest code. At runtime, an instruction fetch is directed to access either the normal system memory or the shadow area, depending on whether the instruction is to be executed in user mode or kernel mode. An attempt to execute unvalidated code can thus be detected and prevented. Recently, attacks that do not inject malicious code but construct it from existing fragments of the attacked program have been presented [2,9,18]. These attacks effectively bypass countermeasures such as SecVisor and NICKLE.

Rootkits commonly modify a system’s function pointers to ensure execution. HookScout [21] detects such rootkits. The tool employs a system emulator to infer a policy for function pointer propagation in kernel memory. A separate detection system is then used to detect violations of this policy during normal operation of the OS. Since the detection system runs on the target machine, it may be disabled by an attack. HookSafe [20] protects kernel hooks that are dynamically allocated by relocating these kernel hooks to dedicated memory pages. Regular page-level protection through the hardware’s Memory

Management Unit is then used to protect the pages. Yet, the technique does not prevent non-control data from being compromised.

HyperForce can be utilized to effectively protect the integrity of the in-guest components of systems such as HookScout and HookSafe. Our experimental results obtained from implementing *Hello rootKitty* [7] in HyperForce show that our technique leverages the use of in-guest protection mechanisms. That is, HyperForce substantially reduces the performance overhead that would occur if the countermeasure would be implemented in-hypervisor, while strong security guarantees are maintained.

Security Agent Injection. Closely related to HyperForce is work by Lee et al. [11] and Chiueh et al. [4] on deploying agents by means of code injection from a hypervisor. Both approaches are applicable to guest OSs that have not been previously prepared by loading a special driver or similar. In [11], Lee et al. proposes to protect agent code that is executing in a compromised guest OS kernel by the use of cryptography and by injecting this code on demand from the hypervisor. As there is no implementation and no experimental evaluation given, a comparison with HyperForce is not feasible. Similarly, work on SADE [4] by Chiueh et al. uses VMWare’s ESX server API to inject and execute code in a guest OS so as to disable and remove a previously detected malware infection from that guest. In difference to the HyperForce approach, the agent code in SADE is not protected from malicious interference on the guest. Chiueh et al. argue that on-demand injection leaves a relatively short time span for such interference. SADE is used by a virtual appliance that implements out-of-guest monitoring of VMs’ memory, scanning for malware signatures. The paper presents experimental data on the code injection process but does not discuss the overhead implied by mapping memory pages between the virtual appliance and the VMs. We expect in-guest memory inspection, as implemented by our *Hello rootKitty* in HyperForce, to outperform SADE.

5 Conclusion

The attractive properties offered by virtualization are a foundational block for the whole “cloud technology”. At the same time, virtualization is already being used for purposes other than the deployment of multiple operating systems as a way of increasing the security of a single virtualized operating system. In this paper we briefly discuss the differences between security mechanisms deployed within an operating system (*in-guest*) and the ones deployed within a hypervisor (*in-hypervisor*) and bring attention to the, seemingly exclusive, choice between the performance benefits of the former versus the security benefits of the latter. We tackle this choice by developing HyperForce, a hybrid framework allowing security mechanisms to be developed in a way that provides them with performance analogous to in-guest systems while maintaining the security of in-hypervisor systems. Using HyperForce, we re-implemented an in-hypervisor rootkit detection system and show how the new version significantly outperforms

the original without compromising the security or integrity of the detection system.

We conclude that hybrid security systems that are built on top of HyperForce can provide effective and efficient alternatives to mitigate the overhead of techniques that exclusively operate in-hypervisor. Interesting candidate applications for our framework are, e.g., malware detection and removal software. For future work we envisage to extend HyperForce with techniques to inject security agents into a guest operating system so as to provide secure means of on-demand deployment of such agents.

Acknowledgments. This research is partially funded by the Interuniversity Attraction Poles Programme Belgian State, Belgian Science Policy, the Research Fund KU Leuven and the EU FP7 project NESSoS.

References

1. Baliga, A., Ganapathy, V., Iftode, L.: Detecting kernel-level rootkits using data structure invariants. *IEEE Transactions on Dependable and Secure Computing* 8, 670–684 (2011)
2. Buchanan, E., Roemer, R., Shacham, H., Savage, S.: When good instructions go bad: generalizing return-oriented programming to RISC. In: *CCS 2008: Proceedings of the 15th ACM Conference on Computer and Communications Security*, pp. 27–38. ACM (2008)
3. Business Insider. Amazon’s Cloud Crash Disaster Permanently Destroyed Many Customers’ Data (2011), http://articles.businessinsider.com/2011-04-28/tech/29958976_1_amazon-customer-customers-data-data-loss
4. Chiueh, T.C., Conover, M., Lu, M., Montague, B.: Stealthy deployment and execution of in-guest kernel agents. In: *Proceedings of the Black Hat USA Security Conference* (2009)
5. Criswell, J., Lenharth, A., Dhurjati, D., Adve, V.: Secure Virtual Architecture: A Safe Execution Environment for Commodity Operating Systems. In: *SOSP 2007: Proceedings of the 21st ACM Symposium on Operating Systems Principles*, pp. 351–366. ACM (2007)
6. Dewan, P., Durham, D., Khosravi, H., Long, M., Nagabhushan, G.: A hypervisor-based system for protecting software runtime memory and persistent storage. In: *SpringSim 2008: Proceedings of the 2008 Spring Simulation Multiconference*, pp. 828–835. Society for Computer Simulation International (2008)
7. Gadaleta, F., Nikiforakis, N., Younan, Y., Joosen, W.: Hello rootKitty: A Lightweight Invariance-Enforcing Framework. In: Lai, X., Zhou, J., Li, H. (eds.) *ISC 2011*. LNCS, vol. 7001, pp. 213–228. Springer, Heidelberg (2011)
8. Gadaleta, F., Younan, Y., Jacobs, B., Joosen, W., De Neve, E., Beosier, N.: Instruction-level countermeasures against stack-based buffer overflow attacks. In: *Eurosys*, pp. 7–12. ACM (2009)
9. Hund, R., Holz, T., Freiling, F.C.: Return-oriented rootkits: Bypassing kernel code integrity protection mechanisms. In: *SSYM 2009: Proceedings of the 18th Conference on USENIX Security Symposium*, pp. 383–398. USENIX Association (2009)

10. Open Kernel labs. Why lmbench is evil?,
<http://www.ok-labs.com/blog/entry/why-lmbench-is-evil/>
11. Lee, Y.-C., Rahimi, S., Harvey, S.: A pre-kernel agent platform for security assurance. In: IEEE Symposium on Intelligent Agent (IA), pp. 1–7. IEEE (2011)
12. McVoy, L., Staelin, C.: LMBench: Portable tools for performance analysis. In: Proceedings of the 1996 Annual Conference on USENIX Annual Technical Conference, pp. 23–39. USENIX Association, Berkeley (1996)
13. Nikiforakis, N., Balduzzi, M., Van Acker, S., Joosen, W., Balzarotti, D.: Exposing the lack of privacy in file hosting services. In: Proceedings of the 4th USENIX conference on Large-Scale Exploits and Emergent Threats, LEET (2011)
14. Petroni Jr., N.L., Fraser, T., Molina, J., Arbaugh, W.A.: Copilot - a coprocessor-based kernel runtime integrity monitor. In: Proceedings of the 13th USENIX Security Symposium, p. 13. USENIX Association (2004)
15. QubesOS: Architecture Specification,
<http://qubes-os.org/files/doc/arch-spec-0.3.pdf>
16. Riley, R., Jiang, X., Xu, D.: Guest-Transparent Prevention of Kernel Rootkits with VMM-Based Memory Shadowing. In: Lippmann, R., Kirda, E., Trachtenberg, A. (eds.) RAID 2008. LNCS, vol. 5230, pp. 1–20. Springer, Heidelberg (2008)
17. Seshadri, A., Luk, M., Qu, N., Perrig, A.: SecVisor: a tiny hypervisor to provide lifetime kernel code integrity for commodity OSes. In: Proceedings of Twenty-First ACM SIGOPS Symposium on Operating Systems Principles, pp. 335–350. ACM (2007)
18. Shacham, H.: The geometry of innocent flesh on the bone: return-into-libc without function calls (on the x86). In: CCS 2007: Proceedings of the 14th ACM Conference on Computer and Communications Security, pp. 552–561. ACM (2007)
19. Shinagawa, T., Eiraku, H., Tanimoto, K., Omote, K., Hasegawa, S., Horie, T., Hirano, M., Kourai, K., Oyama, Y., Kawai, E., Kono, K., Chiba, S., Shinjo, Y., Kato, K.: BitVisor: a thin hypervisor for enforcing I/O device security. In: VEE 2009: Proceedings of the 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments, pp. 121–130. ACM (2009)
20. Wang, Z., Jiang, X., Cui, W., Ning, P.: Countering kernel rootkits with lightweight hook protection. In: CCS 2009: Proceedings of the 16th ACM Conference on Computer and Communications Security, pp. 545–554. ACM (2009)
21. Yin, H., Poesankam, P., Hanna, S., Song, D.: HookScout: Proactive Binary-Centric Hook Detection. In: Kreibich, C., Jahnke, M. (eds.) DIMVA 2010. LNCS, vol. 6201, pp. 1–20. Springer, Heidelberg (2010)

RandHyp: Preventing Attacks via Xen Hypercall Interface

Feifei Wang, Ping Chen, Bing Mao, and Li Xie

State Key Laboratory for Novel Software Technology, Nanjing University
Department of Computer Science and Technology, Nanjing University

Abstract. Virtualization plays a key role in constructing cloud environments and providing services. Although the main jobs of the hypervisors are to guarantee proper isolation between domains and provide them services, the hypercall interface provided by the hypervisor for cross-layer interactions with domains gives attackers the possibility to breach the isolation or cause denial of service from inside the domains. In this paper, we propose a transparent approach that uses randomization technique to protect the hypercall interface. In our approach, even facing a total compromise of a domain, the security of the virtualization platforms can be guaranteed. We have built a prototype called RandHyp based on Xen. Our experimental results show that RandHyp can effectively prevent attacks via Xen hypercall interface with a small overhead.

Keywords: Xen, Hypercall, Protection, Randomization.

1 Introduction

Nowadays, virtualization has gained an increasingly concern in both industry and academic world. Cloud hosting providers have exploited the abstraction and isolation provided by the hypervisor to allow individual paying customers to share large-scale datacenter facilities, such as Amazon EC2[2], and Linode[3]. A lot novel projects also take advantage of hypervisor's supervision property in intrusion detection systems[19], workload isolation[20], attack investigation and debugging[21], and system monitoring[4].

However, all these applications are based on the belief that the hypervisor is sufficiently trustworthy to provide services and maintain isolation. But is it really like that? While the hypervisor itself can be regarded as secure due to its small size and a well-defined narrow interface, a number of different commodity operating systems are coresident on the same host[14], numerous vulnerable applications communicate with the uncertain outside world[13]. Under these circumstances, it is very likely that by exploiting the vulnerabilities of applications and operating systems, attackers can compromise a domain in the same way as they do in conventional operating systems. After that, attackers can escalate privileges by applying to the hypervisor for illegitimate resources, which may lead to information leakage or denial of service. Specifically, if dom0 is compromised, attackers are able to access other domains' memory and I/O informations, and even create or shutdown other domains at will. In situations where

dom0 is simplified to reduce the vulnerabilities exposed [6], attackers can turn to compromise domUs. And according to CVE bug reports (e.g. CVE-2011-1898, CVE-2010-4238) [1], domU users can gain dom0 privileges and can either cause a denial of service or read arbitrary files in dom0. Further, any domain can be dominated to keep requesting critical resources, such as cpu and memory, which may lead to denial of services to other domains.

Since the hypercall interface is used to access hardware resources and execute sensitive instructions, the malicious goals listed above have to be achieved by invoking hypercalls. Therefore, preventing the hypercall interface from being used is very essential.

In this paper, we present solutions to protect domains by using Xen hypercall interface randomization technique. In our approach, all hypercall invocations are classified into trusted ones or untrusted ones. For the trusted ones, we randomize each hypercall's arguments, including its hypercall number. We also add paddings to the arguments and permute both the paddings and the arguments to further make our approach less breakable. The untrusted ones do not get randomized and are regarded as invalid by the hypervisor. In such a case, even attackers can get into a domain, he can not further elevate his privileges by exploiting the hypercall interface, and thus avoid more serious damage to the whole system.

We present RandHyp, a modified version of Xen with para-virtualized Linux platforms as dom0 and domUs. Our experimental results show that RandHyp can successfully prevent untrusted hypercalls from executing while incurring low performance penalty.

The rest of this paper is organized as follows. The next section presents a short related work section about existing security mechanisms in virtualization platforms. Section 3 identifies the threat model, explicits our design goals, and gives the design overview. After that, the implementation of RandHyp is described, following by a brief security analysis. The evaluation results are given in section 6, and section 7 concludes this paper.

2 Related Work

Existing security measures, like mandatory access control (MAC) [15] and trusted platform module (TPM) [16], can not be directly applied to protecting the Xen hypercall interface. Most previous efforts concentrate on reducing the probability of domains being compromised by monitoring domains from the hypervisor level [11, 12]. However, designers may not know all threats, and new exploitation techniques can appear, and attack vectors cannot be totally eradicated, thus domains are still untrustable.

Other solutions are based on the assumption that dom0 is malicious (dom0 is a high-valued attack target, resulting from its control over other domains), and aim at protecting domUs against a malicious dom0. For example, CloudVisor [8] introduces a tiny security monitor to protect resources, Xoar [9] breaks the control domain into single-purpose components to reduce the damage attackers

can introduce, Chunxiao Li *et al.* [10] removed the control domain from TCB, and so on.

However, all these solutions are difficult to implement in the commercial products for the reason that they need to modify the existing virtualization architectures, which requires professionals and may lead to problems in updating systems.

Hoang [13] has proposed two innovative approaches, authenticated hypercalls (MAC) and hypercall access table (HAT), that aim at protecting the hypercall interface. The MAC approach is not practical given the limitation of the number of arguments that can be passed to the hypervisor. The HAT approach records the addresses of trusted hypercall invocations and stores them in an HAT table in the hypervisor. However, in kernels of the same version, the addresses are fixed. Hence, getting the addresses of the trusted hypercall invocations are easy. Moreover, the size of the HAT table stored in the hypervisor will increase as the number of domains increases, which contradicts the design concept of Xen [5].

Unlike existing solutions, RandHyp maintains the design concept of Xen with a small modification of the Xen hypervisor and guest operating systems and is transparent to guest applications. Besides, since the changes of the operating systems are mainly in some head files, they would not hinder updating systems or inserting of modules.

3 Xen Architecture, Threat Model, and Design Goals

Since RandHyp is based on Xen, this section first describes the architecture of Xen, especially the hypercall interface. Then the threat model is identified, and our design goals are articulated.

3.1 Architecture Overview

In Xen architecture, the hypervisor is the most privileged software layer that virtualizes the hardware resources and partitions them dynamically to the overlying domains. A single administrative domain (dom0) has the ability to manage all other domains (domU).

Since the hypervisor is responsible for monitoring all privileged state, domains have to transfer control into the hypervisor when executing sensitive instructions, which is realized by using hypercalls [7]. Hypercalls are made to execute high-privileged operations, such as exception handling, scheduling, physical memory management, access-control of disks and network devices, virtual CPU operations, and inter-domain communications.

Hypercalls are similar to system calls in conventional operating systems. A software interrupt INT \$0x82 transfers the control from the domain into the hypervisor, where operations are validated and applied, and when completed, the control is returned to the calling domain [5]. The particular hypercall to be invoked is contained in `rax` with all the arguments contained in `rbx`, `rcx`, `rdx`, `rsi`, `rdi` (x86_64 Linux) or `eax`, `ebx`, `ecx`, `edx`, `esi`, `edi` (x86_32 Linux).

An example use of hypercall is to update an individual segment descriptor in the GDT or LDT by using hypercall `HYPERVISOR_update_descriptor(unsigned long ma, unsigned long word)`.

3.2 Threat Model

We assume the hardware and hypervisor are trusted, which means that we are not concerned with the violation of security by the service providers. We focus on the threat when a well-behaved domain is compromised and used as an entry to breach isolations among domains.

In a virtualization environment, since domains are usually commodity operating systems with vulnerabilities, it is prudent to assume that they may be compromised to behave with evil intentions. Thus, the attacker in our model is a domain aims at violating the security of other domains, including violating the data integrity or confidentiality of the target domains or causing denial of service to other domains with whom it is sharing the same underlying resources.

In order to achieve the malicious goals, the attacker inevitably has to go through the hypercall interface to achieve hardware resources. For example, domains cannot directly apply for extra memory space or modify critical data structures such as page tables, but these operations can be done through hypercall requests. In such a case, the attacker needs to invoke a series of dedicatedly arranged hypercalls to make his malicious goals realized.

Since hypercalls are much similar to system calls, hypercall attacks could be in any form known for system call attacks such as argument hijacking or mimicry [13]. Faking a series of hypercalls to sniff other domains' information or cause denial of service is applicable.

Information Leakage Attack. Information leakage attacks are mainly caused by a malicious dom0, for the reason that dom0 can access memory and I/O informations belonging to domUs. Given the evidence that information leakage may bring more profits to attackers, dom0 turns out to be a high-valued attack target.

Most projects assume a small-sized dom0 and deduce that the vulnerabilities exposed to be rare. However, this is undesirable as demonstrated by Colp [9] that dom0 in a mature virtualization platform is actually larger than a conventional server operating system for it is often relied on to provide additional shared services, such as drivers for physical devices, device emulation, and administrative tools.

The potentially malicious dom0 is often ignored in discussing the security of virtualization systems. Once dom0 is compromised, personal informations of customers can be stealed, which would be a total disaster.

Denial of Service Attack. Both dom0 and domUs can launch denial of service attacks through keeping occupying resources. Further, dom0 can cause a denial of service to certain domains by changing the scheduling flows.

3.3 Design Overview

Rather than exploring techniques to construct a bug-free system, a more pragmatic goal is to provide an architecture that can prevent the hypercall interface from being used, so that attackers in subverted domains cannot further mount successful attacks against other domains. With this in scheme, RandHyp is designed to guarantee that a malicious domain cannot issue arbitrary control transfers into the hypervisor and the execution context cannot be faked.

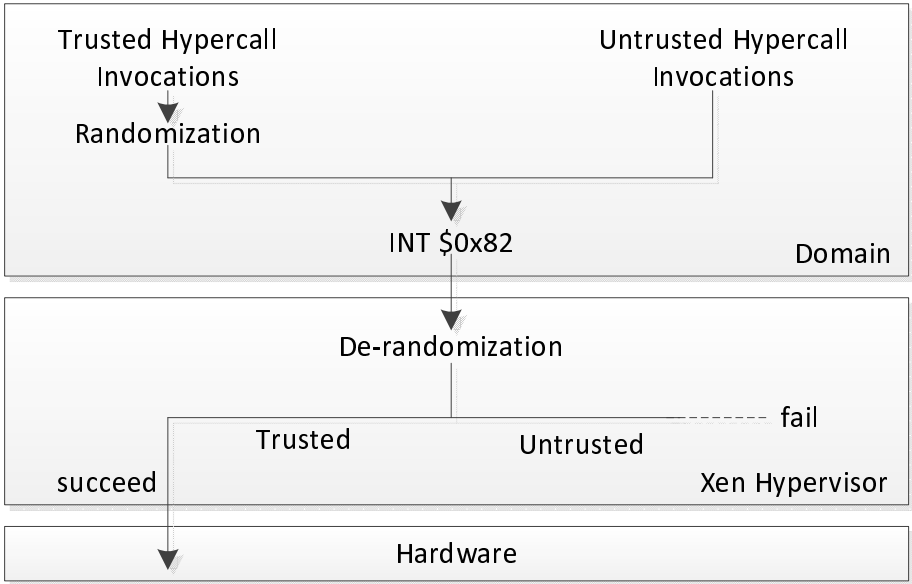


Fig. 1. RandHyp architecture

Fig. 1 shows the architecture of RandHyp. The general idea is to make the untrusted hypercall invocations unrecognized by the hypervisor, thus avoid attackers to execute privileged operations. The method we choose is to randomize trusted hypercall invocations without the attackers' consciousness, and de-randomize them in the hypervisor where attackers cannot access.

4 Implementation

In this section, we present our design of randomization and de-randomization schemes in RandHyp. RandHyp protects domains against all other domains, including dom0. Hypercall `HYPERVISOR_update_descriptor` is used as an example to typify all the rest hypercalls.

4.1 Randomization

In x86_64 XenLinux, the hypercall number of `HYPERVISOR_update_descriptor` is 10, the arguments are `ma` and `word`. What we need to do is re-assign the numbr and the arguments new values generated by some algorithm to make them not only unaccessable by attackers, but also difficult for them to guess. The randomization process is described as below.

Kernel-Level Randomization. Kernels, including loadable kernel modules and drivers, invoke hypercalls directly by calling the function `HYPERVISOR_update_descriptor` (unsigned long `ma`, unsigned long `word`), so that is the place we do randomization operations.

Firstly, we add paddings to maximum the number of arguments. After that, the function looks like `HYPERVISOR_update_descriptor(ma, word, pad1, pad2, pad3)`.

Secondly, every argument is encrypted to make a new, randomized argument using the following equation:

$$\text{arg}'_i = R_K(\text{arg}_i, \text{key}) . \quad (1)$$

R_K is our randomization algorithm using a key `key`. In RandHyp, we choose RC4 [17] encryption algorithm which is one of the most secure symmetric encryption algorithms in the world. Other algorithms are also applicable. After this step, the hypercall invocation would be `HYPERVISOR_update_descriptor(arg1', arg2', arg3', arg4', arg5')`.

Thirdly, to make our scheme more aggressive, RandHyp permutes the roles among `rbx, rcx, rdx, rsi, rdi` (`ebx, ecx, edx, esi, edi` in x86_32 Linux) registers.

Lastly, due to the design of Xen [5], hypercall numbers are limited between 0 and 128, so it is impossible for RandHyp to randomize hypercall numbers in the way it does to hypercall arguments. In this way, RandHyp adjusts by choosing a number between 0 and 128 randomly for each hypercall.

User-Level Randomization. Hypercalls may be indirectly used in user-level. The randomization operations in user-level are the same as in kernel-level, while the difference lies in where the operations are done. To invoke hypercalls in user-level, a structure `privcmd_hypercall_t`, which contains the hypercall arguments and number, are passed to the kernel by a kernel driver `privcmd`. In function `privcmd_ioctl` the arguments are written into registers and the software trap `INT $0x82` is executed. Consequently, the hypercall arguments and number should be randomized when `privcmd_hypercall_t` is constructed.

4.2 De-randomization

No matter where a hypercall comes from, the execution of the hypercall instruction `INT $0x82` generates a software trap into the Xen hypervisor, where the

hypercalls are validated and executed by the same handler. Note that we have randomized trusted kernel-level hypercalls and trusted user-level hypercalls in the same method, so we don't need to differentiate the de-randomization scheme.

RandHyp has constructed a correlation between the hypercall numbers in domains and the hypercall numbers in the Xen hypervisor. Thus, when the hypercall interrupt handler catches a hypercall, the hypervisor can pass it to the right handle function according to its number, in our cases, it is the `do_update_descriptor` function.

In order to avoid the randomized hypercall arguments from being used by the hypervisor, RandHyp de-randomizes the hypercall arguments immediately when entering the `do_update_descriptor` function.

Firstly, real arguments are selected from the paddings.

Secondly, given the encryption algorithm we use is symmetrical, the de-randomization algorithm is the same as the randomization algorithm. RandHyp recovers the original hypercall arguments $arg_i = R_K^{-1}(arg'_i, key)$ using the same key used during the randomization process. After that, Xen can do what the hypercall requires to do.

5 Security Discussion

Attacks Using Direct Hypercall Invocation. An attacker may directly invoke hypercalls in kernel-level or user-level. RandHyp can defeat such straightforward attacks.

Since hypercalls created by attackers are untrusted, they are not randomized before trapping into the hypervisor, so that they will be de-randomized into meaningless informations and fail to execute.

Attackers may attempt to acquire the randomization key directly, which is also defeated by RandHyp. The reason is that the randomization key is stored in the memory space of Xen, where attackers in domains can't access. Attackers are forced to scan the kernel code memory and collect the semantics of the instructions to get the key, which is hard to perform.

Even if the attacker can successfully get the key, the probability of him to create a right argument would be very small, for the reason that there are 38 hypercalls in a system, and each hypercall contains 5 arguments, the possibility for an attacker to get the right arguments of `HYPERVISOR_update_descriptor` is $\frac{1}{C_{(5,2) \times 38}}$, which is about 0.26%.

Attackers may try to construct plaintext-ciphertext pairs to brute-force the key. RandHyp makes this very difficult. Firstly, a strong encryption algorithm and a long key makes it almost impossible to crack the key. Secondly, because we have added paddings to the arguments, and permuted the roles among the registers, and the hypercall number is re-assigned, the attackers face difficulties in constructing plaintext-ciphertext pairs.

Attacks Using Indirect Hypercall Invocation. Instead of invoking hypercalls directly, an attacker may try to reuse existing hypercalls. In this situation,

an attacker has to achieve the memory address of the desired hypercall instructions accurately, and then jump there to eventually invoke the intended hypercalls. Although RandHyp cannot directly prevent this kind of attacks currently, existing techniques such as address space layout randomization (ASLR) [18], which makes the memory location of pre-existing code hard to predict, can be utilized to enhance security. Moreover, even if attackers can successfully hijack the control flow of a process, the hypercalls used by kernel only can accomplish basic and simple functionalities. Combining them to achieve malicious goals is almost impossible.

6 Evaluation

In this section, we first present RandHyp latency measurement results, and then present a number of attack experiments that cover all hypercalls.

6.1 Experimental Setup

Our experiments were run on a Lenovo desktop PC with a 2.53GHz Intel(R) Core(TM)2 Duo CPU processor and 2GB of RAM. All Xen modifications necessary for RandHyp were implemented on Xen 4.0.1 and Linux 2.6.32.13 kernel with Xen patches applied. The changes required to Xen were minimal, spanning only a handful of source files including the files containing the hypercall service routines and several head files. On the guest domain kernel, we modified the files containing the routines which invoke hypercalls and the file that defines hypercall number. For convenience, we give the modified system an alias, Rand-Domain, to distinguish the unmodified system, Orig-Domain.

6.2 Performance Evaluation

The performance of RandHyp should be evaluated in the following aspects: 1) Disk I/O throughput; 2) Network I/O throughput; 3) Overall system performance.

Disk I/O Performance. Disk performance is tested by using *dd*. Fig. 2 shows the results of these tests with different configuration parameters. Overall, disk throughput is down by 1-4.5%. The reading and writing latency incurred by small files is larger than that by big files, which may be caused by more frequent transversions into the hypervisor.

Network I/O Performance. *Netperf* is used to test network performance. Fig. 3 shows the results. RandHyp only causes 1.5-3% drop in throughput when using TCP, and less than 0.2% drop while using UDP. Because TCP is a connection-oriented protocol, more hypercalls have to be used to keep the connection.

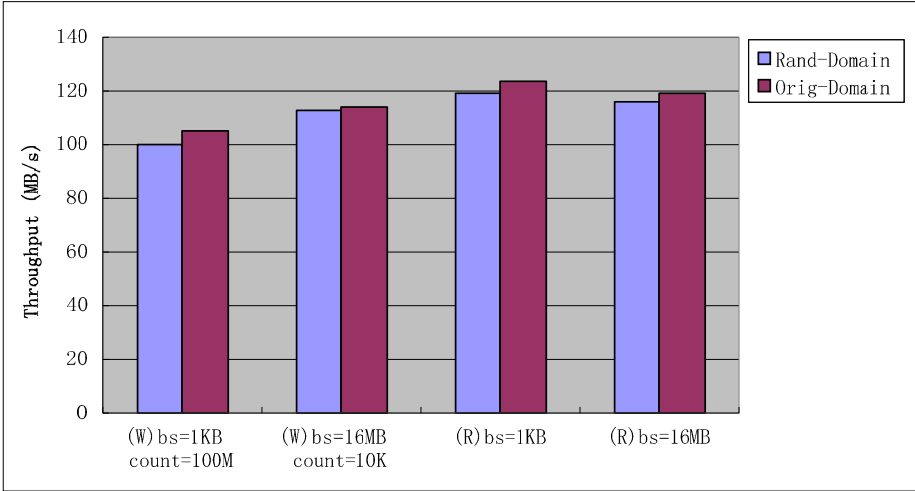


Fig. 2. Disk I/O performance using dd (higher is better)

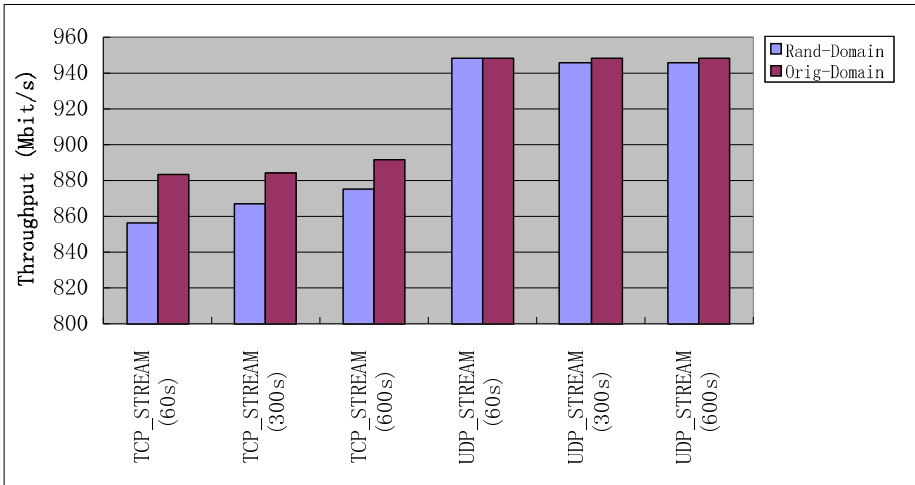


Fig. 3. Network I/O performance using Netperf (higher is better)

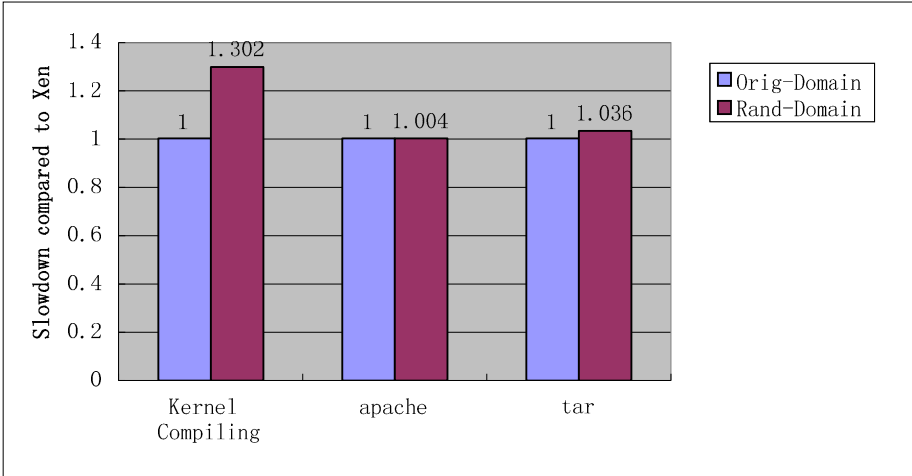


Fig. 4. Real-word benchmarks (lower is better)

Real-world Benchmarks. We use several popular applications for RandHyp latency measurement, including Linux kernel compilation, Apache *ab* benchmarking tool and *tar*. As Fig. 3 shows, the kernel compiling overhead added by RandHyp is about 30%, the Apache web server serving a 4KB static webpage 10000 times for 500 simultaneous clients adds only 0.4% overhead, and the *tar* operation adds about 0.36% overhead.

6.3 Effectiveness Evaluation

The commonly received approach to evaluate a security measure is to subject it to existing attacks. We have discussed the possible attack scenario in Section 2. However, attacks target at crushing virtualization platforms from the overlying domains have not gained widely applied in reality, which makes evaluating security a challenging job. Moreover, constructing the attacks is out of our boundaries. Given this situation, we make an attempt to demonstrate the improvement to the state of security for hypervisors by exposing them to loadable kernel modules (LKM) that are used to imitate the intrusion attacks. Each LKM tries to execute one or a set of non-randomized hypercalls as a real attacker may do. We assume that such LKMs could be successfully injected into the domain through conventional operating system security breaches in real world.

We have tested all hypercalls, and the experiment results show that these hypercalls cannot execute correctly, which verifies that the hypercalls which are not in our protection coverage can be caught by RandHyp.

These simple experiments are, though very specific and even quite contrived, they serve the purpose of verifying the effectiveness of our prototype nevertheless.

7 Conclusion

While virtualization is becoming widely accepted in both industry and academic world, its security is put on the agenda. Enhancing the security of virtualization platforms would bring more values to their popularity and usability. Although the sizes of hypervisors are a lot smaller than that of conventional operating systems, the domains running on is untrusted, and can perform illegal operations through the hypercall interface. This paper focuses on protecting the hypercall interface so as to maintain normal services to domains and guarantee the isolation between them. A transparent approach, namely RandHyp, was proposed on Xen. In RandHyp, any hypercall out of our protect range will be detected and refused to execute. RandHyp achieved this by using randomization techniques. Both hypercall's number and arguments are randomized in order to mess the attackers. Experiments show that RandHyp can effectively counter illegal hypercalls while incurring only a small overhead.

Acknowledgments. This work was supported in part by grants from the Chinese National Natural Science Foundation (61073027, 60773171, 90818022, and 61021062), and the Chinese 973 Major State Basic Program(2009CB320705).

References

1. CVE, www.cve.org
2. Amazon EC2, www.amazon.com
3. Linode, www.linode.com
4. Xenaccess library, <http://code.google.com/p/xenaccess/>
5. Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., Warfield, A.: Xen and the Art of Virtualization. In: 19th ACM Symposium on Operating Systems Principles, pp. 164–177. ACM Press, New York (2003)
6. Steinberg, U., Kauer, B.: NOVA: a Microhypervisor-Based Secure Virtualization Architecture. In: 5th EuroSys, pp. 209–222
7. Xen Interface Manual, www.xen.org
8. Zhang, F., Chen, J., Chen, H., Zang, B.: CloudVisor: Retrofitting Protection of Virtual Machines in Multi-tenant Cloud with Nested Virtualization. In: SOSP 2011 Proceedings of the 23th ACM Symposium on Operating Systems Principles, pp. 203–216 (2011)
9. Colp, P., Nanavati, M., Zhu, J., Aiello, W., Coker, G., Deegan, T., Loscocco, P., Warfield, A.: Breaking Up is Hard to Do: Security and Functionality in a Commodity Hypervisor. In: SOSP 2011 Proceedings of the 23th ACM Symposium on Operating Systems Principles, pp. 189–202 (2011)
10. Li, C., Raghunathan, A., Jha, N.K.: A Trusted Virtual Machine in an Untrusted Management Environment. In: 3rd IEEE International Conference on Cloud Computing, pp. 172–179 (2010)
11. Chen, X., Garfinkel, T., Christopher Lewis, E., Subrahmanyam, P., Waldspurger, C.A., Boneh, D., Dvoskin, J., Ports, D.R.K.: Overshadow: A Virtualization-Based Approach to Retrofitting Protection in Commodity Operating System. In: 13th International Conference on Architectural Support for Programming Languages and Operating Systems, pp. 2–13. ACM Press, New York (2008)

12. Wang, Z., Jiang, X., Cui, W., Ning, P.: Countering Kernel Rootkits with Lightweight Hook Protection. In: 16th ACM Conference on Computer and Communications Security, pp. 545–554. ACM Press, New York (2009)
13. Hoang, C.: Protecting Xen Hypercalls. Intrusion Detection/Prevention in a Virtualized Environment. MS Thesis, Univeristy of British Columbia (July 2009)
14. Ristenpart, T., Tromer, E., Shacham, H., Savage, S.: Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In: CCS 2009 Proceedings of the 16th ACM Conference on Computer and Communications Security, pp. 199–212 (2009)
15. McCune, J.M., Jaeger, T., Berger, S., Caceres, R., Sailer, R.: Shamon: A System for Distributed Mandatory Access Control. In: ACSAC 2006: Proceedings of the 22nd Annual Computer Security Applications Conference, pp. 23–32 (2006)
16. Garfinkel, T., Pfaff, B., Chow, J., Rosenblum, M., Boneh, D.: Terra: A Virtual Machine-Based Platform for Trusted Computing. In: 19th Symposium on Operating System Principles
17. Rivest, R.L.: The RC4 Encryption Algorithm. RSA Data Security, Inc. (March 1992)
18. Shacham, H., Page, M., Pfaff, B., Modadugu, N., Boneh, D.: On the Effectiveness of Address-Space Randomization. In: 11th ACM Conference on Computer and Communications Security, pp. 298–307 (2004)
19. Wang, Z., Jiang, X.: HyperSafe: A Lightweight Approach to Provide Lifetime Hypervisor Control-Flow Integrity. In: Proceedings of the 31st IEEE Symposium on Security and Privacy, Oakland, CA (May 2010)
20. Gupta, D., Cherkasova, L., Gardner, R., Vahdat, A.: Enforcing Performance Isolation Across Virtual Machines in Xen. In: van Steen, M., Henning, M. (eds.) Middleware 2006. LNCS, vol. 4290, pp. 342–362. Springer, Heidelberg (2006)
21. Kamble, N.A., Nakajima, J., Mallick, A.K.: Evolution in kernel debugging using hardware virtualization with xen. In: Proceedings of the 2006 Ottawa Linux Symposium, Ottawa, Canada (July 2006)

Role Mining under Role-Usage Cardinality Constraint

John C. John¹, Shamik Sural¹, Vijayalakshmi Atluri², and Jaideep S. Vaidya²

¹ School of Information Technology,
Indian Institute of Technology, Kharagpur, India
{johnj,shamik}@sit.iitkgp.ernet.in
² MSIS Department, Rutgers University, USA
{atluri@cimic,jsvaidya@rbs}.rutgers.edu

Abstract. With the emergence of Role Based Access Control (RBAC) as the de facto access control model, organizations can now implement and manage many high level security policies. As a means of migration from traditional access control systems to RBAC, different role mining algorithms have been proposed in recent years for finding a minimal set of roles from existing user-permission assignments. While determining such roles, it is often required that certain security objectives are satisfied. A common goal is to enforce the role-usage cardinality constraint, which limits the maximum number of roles any user can have. In this paper, we propose two alternative approaches for role mining with an upper bound on the number of roles that can be assigned to each user, and validate their performance with benchmark data sets.

Keywords: Role Based Access Control, Constrained Role Mining, Boolean Matrix Decomposition.

1 Introduction

In traditional access control mechanisms, a user can access a resource if he has been given appropriate permission on that resource. These user permissions can be represented as a matrix, called User to Permission Assignment (UPA), where the value in each cell (1 or 0) indicates whether a user has the permission or not. In a typical organization with tens of thousands of users and hundreds of thousands of permissions, the size of the UPA matrix could be quite large, making security administration increasingly difficult.

To alleviate the burden of security administration, organizations are migrating to an alternative access control mechanism known as Role Based Access Control (RBAC) [11][12]. In RBAC, users obtain permissions by virtue of being assigned to roles. A role may be assigned to multiple users. Since the number of roles is orders of magnitude lower than the number of users, RBAC significantly reduces security administration overhead.

For implementing RBAC in an organization currently using traditional UPA based access control, suitable roles have to be formulated. This process of formulation of roles is known as Role Engineering [8]. Generally there are two

approaches to role engineering: top down and bottom up. In the top down approach, business processes are initially divided into independent functional units called job functions. Roles are assigned to these job functions by associating the required permissions. Since the usual number of business processes, job functions and users is quite large, real life implementation of top down approach happens to be difficult, error prone and not quite cost effective. On the other hand, bottom up approaches use the existing user permission assignments. From the UPA, roles with corresponding permissions are derived and users are assigned to these roles. Two binary matrices UA (User to Role Assignment) and PA (Permission to Role Assignment) are thus formed. UA represents which users belong to which roles, and PA represents which permissions are contained in which roles. It is intuitively obvious that the bottom up approach can be conveniently automated. Such an automated procedure for deriving roles from a given UPA matrix, and thereby forming the UA and PA matrices is known as role mining [11,13].

While there could be many possible decompositions of a given UPA matrix into UA and PA matrices, a useful notion is to obtain a decomposition that minimizes the number of roles, which is the number of columns in the UA matrix and equivalently the number of rows in the PA matrix. This is known as the basic Role Mining Problem (RMP) [1]. RMP is an optimization problem and its decision version has been shown to be NP-Complete.

An important feature of any RBAC system is the ability to impose various constraints [11,12]. The constraints help to express the organizational security policy, thereby achieving desired security objectives. In many situations, a restriction is imposed on the maximum number of roles that can be played by any user, either due to security restrictions or due to balanced work distribution. We denote this as the *role-usage cardinality* constraint. Since the role mining process is expected to generate organizational roles and corresponding user assignments in an automated way, it is imperative that such a constraint be taken into consideration while mining the roles from a given UPA matrix. Further, from a security administration point of view also, it makes sense to let users achieve their requisite permission through lower number of roles. However, none of the existing work on role mining addresses this issue.

In this paper, we propose two alternative approaches for considering role-usage cardinality constraints while mining the roles. The first approach, called the *Role Priority based Approach* (RPA), as the name suggests, first prioritizes the roles based on their size (i.e., number of permissions within the role) and then limits the number of roles assigned to a user using this priority order. The second approach, called the *Coverage of Permissions based Approach* (CPA) chooses roles by iteratively picking the role with the largest number of permissions that are not yet assigned to that user by any other role, and then imposes the role-usage cardinality constraint. Since both these approaches use different greedy strategies, we have implemented and validated both RPA and CPA using benchmark data sets; our results show that RPA fares significantly better than CPA.

The rest of this paper is organized as follows. In Section 2, we present our proposed approaches with detailed algorithms. In Section 3, the results of running

the two proposed algorithms on different datasets are presented and analyzed. In Section 4, we review the related work on role mining. Finally, Section 5 concludes this paper and provides directions for future work.

2 Role Mining with Role-Usage Cardinality Constraints

In this section, we present our two proposed approaches: the Role Priority based Approach (RPA) and Coverage of Permissions based Approach (CPA), which mine the minimum set of roles from a UPA matrix under the constraint that the number of roles assigned to any user cannot exceed a specified upper bound.

We start off with some notation. First, as discussed earlier, UA , PA and UPA are all boolean matrices. Assume that there are n users and m permissions, with q candidate roles. Thus, UA is $n \times q$, PA is $q \times m$ and UPA is $n \times m$. Let c_{ij} represent the UA matrix entry for user i ($i = 1 \dots n$) and role j ($j = 1 \dots q$), and let r_{jt} represent the PA matrix entry for role j ($j = 1 \dots q$) and permission t ($t = 1 \dots m$). Finally, let x_{it} represent the UPA matrix entry for user i and permission t . Then, for any correct decomposition, the following conditions must be met [2]: $\sum_{j=1}^q c_{ij}r_{jt} \geq 1$ for $x_{it} = 1$ and $\sum_{j=1}^q c_{ij}r_{jt} = 0$ for $x_{it} = 0$.

Given the above, the role-usage cardinality constraint can be represented as $\sum_{j=1}^q c_{ij} \leq \text{maxcount}$, for $1 \leq i \leq n$.

2.1 Role Priority Based Approach (RPA)

The Role Priority based approach prioritizes roles by their size (i.e., the number of permissions in the role). Now, roles are assigned to each user by picking the roles whose permissions are a subset of the permissions required by that user, and are not redundant (i.e., the permission has not already been assigned through another prior-picked role). While any set of candidate roles could be used, in this paper, we use the candidate roles generated from the UPA by the greedy strategy for optimal boolean matrix decomposition (referred to as OBMD) proposed by Liu et al. [2]. Note that OBMD can produce redundant roles for some of the users. A role r is considered redundant (for a user i) if the permissions of r form a subset of the permissions of one or more other roles of the same user i . This is because the greedy approach only eliminates those candidate roles which are not used by any of the users. OBMD does not check the redundancy of roles for each user. Therefore, we eliminate such redundant roles in the candidate roles generated by OBMD. Also, the roles generated by Fast Miner [5], which OBMD uses as base, do not cover the permissions which are exclusively assigned to a user. To cover these permissions, separate roles are generated. The roles are then prioritized based on the number of permissions contained in each role. To restrict the number of roles of each user upto the specified upper bound, the roles are selected in the order of their priority. If all of the permissions of the user are not covered after selecting $\text{maxcount} - 1$ roles, where maxcount is the specified upper bound, the remaining uncovered permissions are formed into a separate role.

These steps are summarized in Algorithm 1. Formation of initial UA matrix and the associated PA matrix using OBMD is done in Line 1. Then, a Role Priority (RP) list is created in which roles are maintained in the descending order of the number of permissions associated with each role (Line 2). The variable *maxcount* denotes the maximum number of roles allowed for a user and *rolecount* denotes the number of roles so far assigned to a user. In lines 4-12, for each user, the roles are selected one by one as per the order in the RP list until all permissions of the user are covered or until the *rolecount* becomes *maxcount* - 1. If the permissions of the selected role form a subset of already covered permissions of the user, they are redundant and are eliminated during the process (Lines 7-8). If the *rolecount* value reaches *maxcount* - 1 and all the permissions of the user are not yet covered, a new role *r* is generated comprising of all the uncovered permissions (if it is not an existing role; otherwise, uses the existing role *r*) and includes *r* in the PA matrix, also setting the corresponding c_{ir} cell of the UA matrix to 1 (Lines 13-23).

Algorithm 1. Role Priority based Algorithm (RPA)

```

1: Use the OBMD algorithm to form the decomposition into  $UA$  and  $PA$  matrices
2: Prepare Role Priority (RP) list by ordering the roles in descending order of the
   number of permissions in the role
3: for each user  $i$  do
4:   Set  $rolecover = null$  and  $rolecount = 0$ 
5:   while  $rolecover$  does not contain all permissions of  $i$  and  $rolecount \neq$ 
       $maxcount - 1$  do
6:     Find the next role,  $k$ , in the RP list whose permissions are a subset of the
       permissions of user  $i$ 
7:     if  $k \subseteq rolecover$  then
8:       Set  $c_{ik} = 0$ 
9:     else
10:       $rolecover = rolecover \cup \{\text{permissions of } k\}$  and Set  $rolecount = rolecount +$ 
        1
11:    end if
12:  end while
13:  Set  $D = \text{Permissions of } i - \text{union of permissions of all roles of } i$ 
14:  if  $D = \phi$  then
15:    Set all remaining roles of  $i$  to be 0
16:  else
17:    if  $D = \text{existing role } r \text{ in the PA matrix}$  then
18:      Set  $c_{ir} = 1$  and Set all remaining roles of  $i$  to 0
19:    end if
20:  else
21:    Generate a new role  $r$  with all permissions of  $D$ 
22:    Include  $r$  in the PA and UA matrices(set  $c_{ir} = 1$ ) and Set all remaining roles
      of  $i$  to 0
23:  end if
24: end for

```

Table 1. Sample input data set (in the form of UPA matrix)

	p1	p2	p3	p4	p5
u1	1	0	0	0	1
u2	0	0	1	1	0
u3	1	0	1	1	0
u4	1	1	1	1	1
u5	0	0	1	1	0
u6	1	1	0	0	0

Table 2. Initial UA and PA matrices generated by Fast Miner

	r1	r2	r3	r4	r5		p1	p2	p3	p4	p5
u1	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}	r1	1	0	0	0	0
u2	c_{21}	c_{22}	c_{23}	c_{24}	c_{25}	r2	1	0	0	0	1
u3	c_{31}	c_{32}	c_{33}	c_{34}	c_{35}	r3	0	0	1	1	0
u4	c_{41}	c_{42}	c_{43}	c_{44}	c_{45}	r4	1	1	0	0	0
u5	c_{51}	c_{52}	c_{53}	c_{54}	c_{55}	r5	1	0	1	1	0
u6	c_{61}	c_{62}	c_{63}	c_{64}	c_{65}						

Table 3. Final UA and PA matrices obtained from OBMD

	r1	r2	r3	r4		p1	p2	p3	p4	p5
u1	1	1	0	0	r1	1	0	0	0	0
u2	0	0	1	0	r2	1	0	0	0	1
u3	1	0	1	0	r3	0	0	1	1	0
u4	1	1	1	1	r4	1	1	0	0	0
u5	0	0	1	0						
u6	1	0	0	1						

Illustrative Example. We now consider an illustrative example given in Table 1 for explaining both the OBMD and RPA algorithms. OBMD employs the Fast Miner algorithm to generate q number of candidate roles and to form the binary PA matrix. The initial UA and PA matrices consisting of five candidate roles generated through Fast Miner are shown in Table 2.

The UA matrix is formed by selecting appropriate roles from the PA matrix based on the count of the conditions being satisfied while selecting a role. Some of the candidate roles which are not used by any of the users can finally be rejected, thereby reducing the total number of roles. The final matrices after applying the appropriate conditions as mentioned above are given in Table 3.

Assume the value of the cardinality constraint ($maxcount$) is 2. In Step 1, the OBMD algorithm gives an output as shown in Table 3. The RP list is prepared as $\{r2, r3, r4, r1\}$. RPA then selects the appropriate roles from the list for each user. For user $u1$, role $r2$ is selected. With this role all permissions of $u1$ are

covered. Similarly, role $r3$ is found for user $u2$. The first role selected for user $u3$ is role $r3$. Now the value of $rolecount$ becomes 1 which is equal to $maxcount - 1$. It can be observed that the remaining set of uncovered permissions is the same as the existing role $r1$, and hence, $r1$ is selected. For user $u4$, role $r2$ is initially selected. Again the value of $rolecount$ becomes equal to $maxcount - 1$. Now the uncovered permissions are not the same as any existing role and so a new role $r5$ is created with all the uncovered permissions. For user $u5$, the algorithm initially selects role $r4$. With this role all permissions of the user are covered. Thus, the algorithm finds a total of 5 roles for a $maxcount$ value of 2. The final UA and PA matrices are shown in Table 4.

It may be noted that the OBMD algorithm generates four roles as shown in Table 3. But user $u4$ has four roles which exceeds $maxcount$. The decomposition done by RPA restricts the number of roles of users to the specified upper bound. The total number of roles is increased to five to satisfy this constraint.

Table 4. UA and PA matrices generated by RPA

	r1	r2	r3	r4	r5
u1	0	1	0	0	0
u2	0	0	1	0	0
u3	1	0	1	0	0
u4	0	1	0	0	1
u5	0	0	1	0	0
u6	0	0	0	1	0

	p1	p2	p3	p4	p5
r1	1	0	0	0	0
r2	1	0	0	0	1
r3	0	0	1	1	0
r4	1	1	0	0	0
r5	0	1	1	1	0

2.2 Coverage of Permissions Based Approach (CPA)

The second approach, called the Coverage of Permissions based Approach (CPA), selects roles based on the number of permissions that are as yet unassigned to a user, while still enforcing the cardinality constraint. Fast Miner is employed to form the initial set of candidate roles, similar to that shown in Table 2. Effectively, it works on decompositions containing all candidate roles. In order to limit the number of roles of each user upto the specified bound, the algorithm keeps on finding roles which contain the largest number of uncovered permissions (permissions not yet assigned through any role) for the user. If all permissions of the user are not covered after selecting $maxcount - 1$ roles, the remaining uncovered permissions are grouped into a separate role. The redundant roles are also eliminated during the process.

The steps are summarized in Algorithm 2. Fast Miner is used to generate q candidate roles and form the initial PA matrix and the UA matrix in Line 1. Next, c_{ij} values of the UA matrix are set to 0 according to the condition $\sum_{j=1}^q c_{ij}r_{jt} = 0$ (Line 2). For each user i , in each step, the algorithm finds a role which covers the maximum number of uncovered permissions of i from among all the roles r where $c_{ir} = 1$. The process is repeated until all the permissions of the user are covered or $rolecount$ becomes $maxcount - 1$ (Lines 4-8). If the

rolecount is *maxcount* – 1 and all the permissions of user *i* are not yet covered, a new role *r* is generated with the uncovered permissions (if it is not an existing role; otherwise, the existing role *r* is used). Finally, *r* is included in the PA matrix and the corresponding c_{ir} of the UA matrix is set to 1 (Lines 9-20).

Algorithm 2. Coverage of Permissions based Algorithm (CPA)

```

1: Generate candidate roles by taking intersection of the permissions of every pair of
   users. Form PA matrix and associated UA matrix. (Fast Miner approach)
2: Set the required cells of  $c_{ij}$  to 0 as per  $\sum_{j=1}^q c_{ij}r_{jt} = 0$  for all  $x_{it} = 0$ 
3: for each user i do
4:   Set rolecover = null and rolecount = 0
5:   while rolecover does not contain all permissions of i and rolecount  $\neq$ 
     maxcount – 1 do
6:     Find the role k that contains the maximum number of permissions of i which
       are not yet contained in rolecover, while not adding any unallowed permissions
       to i
7:     Set  $c_{ik}$  to be 1, Set rolecover = rolecover  $\cup$  {permissions of k} and Set
       rolecount = rolecount + 1
8:   end while
9:   Set D = Permissions of i - union of permissions of all roles of i
10:  if D =  $\phi$  then
11:    Set all remaining roles of i to be 0
12:  else
13:    if D = existing role r in the PA matrix then
14:      Set  $c_{ir} = 1$  and Set all remaining roles of i to 0
15:    end if
16:  else
17:    Generate a new role r with all permissions of d
18:    Include r in the PA and UA matrices
19:    Set  $c_{ir} = 1$  and Set all remaining roles of i to 0
20:  end if
21: end for

```

Illustrative Example. Consider again the same dataset given in Table 1 and assume the value of cardinality constraint to be 2. The initial decomposition is as given in Table 2. In the next step, the CPA algorithm finds the cells of UA matrix which are to be set to 0 using the condition $\sum_{j=1}^q c_{ij}r_{jt} = 0$. Then, for each user, it finds roles which cover maximum number of uncovered permissions. User *u1* has two roles from which the algorithm selects role *r2* since it covers more permissions of *u1* as compared to role *r1* and with that role all permissions of *u1* are covered. So it sets the remaining cells of *u1* in the UA matrix to 0. Similarly, user *u2* gets role *r3* and user *u3* gets role *r5*. User *u4* is assigned to role *r5* initially. Now the *rolecount* becomes equal to *maxcount* – 1. So a new role *r6* is formed with all uncovered permissions. The new role is included in the UA matrix and c_{61} is set to 1. Then user *u5* gets role *r3* and finally user *u6* gets role *r4*. The unused role *r1* is removed. The algorithm completes execution after

Table 5. UA and PA matrices generated by CPA

	r1	r2	r3	r4	r5		p1	p2	p3	p4	p5
u1	1	0	0	0	0	r1	1	0	0	0	1
u2	0	1	0	0	0	r2	0	0	1	1	0
u3	0	0	0	1	0	r3	1	1	0	0	0
u4	0	0	0	1	1	r4	1	0	1	1	0
u5	0	1	0	0	0	r5	0	1	0	0	1
u6	0	0	1	0	0						

mining five roles. The final decomposition matrices are shown in Table 5 (after renaming roles from r_2 to r_6 as r_1 to r_5).

As is evident, from the above discussions, RPA and CPA use different greedy choices. The output decompositions in the examples are also different. But the number of roles obtained is the same in the two cases for the example dataset even though the actual roles are different. In the next section, we give a comparative performance of both these algorithms with benchmark datasets.

3 Experimental Results

Algorithms 1 (RPA) and 2 (CPA) were executed on a number of datasets as listed in Table 6. These are real-world datasets used to evaluate the work reported in 9. Table 6 also provides the sizes of these datasets in terms of the number of users, number of unique permissions as well as the permission size, which is the total number of permissions added over all users.

Table 6. Data sets

Dataset	Users	Permissions	Total permission size
Healthcare	46	46	1,486
Domino	79	231	730
EMEA	35	3,046	7,220
Firewall 1	365	709	31,951
Firewall 2	325	590	36,428

We plot the variation of the number of roles (y-axis) generated by RPA and CPA with varying constraint value (x-axis). The roles discovered by both algorithms satisfy the given constraint. The plots are shown in Figures 1 to 3.

The number of roles generated by RPA is less than or equal to that of CPA in all datasets. By observing the output for the datasets Healthcare and Firewall 2, it can be concluded that the number of generated roles decreases when the value of the constraint increases. The number of roles generated by OBMD in the Healthcare dataset is 17. But RPA generates only 15 roles when the value of the constraint is greater than or equal to 2. When the constraint value is set to

1, it generates 18 roles. On the contrary, CPA gives 18 roles for all values of the constraint. Similarly, for Firewall 2 dataset, OBMD generates 10 roles and RPA produces the same number of roles when the constraint value remains greater than or equal to 2. It generates 11 roles for a constraint value of 1. Here also CPA generates 11 roles for all cases.

For the datasets Domino and EMEA, the reverse occurs. There the number of roles generated shows a decrease when the value of the constraint decreases. This appears to be contradictory. However, the reason is that the candidate roles generated by the Fast Miner algorithm in those datasets do not contain roles which produces optimum result. Also, presence of many exclusive permissions in the dataset, results in increased number of roles. Such exclusive roles are not part of the roles generated by Fast Miner. For the Domino dataset, OBMD produces 29 roles. However, the optimal number is 23, which is produced by both RPA and CPA algorithms for the constraint value 1. Similarly for the EMEA dataset, the number of roles generated by OBMD is 122 but the optimal number is 34 which is produced by both the proposed algorithms for the constraint value 1. Thus, OBMD does not give optimal number of roles for these datasets. In both of our approaches, when the value of the constraint is reduced and it approaches a value of 1, many of the roles get merged with other roles to form a new role. As a result, in these datasets, the algorithms give optimal result when the value of the constraint is low. An improvement in the Fast Miner algorithm would give a better candidate role generation.

For the Firewall 1 dataset, OBMD produces 75 roles. RPA gives the optimal result for constraint values greater than or equal to 4 and the minimum number of roles generated by this algorithm is 72. When the value of the constraint is less than 4, the number of roles generated by RPA increases and for a value of 1 it produces the highest number of roles, which is 90. However, for this dataset, CPA never finds an optimal result. It produces minimum number of roles (90 roles) for the constraint value 1.

4 Related Work

Several attempts have been made so far for bottom-up identification of roles in RBAC. Schlegelmilch and Steffens [4] introduced a role mining tool named ORCA which forms a hierarchy of permission clusters. A subset enumeration technique is used in [5]. After clustering the users having similar permissions, it intersects all possible combinations of user's permissions and generates a set of candidate roles. These roles are then prioritized based on the number of users sharing the roles. The work in [1] maps RMP into an equivalent problem – the Largest Uncovered Tile Mining (LUTM) problem, which uses the concept of database tiling [6]. Generating minimum number of roles in RMP is equivalent to finding the minimum number of tiles in the tiling problem. Zhang et al. [8] use a graph optimization technique to form role hierarchies. In [2], Lu et al. model the problem into optimal Boolean matrix decomposition problem. In order to generate practical roles, [7] ignores some permission assignments. Authors

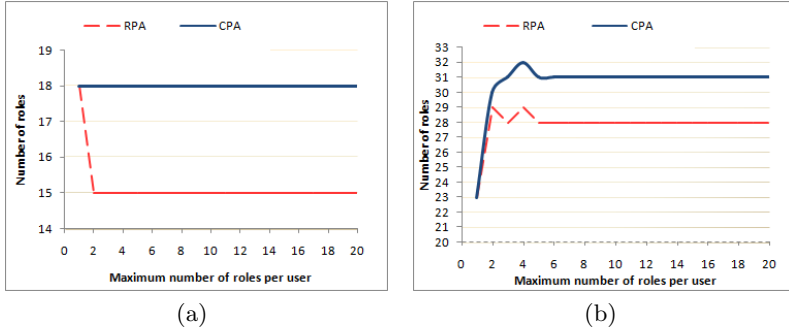


Fig. 1. Number of roles generated for (a) Healthcare dataset and (b) Domino dataset

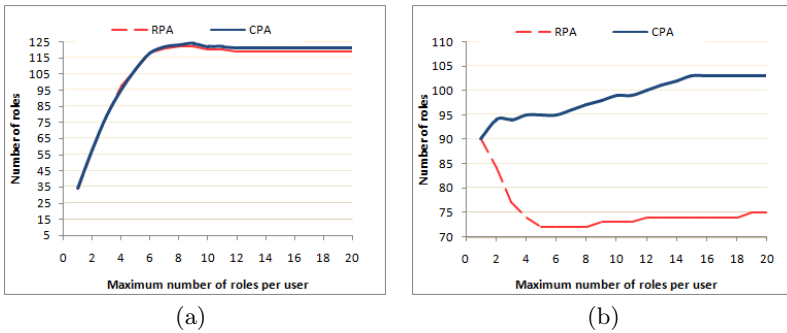


Fig. 2. Number of roles generated for (a) EMEA dataset and (b) Firewall 1 dataset

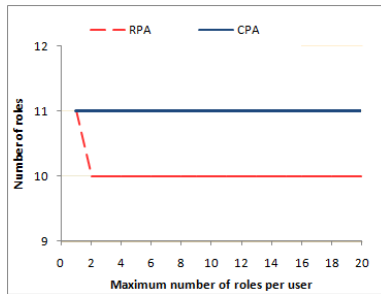


Fig. 3. Number of roles generated for Firewall 2 dataset

of [9] proposed an algorithm which uses bipartite graph formulation to find approximately minimum number of roles by forming biclique cover of edges of the graph.

Little work has been done on RMP with cardinality constraints. Kumar et al. [10] considered RMP with a constraint which limits the maximum number of permissions that a role can have. They use a combination of clustering and

constrained permission set mining to generate roles. Recently, Hingankar and Sural [3] have worked towards role mining with a constraint on the maximum number of users in a role. They use the biclique cover approach to arrive at the solution. However, there is no available literature on RBAC role mining with an upper bound on the number of roles a user can have.

5 Conclusions and Future Direction

In this paper, we consider the problem of constrained role mining, by enforcing the limit on the number of roles a user can have. We have presented two different approaches to enforce the role-usage cardinality constraint. Variation in the number of generated roles by varying the value of the constraint has been studied on real-world datasets for both these approaches.

Organizations may impose restriction on multiple RBAC parameters at the same time. The parameters could be upper bounds on number of roles to a user, number of permissions in a role, number of roles in a permission, etc. While we consider a single constraint in this paper, the combined effect on the mining process when many constraints are considered simultaneously needs be studied. The work can be extended to find an optimal solution to the role mining problem with such multiple constraints.

Acknowledgement. This work is partially supported by a research grant from the Department of Science and Technology, Government of India, under Grant No. SR/S3/EECE/082/ 2007. The work of Atluri is supported in part by the National Science Foundation under grant CNS-0746943.

References

1. Vaidya, J., Atluri, V., Guo, Q.: The Role Mining Problem: Finding a Minimal Descriptive Set of Roles. In: Proceedings of the ACM Symposium on Access Control Models and Technologies (SACMAT), pp. 175–184 (2007)
2. Lu, H., Vaidya, J., Atluri, V.: Optimal Boolean Matrix Decomposition: Application to Role Engineering. In: Proceedings of the IEEE 24th International Conference on Data Engineering, pp. 297–306 (2008)
3. Hingankar, M., Sural, S.: Towards Role Mining with Restricted User-Role Assignment. In: Proceedings of the 2nd International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace Electronic Systems Technology (Wireless VITAE), pp. 1–5 (2011)
4. Jürgen, S., Ulrike, S.: Role mining with ORCA. In: Proceedings of the Tenth ACM Symposium on Access Control Models and Technologies, pp. 168–176 (2005)
5. Vaidya, J., Atluri, V., Warner, J.: Role Miner: Mining Roles using Subset Enumeration. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, pp. 144–153 (2006)
6. Geerts, F., Goethals, B., Mielikäinen, T.: Tiling Databases. In: Suzuki, E., Arikawa, S. (eds.) DS 2004. LNCS (LNAI), vol. 3245, pp. 278–289. Springer, Heidelberg (2004)

7. Zhang, D., Kotagiri, R., Tim, E., Trevor, Y.: Permission Set Mining: Discovering Practical and Useful Roles. In: Computer Security Applications Conference, ACSAC, pp. 247–256 (2008)
8. Zhang, D., Kotagiri, R., Tim, E.: Role Engineering using Graph Optimization. In: Proceedings of the 12th ACM Symposium on Access Control Models and Technologies, pp. 139–144 (2007)
9. Alina, E., William, H., Nikola, M., Prasad, R., Robert, S., Tarjan Robert, E.: Fast Exact and Heuristic Methods for Role Minimization Problems. In: Proceedings of the 13th ACM Symposium on Access Control Models and Technologies, pp. 1–10 (2008)
10. Kumar, R., Sural, S., Gupta, A.: Mining RBAC Roles under Cardinality Constraint. In: Jha, S., Mathuria, A. (eds.) ICISS 2010. LNCS, vol. 6503, pp. 171–185. Springer, Heidelberg (2010)
11. Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Role Based Access Control Models, pp. 38–47. IEEE Computer Society Press (1996)
12. Ferraiolo, D.F., Sandhu, R., Gavrila, S., Richard Kuhn, D., Chandramouli, R.: Proposed NIST Standard for Role-Based Access Control. *ACM Transactions on Information and System Security* 4(3) (2001)
13. Molloy, I., Chen, H., Li, T., Wang, Q., Li, N., Bertino, E., Calo, S.B., Lobo, J.: Mining Roles with Multiple Objectives. *ACM Transactions on Information and System Security* 13(4), 36 (2010)

HIDE_DHCP: Covert Communications through Network Configuration Messages

Ruben Rios, Jose A. Onieva, and Javier Lopez

Computer Science Department
University of Malaga, Spain
{ruben,onieva,jlm}@lcc.uma.es

Abstract. Covert channels are a form of hidden communication that may violate the integrity of systems. Since their birth in multilevel security systems in the early 70's they have evolved considerably, such that new solutions have appeared for computer networks mainly due to vague protocols specifications. We analyze a protocol extensively used today, the Dynamic Host Configuration Protocol (*DHCP*), in search of new forms of covert communication. From this analysis we observe several features that can be effectively exploited for subliminal data transmission. This results in the implementation of HIDE_DHCP, which integrates three covert channels that accommodate to different stealthiness and bandwidth requirements.

Keywords: Covert channels, System Information Security, Network Security.

1 Introduction

Evolution of computer networks in recent years has led to the development of new services and, simultaneously, to the emergence of new threats for interconnected systems. *Covert Channels* is a sub-discipline of Information Hiding which has been usually considered a threat to security in both centralized (e.g. [14,10]) and distributed systems (e.g. [15]). Network covert channels can be defined as a way of transmitting hidden information (i.e. unnoticed to a possible observer) using communication protocol features that are not properly defined or whose main functionality is misused. Therefore, as established, it should be possible to design covert channels at any level of the OSI stack, from the link layer to the application layer.

Traditionally, covert channels have been grouped into two main categories [1]: *storage* and *timing* channels. This classification refers to how the information to be transmitted is hidden. In the former, the sender hides data in memory areas to which the receiver has access. These memory areas might be certain header fields in network packets which are either obsolete or whose modification does not affect the proper functioning of the protocol. On the other hand, timing channels are based on modulating the behavior of the sender in order to encode information, which in practice is implemented packet rate alterations.

Covert channels have not only been studied from a theoretical perspective. Additionally, several tools have been designed to hide information flows between two or more hosts. Usually, these tools take advantage of protocols widely used in most existing networks, such as TCP/IP [19], HTTP [6] or DNS [13]. Nevertheless, there are still many protocols that have not been explored for covert communication channels. In this work we focus on the analysis and implementation of covert communications in the widely deployed DHCP protocol.

The rest of this paper is organized as follows. Section 2 provides an overview of the evolution of covert channels, from its origins to the present. Section 3 depicts a potential usage scenario, obtains its requirements and provides an exhaustive analysis on the opportunities for information hiding in DHCP. Next, Section 4 presents HIDE_DHCP, a new tool that integrates three forms of covert communications resulting from the previous analysis. Moreover, Section 5 presents the advantages and limitations of the implemented methods. Finally, Section 6 concludes this work and examines future research directions.

2 Related Work

The covert channel concept was first introduced by Lampson [14] in multilevel security systems to describe the ability of high security processes to signal information to other processes with lower security requirements. This concept began to draw security experts attention and was included in [1] and later in [10] for the evaluation of systems security. Also, a chapter of [18] was devoted to covert channels analysis and detection, where covert channels are first defined from a perspective that could be applied not only to multilevel security systems but also to computer networks.

Network covert channels were originally studied in [9], where two storage channels and one timing channel were identified. This work paved the way for new studies like [22], which analyses the IEEE 802.2 and 802.5 protocol families, and [11] which discusses the entire OSI reference model. The first known implementation, *Covert_TCP*, which is owed to Rowland [19] uses three methods to hide information in the TCP and IP headers [4]. Shortly after, *LOKI2* [4], a new covert channel that uses the payload in ICMP packets, was developed. Besides, *Ping-Tunnel* [21] took advantage of ICMP to implement a subliminal channel. Some other well-known protocols were also exploited by *FirePass* [6] and *Ozyman* [13], which created covert channels in HTTP and DNS respectively.

In general, the aforementioned channels benefit from misused packet headers but some other authors developed timing channels as well. The first timing channel implementation is presented in [2], where the authors had to deal with synchronization problems due to the absence of a common precision clock. In [20], Shah et al. implement the *JitterBug*, which adds negligible delays to keystrokes in interactive network applications (e.g., telnet) and these delays conceal a message that is retrieved by a remote party. Also, [17] presents a highly reliable

¹ Even IPv6 was later found to be vulnerable to 22 forms of covert channels [16].

timing channel, *Cloak*, that encodes a message by a unique distribution of various packets over several TCP flows. In addition, many advances have been done on the detection of timing channels based on, for example, the similarity of time intervals [3] and the use of entropy and conditional entropy [7].

Although, extensive research has been conducted in the design and implementation of covert communication channels there are still numerous protocols which are susceptible to convey hidden data either for legitimate or illegitimate purposes. In particular, we evaluate the DHCP protocol which, to the best of our knowledge, has not been exploited for such purposes yet.

3 Covert Channel Analysis and Requirements

In this section we consider a fictitious setting from which we obtain the requirements for a new covert channel. After the identification of the needs, we perform an analysis on the potential provided by the protocol and devise several methods for information hiding.

3.1 The Scenario

Consider a scenario where *Alice* and *Bob* want to communicate. Consider also that Alice works and has privileged access to part of a network deployed in the embassy of a country that is holding a summit of the Heads of State. Bob is visiting the embassy and has some information of extreme importance but they cannot be seen communicating or they would raise suspicion among the rest of participants. From this scenario we identify the following properties for the hidden communication channel:

- **Stealthiness:** This feature is leveraged by the fact that the protocol to be chosen has not been previously used to transmit covert data.
- **Moderate bandwidth:** The capacity is not a critical factor since the motivation of the channel is the transmission of small amounts of information, such as a cryptographic key.
- **Reliability:** The data being communicated is sensitive, therefore it is necessary that these are correctly received. The loss of small pieces of data may turn into a great loss of information.
- **Locality.** The goal is not to convey information through the Internet but to create a hidden communication channel between nodes in a local or personal area network, where there might be other entities monitoring the communications.
- **Unidirectionality.** The channel is not intended to allow the users to exchange information, thus having a one-way communication channel will suffice.

A suitable candidate given the above requirements is DHCP [5]. In the following we analyze the protocol in order to find the opportunity to conceal information within DHCP communications.

Table 1. DHCP Messages

Packet	Direction	Description
Discover	C → S	Broadcast message used to find servers.
Offer	C ← S	Response to DHCP Discover. It contains a configuration offer.
Request	C → S	Message to confirm the acceptance of the parameters offered by the server or the renewal of a previous configuration.
Ack	C ← S	Message acknowledging the parameters agreed with the client. It includes the IP address to be used.
Nak	C ← S	Indicates the client the non acceptance of the configuration, either because the IP is incorrect or the lease has expired.
Decline	C → S	Message to indicate that the IP address is already in use by another client.
Release	C → S	Message to reject the given IP address, thus canceling the current lease.
Inform	C → S	Message used to request more information about the configuration; the client already has an IP address.

3.2 Protocol Analysis

DHCP (*Dynamic Host Configuration Protocol*), an auto-configuration protocol used on IP networks, can be considered as an extension of BOOTP (*Bootstrap Protocol*). It is an application-level protocol which uses UDP as its transport layer on ports 67 and 68 for the server and the client respectively. Regardless of using a datagram service, packet loss is unusual since the scope of this protocol is usually found within the same local area network. Despite this, DHCP provides some recovery mechanisms against packet loss, such as the retransmission of packets when no response is received after a given period of time.

The DHCP protocol uses a request-response model in which the client is always in charge of starting the communication. Client-server interaction is done in transactions, where several messages are exchanged. Table 1 provides a description of the different types of messages and the direction of the communication, where C and S represent the client and the server, respectively.

Two message exchange models exist in DHCP. The first model is used the first time a client requests the network configuration parameters, or in the case the configuration lease expires. Fig. 1 depicts a complete configuration process. The second model comes into play if the lease is still valid but the client is trying to renew that specific configuration with the server. This model can be regarded as a sub-model of the previous one (see dashed arrows in Fig. 1) since it starts with a Request message aiming to renew the configuration parameters with the DHCP server. The usual exchange is as follows: Discover, Offer, Request, Ack and, optionally (dotted arrow), Release. However, the first two messages are only possible in the first model. This is important because modifying the natural communication models might alert an observer of the presence of the channel.

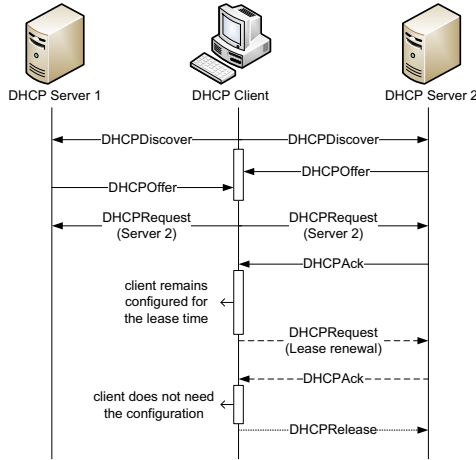


Fig. 1. Potential exchange models between DHCP clients and servers

DHCP messages share a common header structure regardless of whether they come from the server or the client. A detailed illustration of a DHCP packet is given in Fig. 2. The packet is divided into various fields which are kept for backward compatibility with the original BOOTP protocol, thus providing more chances to allocate hidden data. In the following, we analyze the fields that we consider more relevant for covert communication purposes.

First, the transaction identifier (*xid*) field has the potential to convey up to 32 bits of covert information. Interestingly, according to the protocol RFC [5] this value must be randomly created by the client. This implies that there is no agreed algorithm for the identifier generation, as with the sequence number generator in TCP, which makes the detection of the covert channel more challenging.

The field *secs* can be used in a similar way as proposed in [8] to hide information in the TCP timestamps option. The low-order bit of TCP timestamps is basically random due to host internal timings, thus it is easy to change this value to convey data without alerting a potential observer. The concealment of information without a technique such as the one proposed in Griffin’s work could raise suspicions or could cause a particular server to stop working correctly upon the reception of new messages apparently delayed in time.

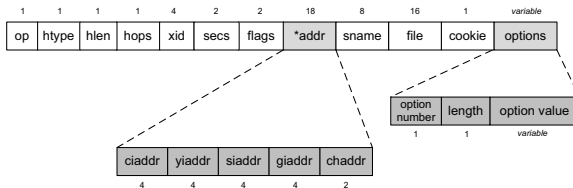


Fig. 2. DHCP header format

Also, the field *chaddr* presents at least two potential ways of carrying hidden data. In the first place, it could be possible to use a technique similar to the one used by Rowland [19] in the third method developed for Covert_TCP, taking advantage of bouncing DHCP servers. In order to make this possible, the MAC address of the entity to be contacted should be included in *chaddr*. Besides, the data to be sent must be included in a header field which is not modified during the transaction, such as the *xid*. This makes the DHCP server to respond to the client specified in the *chaddr* instead of the real sender of the message, thus unknowingly helping the sender to convey the data to its destination. The only inconvenience would be having a datagram with a different MAC address to the one specified in the DHCP header. Although this is technically possible without disturbing the normal operation of the network, this could be easily identified as a spoofing attack by an intrusion detection system. The second chance for concealing data is due to the length of the *chaddr* field, which is 16 bytes long while most of the times it is used for Ethernet addresses (6 bytes). Since the amount of relevant data in *chaddr* is defined by the *hlen* field, the remaining bytes can be used to convey the covert data. These remaining bytes are to be considered by ordinary servers as garbage and they do not analyze them.

The fields *sname* and *file* are potentially excellent carriers of information due to their large size, 64 and 128 bytes respectively. Both fields consist of null-terminated strings ('\0'), thus our data might be included after this character without negatively impacting other clients or servers, which would consider that such fields do not contain relevant information. Although these fields are usually set to null by the operating system when they are not carrying their own data, in some situations they might contain information belonging to the Options field, which for reasons of packet capacity cannot be held within the field designed for such purpose. To indicate this situation, the RFC states that the option 52 (*Overload*) must be included. The main inconvenient in using these fields as covert information carriers is that, though not specified in the protocol definition, they are often filled with zero bytes when the *Overload* option is not active. This may alert to trained traffic analyzers.

Moreover, the *Options* field also presents interesting features which might be used to conceal information. The fact that it is a variable length field provides the ability to withhold much data, either on (1) the number of options used or (2) on the way options are ordered. Furthermore, we might also encode data by (3) placing a specific option in a particular position within the Options field. To clarify this, we provide further explanations on how each of the proposed methods could be used to encode the "ALOHA" string:

1. Number of options: for the sake of simplicity we assume that only capital letters can be sent, thus limiting the number of options to be included. To further reduce the number of options, instead of using the usual ASCII encoding we might use our own codification. For example, letter 'A' could

- be signaled by transmitting a message with 2 options², letter 'B' with 3 options, and so on. Therefore, to transmit "ALOHA" the client needs to send 5 DHCP messages, each of them with the following number of options: 2 ('A'), 13 ('L'), 16 ('O'), 9 ('H'), and 2 ('A') options respectively.
2. Options ordering: we assume that we want to encode the ASCII alphabet (8 bits) by setting options in a particular order. There are at least two ways of doing this. In the first case, we must have 8 reference options and if a particular option appears, the bit related to that option is equal to 1 and 0 otherwise. The second way is to have a reference message to compare with: if a particular option appears in the message received in the same order as in the reference message, the bit corresponding to this option is set to 1 and 0 otherwise. For example, assume we have an alphabet comprised of 16 symbols, then we should use 4 options to encode one character. Also assume that we considered the options *A*, *B*, *C* and *D* in that particular order. If we receive a message containing *C*, *B*, *A*, *D*, then the hidden message is 0101.
 3. Option type: by using the type of an option placed in a particular position, we might encode an alphabet of up to 8 bits (options types are within the range 0 to 255). Without loss of generality, let us consider the option placed in the second position as the option used to conceal the covert data. In order to send the message "ALOHA" five messages are needed and, for each of them, the second option must be option number 65 ("*NIS-Server-Addr*"), 76 ("*STDA-Server*"), 79 ("*Service Scope*"), 72 ("*WWW-Server*") and 65; which are the ASCII equivalent codes of the string to be sent. In order to increase the capacity of this channel, instead of encoding a single symbol per packet, several different options could be used as data carriers within every packet.

One of the biggest downsides presented by the use of the above methods is that depending on the type of message there are a number of options which are either mandatory or not permitted. Therefore, the ability to deploy any of these solutions will depend on whether the introduction of unauthorized options in specific packets will influence the operation of the protocol. Another drawback, which is specific to the third proposed method, is that encoding messages with repeated characters may entail the creation of repeated options within the same packet which, although possible in some cases, may raise suspicions. However, this issue might be easily solved by sending characters only if the next character to encode has not already been included in the current message.

Finally, one might take advantage of the existence of some particular options that are either undefined or declared for private use. These options (84, 96, 102-111, 115, 126, 127, 137-149, 151-174, 178-207, 212-219, 222 and 223 are not assigned or have been erased; and 224-254 are for private use), specially those for private use, are potentially excellent information carriers since their structure has not been defined. Thus, anyone could define the *value* field in the option to be as large as necessary, up to 255 bytes.

² The RFC requires the occurrence of at least 2 options: "*DHCP Message Type*" and "*End*".

4 HIDE_DHCP: A New Subliminal Channel

In this section we describe the implementation of HIDE_DHCP, which integrates 3 methods for covert communications using the *xid*, *Sname* and *File*, and *Options* fields. The implementation is based on an already existing DHCP code, that we modified, developed by the Internet Systems Consortium (ISC) numbered as 4.1.1-P1 [12] and distributed in Linux operating systems.

4.1 Xid Implementation

The *xid* field is 4 bytes long and in the original ISC code it is loaded with the result of the *random()* function. This process is performed twice in the client code during the *make_discover* and *state_reboot* procedures, which are invoked every time a client requests a configuration.

The modifications performed on the ISC code were done in such a way that the protocol specification is not altered. This results in a stealthier channel which is fully compliant with any DHCP client or server. In particular, the modified clients are able to request for network configuration parameters while transmitting hidden information to the servers and also they might behave as usual clients without conveying any information at all. Consequently, this choice must be notified to the modified server in order to be able to interpret the *xid* as data. In order to help the server in identifying which client is sending covert data, *start* and *end* delimiters are used (codified within the *xid* field). These delimiters are predefined but they can be modified in order to introduce some uncertainty to potential network traffic analyzers.

Upon the reception of a packet coming from a colluding client and containing the *start* delimiter, a new covert session starts. The received data is stored locally until the reception of the *end* delimiter. The data contained in the *xid* field is retrieved from DHCP Requests since this is a common message in both exchange models (see Fig. 11).

Furthermore, in order to enhance client-server interaction we introduce some mechanisms to allow these entities to determine if they are communicating with the right entity, since several clients and servers might coexist in the same network and sending covert data to unmodified servers is not only useless but also might increase the detection ratio. Therefore, if the client does not receive a reply from the expected server, then it will perform as an ordinary client.

4.2 Sname and File Implementation

Fields *sname* and *file* present very similar features. According to the protocol specification, these two fields are both defined as null-terminated strings and might only carry data in the case of Discover, Inform and Request packets coming from the client side, and Offer and Ack packets coming from the server side. This is taken into consideration in order to avoid altering the normal operation of the protocol.

The strategy in this case is to pretend to be sending empty fields by setting the first byte to the *null* character. Consequently, the implementation is able to hide a maximum of 190 bytes of data per packet, from which 63 bytes are kept within the *sname* field and 127 bytes within the *file* field. In this implementation Discover and Request packets are used as data carriers. This was not possible in the previous implementation because the *xid* field must be constant for a complete transaction. In Fig. 3 we show a snapshot of a modified server simultaneously receiving data from two modified clients.

```

debian@server: ~/Escritorio
Archivo Editor Ver Terminal Ayuda
debsquid@esquid:~$ sudo dhcpcd -d -cc cfile
Internet Systems Consortium DHCP Server 4.1.1-P1
Copyright 2004-2010 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/
Wrote 3 leases to leases file.
Listening on LPF/eth0/00:0c:29:93:37:da/172.16.232.0/24
Sending on LPF/eth0/00:0c:29:93:37:da/172.16.232.0/24
Sending on Socket/fallback/fallback-net
DHCPDISCOVER from 00:0c:29:8b:f2:46 via eth0

Received start of transmission
DHCPPOFFER on 172.16.232.128 to 00:0c:29:8b:f2:46 (onieva.uma) via eth0
DHCPREQUEST for 172.16.232.128 (172.16.232.129) from 00:0c:29:8b:f2:46 (onieva.uma) via eth0
DHCPACK on 172.16.232.128 to 00:0c:29:8b:f2:46 (onieva.uma) via eth0
DHCPDISCOVER from 00:50:56:22:09:d3 via eth0
DHCPPOFFER on 172.16.232.130 to 00:50:56:22:09:d3 (ruben.uma) via eth0

Received start of transmission
Received end of transmission
DHCPREQUEST for 172.16.232.130 (172.16.232.129) from 00:50:56:22:09:d3 (ruben.uma) via eth0
DHCPACK on 172.16.232.130 to 00:50:56:22:09:d3 (ruben.uma) via eth0
DHCPREQUEST for 172.16.232.130 from 00:50:56:22:09:d3 (ruben.uma) via eth0
DHCPACK on 172.16.232.130 to 00:50:56:22:09:d3 via eth0
DHCPREQUEST for 172.16.232.128 from 00:0c:29:8b:f2:46 (onieva.uma) via eth0
DHCPACK on 172.16.232.128 to 00:0c:29:8b:f2:46 (onieva.uma) via eth0

Received end of transmission
DHCPREQUEST for 172.16.232.130 from 00:50:56:22:09:d3 via eth0
DHCPACK on 172.16.232.130 to 00:50:56:22:09:d3 via eth0
DHCPREQUEST for 172.16.232.130 from 00:50:56:22:09:d3 via eth0
DHCPACK on 172.16.232.130 to 00:50:56:22:09:d3 via eth0

```

Fig. 3. Simultaneous covert data reception

In this implementation we also make use of delimiters because some implementations do not set to *null* these fields but they insert garbage data instead. The use of delimiters prevent the modified server from storing undesired data.

4.3 Options Implementation

From the various data hiding opportunities presented in Section 3.2 for the Options field we decided to use the options for private use. The main reason is that these might provide a substantial bandwidth.

The modified client injects its data in the payload of an option for private use³. In this case we selected option 224 and thus the use of delimiters is not required

³ We noticed that some options for private use are being used in an unofficial way by many DHCP servers. A commonly used option is 252 (Proxy Autodiscovery), which is used for indicating the location of the proxy server configuration parameters.

here because this option is not used otherwise. In case the server observes option 224 in the message coming from an colluding client, it stores the information.

Also, since the main advantage of this implementation is the bandwidth, we decided to include up to 255 bytes of covert data. In case the maximum capacity of a message is reached, the Overload option is used and the remaining options are included within the *sname* and *file* fields, if they are not being used.

5 Covert Channel Analysis

The covert implementations presented in Section 4 have different features that allows the user to decide which method to use depending on the circumstances. There is no best covert channel under all circumstances, and usually those who operate well in certain scenarios cease to be useful in others. Three properties stand out as the most important when dealing with covert channels: detectability, bandwidth and reliability. In our implementation, all the methods present full reliability, mainly due to the basic recovery mechanisms provided by DHCP against packet loss. However, the remaining properties differ between the three implementations.

The bandwidth is usually at odds with the ability to pass unnoticed and our solutions also present this trade-off. In terms of detectability, the three proposed methods have the advantage of being the first to use DHCP as a data carrier and moreover they strictly follow the protocol specification. Still, a smart observer might be alerted by some unusual patterns. In particular, the *xid* implementation is even stealthier because the original code fills the *xid* field with random data but the main downside of this method is that it provides a very limited bandwidth. Thus, this covert channel is recommended for small amounts of highly sensitive data, for example the transmission of an elliptic curve cryptography key.

The *sname* and *file* implementation provides a better bandwidth with a maximum capacity of 380 bytes in a full transaction (recall we hide data in Discover and Request packets). This result is significantly higher than the channel provided by the *xid* implementation (4 bytes). The main drawback is increased detectability. Even when both modified clients and servers use some techniques to prevent revealing the presence of the channel, there are some features that might arise suspicion on an experienced observer: in many implementations these fields are filled with zeros when not used. Besides, in order to allow modified clients to contact modified servers, their names are included in the *sname* field in every message coming from the server. By only making use of the *file* field we could create a bidirectional covert channel between clients a servers.

Finally, the *options* implementation provides the higher bandwidth and is recommended for loosely supervised networks because the size of the resulting DHCP messages might attract the attention of casual observers. This method would also allow the creation of a two-way covert channel because option 224 is not used for any other purposes. Also, the detectability of the channel might be improved by reducing the size of the option payload or by using different options to convey the data.

6 Conclusions

This paper presents an exhaustive analysis on the potential for covert communications in DHCP, a protocol that is extensively used for the configuration of IP-based devices. From this analysis we select and implement three storage channels that we integrate on a tool called HIDE_DHCP. These solutions present distinguishing features that makes them suitable under different circumstances. The main factors influencing the selection among the devised methods are the channel bandwidth and its detectability, which is made difficult by the fact that no previous implementations on DHCP exist and also because the normal operation of the protocol is not altered by the subliminal channel.

Several information hiding techniques have been discussed for DHCP but only a few of them have been implemented. We have further analyzed the advantages and limitations as well as possible means of improvement. Our current work goes in this direction. Also, performing more exhaustive detectability tests is a natural next step. Although we have performed some successful detectability tests with out-of-the-box intrusion detection solutions, much work can be done in this direction. Finally, our tests on a controlled LAN environment showed full reliability in terms of packet loss. However, running tests under extreme circumstances is also of interest.

Acknowledgments. This work has been partially funded by the European Commission through the FP7 project NESSoS (FP7 256890) and the Spanish Ministry of Science and Innovation through the research projects: ARES (CSD2007-00004) and SACO (IPT-2011-1593-390000). The first author is supported by the Spanish Ministry of Education through the F.P.U. Program.

References

1. Brand, S.L.: Department of Defense Trusted Computer System Evaluation Criteria, "The Orange Book". Tech. Rep. DoD 5200.28-STD, U.S. Department of Defense (1985), <http://csrc.nist.gov/publications/history/dod85.pdf>
2. Cabuk, S., Brodley, C.E., Shields, C.: IP Covert Timing Channels: Design and Detection. In: Proceedings of the 11th ACM Conference on Computer and Communications Security, CCS 2004, pp. 178–187. ACM Press, New York (2004)
3. Cabuk, S., Brodley, C.E., Shields, C.: IP Covert Channel Detection. ACM Trans. Inf. Syst. Secur. 12, 22:1–22:29 (2009)
4. Daemon9: Loki2 (the implementation) (1997), <http://www.phrack.org/archives/51/P51-06>
5. Droms, R.: RFC 2131 - Dynamic Host Configuration Protocols (1997), <http://www.ietf.org/rfc/rfc2131.txt>
6. Dyatlov, A.: Firepass - Gray-World.net Team (2003), http://gray-world.net/it/pr_firepass.shtml
7. Gianvecchio, S., Wang, H.: An entropy-based approach to detecting covert timing channels. IEEE Trans. Dependable Secur. Comput. 8, 785–797 (2011)
8. Giffin, J., Greenstadt, R., Litwack, P., Tibbetts, R.: Covert Messaging through TCP Timestamps. In: Dingledine, R., Syverson, P.F. (eds.) PET 2002. LNCS, vol. 2482, pp. 194–208. Springer, Heidelberg (2003)

9. Girling, C.G.: Covert Channels in LAN's. *IEEE Trans. Software Eng.* 13(2), 292–296 (1987)
10. Gligor, V.D.: A Guide to Understanding Covert Channel Analysis of Trusted Systems, “The Light Pink Book”. Tech. Rep. NCSC-TG-030, U.S. National Computer Security Center (1993)
11. Handel, T.G., Sandford, M.T.: Hiding Data in the OSI Network Model. In: Anderson, R. (ed.) *IH 1996*. LNCS, vol. 1174, pp. 23–38. Springer, Heidelberg (1996)
12. ISC: DHCP - Internet Systems Consortium, Inc. (2011), <http://www.isc.org/software/dhcp/>
13. Kaminsky, D.: Tunneling Audio, Video, SSH and pretty much anything else over DNS (2004), <http://www.doxpara.com/>
14. Lampson, B.W.: A Note on the Confinement Problem. *Commun. ACM* 16(10), 613–615 (1973)
15. Li, S., Ephremides, A.: Covert channels in ad-hoc wireless networks. *Ad Hoc Netw.* 8, 135–147 (2010)
16. Lucena, N.B., Lewandowski, G., Chapin, S.J.: Covert Channels in IPv6. In: Danezis, G., Martin, D. (eds.) *PET 2005*. LNCS, vol. 3856, pp. 147–166. Springer, Heidelberg (2006)
17. Luo, X., Chan, E.W.W., Chang, R.K.C.: Cloak: A Ten-Fold Way for Reliable Covert Communications. In: Biskup, J., Lopez, J. (eds.) *ESORICS 2007*. LNCS, vol. 4734, pp. 283–298. Springer, Heidelberg (2007)
18. McHugh, J.: Covert Channels Analysis: A Chapter of the Handbook for the Computer Security Certification of Trusted Systems. Technical Memorandum 5540:062A, Naval Research Laboratory, Washington, D.C. (1996), <http://www.windowsecurity.com/uplarticle/12/COVCHAN.pdf>
19. Rowland, C.H.: Covert Channels in the TCP/IP protocol suite (1996), http://www.firstmonday.org/issues/issue2_5/rowland/
20. Shah, G., Molina, A., Blaze, M.: Keyboards and Covert Channels. In: *USENIX-SS 2006: Proceedings of the 15th Conference on USENIX Security Symposium*, pp. 59–75. USENIX Association, Berkeley (2006)
21. Stødle, D.: Ping Tunnel - Send TCP traffic over ICMP (2005), <http://www.cs.uit.no/~daniels/PingTunnel/>
22. Wolf, M.: Covert Channels in LAN Protocols. In: Berson, T.A., Beth, T. (eds.) *LANSEC 1989*. LNCS, vol. 396, pp. 91–101. Springer, Heidelberg (1989)

Handling Stateful Firewall Anomalies

Frédéric Cuppens¹, Nora Cuppens-Boulahia^{1,2}, Joaquin Garcia-Alfaro¹,
Tarik Moataz¹, and Xavier Rimasson¹

¹ Institut Télécom, Télécom Bretagne,
CS 17607, 35576 Cesson-Sévigné, France

forename.surname@telecom-bretagne.eu

² Swid Web Performance Service
Rennes, France

Abstract. A security policy consists of a set of rules designed to protect an information system. To ensure this protection, the rules must be deployed on security components in a consistent and non-redundant manner. Unfortunately, an empirical approach is often adopted by network administrators, to the detriment of theoretical validation. While the literature on the analysis of configurations of first generation (stateless) firewalls is now rich, this is not the case for second and third generation firewalls, also known as stateful firewalls. In this paper, we address this limitation, and provide solutions to analyze and handle stateful firewall anomalies and misconfiguration.

1 Introduction

Firewalls aim at optimizing the degree of security deployed over an information system. Their configuration is, however, very complex and error-prone. It is based on the distribution of several packages of security rules that define properties such as acceptance and rejection of traffic. The assembly of all these properties must be consistent, addressing always the same decisions under equivalent conditions, and avoiding conflicts or redundancies. Otherwise, the existence of anomalies and misconfiguration will lead to weak security architectures, potentially easy to be evaded by unauthorized parties. Approaches based on formal refinement techniques, e.g., using abstract machines grounded on the use of set theory and first order logic, ensures, by construction, cohesion, completeness and optimal deployment [1]. Unfortunately, these approaches have not always a wide follow. Network policies are often empirically deployed over firewalls based on security administrator expertise and flair. It is then relevant to analyze these deployed configurations in order to detect and correct errors, known in the literature as configuration anomaly discovery. Several research works exist to directly manage the discovery and correction of *stateless* firewall configuration anomalies [2,3,4,5]. By stateless firewall configurations we refer to the security policies of first generation firewalls, mostly packet filtering devices working only on the lower layers of the OSI reference model. However, little work has been done to address the case of second and third generation firewalls peeking into the transport and upper layers. In this paper, we are particularly interested in addressing such a problem.

The main goal of a firewall is to control network traffic flowing across different areas of a given local network. It must provide either hardware or software means to block

unwanted traffic, or to re-route packets towards other components for further analysis. First generation firewalls only allow a *stateless* filtering of network traffic. The filtering actions, such as accepting or rejecting packet flows, are taken according to a set of static configuration rules that only pay attention to information contained in the packet itself, such as packet's addresses (source and destination), ports and protocol. Their main advantage is the speed of filtering operations. However, since they do not keep track of state connection data, they fail at handling some vulnerabilities that benefit from the position of a packet within existing streams of traffic. Stateful firewalls solve this problem and improve packet filtering by keeping track of connection status. Indeed, they can block those packages that are not meeting the state machine of a protocol over the transport layer. As with stateless packet filtering, stateful filtering intercepts the packets at the network layer and verifies if they match previously defined security rules. Moreover, stateful firewalls keep track of each connection in an internal state table. Although the entries in this table varies according to the manufacturer of every product, they typically include source and destination IP addresses, port numbers and information about the connection status.

Most methods that have been proposed to detect anomalies in the configuration of firewalls, such as [2][3][4][5], are limited to the stateless case. The detection of anomalies in stateful firewalls is still an unexplored research problem. Existing literature is still very limited. Some approaches aim at describing stateful firewall models [7], while others simply provide straightforward adaptations of management processes previously designed for stateless firewalls [8]. In this paper, we propose to extend the algorithms defined in [2] to include the management of stateful firewall anomalies as well. The principle of our approach is based on the specification of general automata. Such automata describe the different states that traffic packages can take throughout the filtering process. We then extend the taxonomy of anomalies defined in [2] in order to uncover new configuration anomalies for the case of stateful firewalls. Some anomalies occur on rule sets which only contain stateful rules, denoted hereinafter as *intra-state rule anomalies*. Other anomalies may affect those rule sets holding both stateful and stateless rules, denoted in our work as *inter-state rule anomalies*. We define algorithmic solutions to automatically detect and correct these new types of anomalies. The algorithms that we present also provide an extension of MIRAGE [9], a firewall audit tool implemented in the Java language, that allows the automatic detection and correction of stateless firewall configuration anomalies. The extension aims at covering the management of stateful firewalls as well.

The remainder of the paper is organized as follows. Section 2 presents in more detail our motivation and problem domain. Section 3 surveys related work. Section 4 presents a classification of stateful firewall intra-state rule anomalies, and defines an algorithmic solution to handle them. Section 5 extends the approach to the case of inter-state rule anomalies. Section 6 concludes the paper.

2 Motivation

Nowadays, packet filtering requires more than a passive solution to stop malicious traffic. Filtering is evolving into a dynamic process that depends on the protocol states.

Consequently, new policies tend to use more and more the stateful features of next generation firewalls. Stateless inspection consists only on a static verification of the five first flag attributes of the IP (internet protocol) layer with an existing ACL (Access Control List) table. Stateful inspection goes further, and provides the firewall with the means to analyze packets not only by using the matching attribute correspondence, but also by linking each packet to the related connection. Finally, stateful firewalls provide better fine-grained filtering capabilities, and protect against more complex attacks, such as denial of service (DOS) and IP Spoofing. The main disadvantages of stateful filtering are a more complex implementation and greater use of memory to keep the trace of the previous sessions.

In our work, we consider stateful filtering for connection-oriented protocols, such as TCP, DCCP, ATM, Frame Relay, TIPC, SCTP, and IPX/SPX. All these protocols are based on a common principle: sending a tagged stream belonging to a particular session. In the case of TCP and SCTP, this is respectively called a session connection and an association. A stateful inspection can be performed on any of these protocols by defining a protocol automaton. In the literature, there is no formal model that describes a general behavior of firewalls in states with all specifications required and coverage of all protocols. Gouda and Liu [7] proposed in their work a modeling of a state inspection. This inspection integrates a combination of the *state* phase detection then the *free state* one. They proposed the addition of the attribute *state* depending on the protocol used for storing packets belonging to the connection initialization, and one *tag* to check the membership of new packets for this connection. In this paper the attribute *state* is for a state protocol itself, which does not necessarily imply the initialization of a connection-oriented session (e.g., the ICMP protocol can be modeled by the states resulting from the exchange of messages *echo-request* and *echo-reply*, even if the protocol is not connection-oriented). This approach corresponds to the model proposed for stateful firewalls, except that we can give other models satisfying the same functionality as the simultaneous approach explained above, based on stateful and stateless rules at the same time. Specifically, instead of proposing a filtering series through *state*, then the attributes presented in the stateless model, we consider a model of the filter including rules for states and stateless in parallel. Regardless of the model, the purpose of firewalls is to filter states packets according to the membership to a given session. Thus, according to the definition of *session*, modeling and representation of rules in the tables, the specification and identification of anomalies is different. In this paper, we focus, initially, on detecting anomalies on those configurations that contain only stateful rules. Specifically, we define and analyze new types of anomalies depending on the protocol transitions. The protocol chosen to illustrate the approach is TCP as a connection-oriented protocol reference. Then, we extend the approach in order to address the case in which stateful and stateless rules coexist in a given configuration rule set.

3 Related Work

Traditional research work on the design of firewalls, essentially stateless firewalls, mainly address the construction of high level languages for the specification of firewall configurations. This includes functional languages [10], rule-based languages [11]

and higher abstract models that allow capturing some further aspects such as network topologies [12]. Such languages allow security administrators to free themselves from the technical complexity and specificity of proprietary firewalls languages. Some of them allow, moreover, the automatic derivation of concrete access control rules to configure specific firewalls through a translation process. At the same time, research and development work in this context may allow the verification of consistency (i.e., absence of conflicts), completeness (all the expected requirements are covered), and compactness (none of the rules are redundant or unnecessary) [1]. Refinement approaches may also take into account the functionality offered by every firewall manufacturer (stateless, stateful, management of virtual private networking, etc.) [13] to ensure the effective distribution of tasks between a decision module and the eventual filtering (enforcement) components.

For the already deployed firewall configurations, the aforementioned approaches do not solve redundancy or configuration conflicts that might have been introduced due to periodic, often manual, updates. Several studies have been conducted toward audit mechanisms that analyze already deployed configurations, with the goal of signaling inconsistencies and fixing the discovered anomalies. We can classify them into three categories: (I) those that are oriented towards directly querying the firewall itself [14][15][16], (II) those targeting conflict management [17][18] and (III) those focusing on the detection of anomalies [19][4][20][21]. In category I, the analysis problem is relayed towards a process of information retrieval by directly querying the firewall. This requires having highly structured configurations and specific query languages for processing them, as well as for generating complete and effective data queries. The results are, moreover, prone to both false negatives and false positives, since no track from previous filtering matches are taken into account during the audit process. Category II is concerned with packet classification algorithms, mostly for packet filtering routers, and that rely on optimized data structures to speed up the matching process between incoming flows of packets and filtering rules. Then, the goal is to verify that there are no conflicting situations in which several rules with different actions (e.g., accept or reject the traffic) apply to the same traffic. Examples in this category include the use of techniques such as *grid-of-tries* classification [22] and *bit vector aggregation* [17]. Class III improves the detection offered by solutions in class II, by: (1) characterizing in more detail the set of anomalies, e.g., redundancy is also addressed; (2) transforming the rule sets in such a way that the ordering of rules is no longer relevant; (3) considering combinations of rules instead of simply comparing rules two by two as proposed by Al-Shaer *et al.* [19], which enables the detection of a combination of rules that conflict with another rule [2]; and (4) extending the process to distributed setups with multi-firewall scenarios, in order to detect situations in which different firewalls within interconnected paths may perform different actions to the same network traffic.

None of the above surveyed techniques consider the case of stateful firewalls. So far, little work in the stateful case has been conducted. Buttyán *et al.* proposed in [8] an early approach that heads towards this research line. Nevertheless, their solution is limited to the adaptation of existing anomaly detection techniques for stateless firewalls to those that are stateful. Therefore, their work does not take into account anomalies that may impact, for instance, the tracking of connections or the management of the

internal firewall state memory table. In a different vein, Fitzgerald *et al.* propose in [23] an approach based on semantic web technologies to model both stateless and stateful firewalls. Although the generality of their proposed representation is interesting enough, the work fails at characterizing the precise types of errors that would be necessary to handle by the detection process in the stateful case. The approach only represents those good practices that must be followed when configuring a given firewall.

4 Handling Anomalies on Stateful Firewall Intra-State Rules

Like in the case of stateless firewalls, stateful firewall configurations may be affected by the following basic anomalies (cf. algorithmic solutions in [2] and citations thereof, for the discovery and correction of these two anomalies):

- *Shadowing*: Let R be a set of filtering rules. Then, rule R_i is shadowed iff it never applies because all the packets that R_i may match, are previously matched by another rule, or combination of rules, with higher priority in order;
- *Redundancy*: Let R be a set of filtering rules. Then, rule R_i is redundant iff (1) R_i is not shadowed by any other rule (or combination of rules); and (2) when removing R_i from R , the filtering result does not change.

In the stateful case, we can also identify a third type of anomalies, hereinafter denoted as *intra-state protocol anomalies*, in the sense of misconfiguration that may put in risk the inner logic of transport layer protocol states. For instance, in protocols like TCP, we may distinguish the following operations for the establishment of a connection between a server and a client:

- the client sends a SYN packet to a server (i.e., a packet with the SYN flag set);
- the server replies with a SYN+ACK;
- the client sends an ACK back to the server.

Then, it comes a phase of data transfer. Finally, the connection ends with a termination phase. When the client leads the termination phase, the following handshake takes place (still, assuming the case of TCP):

- the client sends a FIN packet to a server (i.e., a packet with the FIN flag set);
- the server replies with an ACK, then with a FIN;
- the client sends an ACK back to the server.

Given such a rationale, the following two anomaly scenarios arise:

1. the client succeeds to start the three-way handshake connection establishment with a server, while the firewall is configured in a way that either the second or third steps of the establishment operations are rejected;
2. the client starts the connection termination, but the firewall rejects, at least, one of the remainder termination operations;

In the sequel, we present an algorithmic solution to handle and correct this third type of anomalies that affects the establishment and termination of transport layer connections.

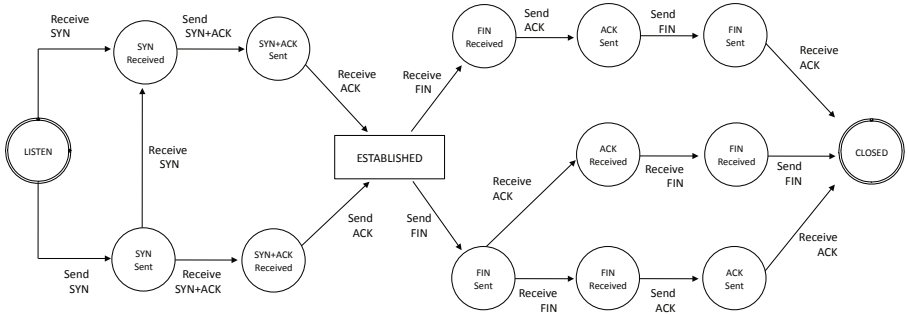


Fig. 1. TCP state automaton

4.1 Discovery and Elimination of Intra-state Protocol Anomalies

Definitions: The communication between two nodes with a transport layer protocol consists of (1) establishment phase and (2) termination phase. We model such a protocol as two deterministic finite-state automata that correspond, respectively, to the establishment (A_1) and termination phase (A_2). We characterize these two automata by giving some examples for TCP:

- Σ is the alphabet of the automaton, which contains the series of state *flags* of a protocol. For instance, for TCP, $\Sigma = \{SYN, SYN + ACK, ACK, FIN, FIN + ACK\}$;
- Q contains the series of states of the protocols, such as the CLOSED, LISTEN, and SYN SENT states in the case of TCP;
- δ defines the transition function, such that $\delta : Q \times \Sigma \rightarrow Q$;
- q_0 defines the initial state. For TCP, this includes LISTEN for A_1 and ESTABLISHED for A_2 .
- q_f defines the final state. For TCP, A_1 equals ESTABLISHED; and A_2 equals CLOSED.

Figure 1 depicts the state automaton of TCP. It includes both A_1 and A_2 , which share the ESTABLISHED state.

The following functions are used for the construction of our algorithms:

- $Adjacent(A, State_i, State_j)$: boolean function that holds true iff there exists within automaton A a transition from $State_i$ to $State_j$. Equivalently, the expression is true iff there exists $a \in \Sigma$ such that $\delta(State_i, a) = State_j$
- $Next(A, State, Symbol)$: if $\delta(State, Symbol) = State'$ exists, the function returns $State'$. Otherwise, it returns the empty set.

Finally, we consider a rule set R of size n containing R_i rules ($i \leq n$). Each rule R_i typically specifies an *Action* (e.g., ACCEPT or DENY) that applies to a set of condition attributes, such as *SourceAddr*, *DestAddr*, *SrcPort*, *DestPort*, *Protocol*, *Flag*, *State*.

Algorithms: We assume a rule subset R whose protocol ($R_i[protocol]$ attribute) stands for a given protocol. Our algorithm uses the establishment (A_1) and termination (A_2) automata of the protocol.

If an ordered pair of rules stands for two adjacent states of the automaton, we check if the related flags can be used in order to switch from the first state to the second. If so, and if the actions of the two rules are different, then we raise an anomaly. First, we apply the algorithm for anomaly corrections on the rules which are related to the establishment of transport layer connection; secondly we do it for the termination. In order to correct the anomalies of establishment connection, we change the action of the rules to DENY. Therefore we avoid connection failures. Regarding the anomalies of termination connection, we set the action of the rules to ACCEPT in order to avoid termination failure. Algorithm 1 sums up the corrections; it uses Algorithm 2 which analyzes the rules, detects the anomalies and then carries out the corrections.

Figure 2 gives an example of a correction on a subset of stateful rules by using Algorithm 1. In the initial configuration of the example, a first rule allows packets with the SYN flag and the LISTEN state. In accordance with the automaton of TCP on [1], the rule actually allows the first step of a TCP connection establishment (i.e. the shift

Algorithm 1. HandleAnomalies(R, A_1, A_2)

```

/*Handle, first, the connection
  establishment automaton */
R ← HandleRules(R, A1, DENY)
/*Then, handle the connection
  termination automaton */
R ← HandleRules(R, A2, ACCEPT)
return R

```

Algorithm 2. HandleRules($R, A, Action$)

```

n ← size(R);
for i ← 1 to n - 1 do
  for j ← i + 1 to n do
    if  $R_i[SourceAddr] = R_j[DestAddr]$ 
       $\wedge R_i[SrcPort] = R_j[DestPort]$ 
       $\wedge R_i[Action] \neq R_j[Action]$  then
      if  $Adjacent(A, R_i[State], R_j[State])$  then
        if  $(R_j[State] \in Next(A, R_i[State],$ 
           $R_i[Flag]) \wedge Next(A, R_j[State], R_j[Flag])$ 
           $\neq \emptyset) \vee (R_i[State] \in Next(A, R_j[State],$ 
           $R_j[Flag]) \wedge Next(A, R_i[State], R_i[Flag])$ 
           $\neq \emptyset)$  then
           $R_i[Action] \leftarrow Action$ 
           $R_j[Action] \leftarrow Action$ 
        end
      end
    end
  end
end
return R

```

from LISTEN state to the SYN Received state). However the second rule forbids the acknowledgment. Then the connection cannot be established. In that case, the algorithm corrects the first rule in order to deny the connection.

<i>SourceAddr</i>	<i>DestAddr</i>	<i>SrcPort</i>	<i>DestPort</i>	<i>Protocol</i>	<i>Flag</i>	<i>State</i>	<i>Action</i>
41.1.1.1	193.1.1.2	8080	2011	TCP	SYN	LISTEN	ACCEPT DENY
193.1.1.2	41.1.1.1	2011	8080	TCP	SYN+ACK	SYN RCVD	DENY

Fig. 2. Example of an intra-state protocol anomaly in the rule set of a stateful firewall

5 Handling Inter-state Rule Anomalies

We have previously addressed the case of intra-state rule anomalies, in which a set of stateful rules, at the transport layer, contains anomalies that may put in risk the inner logic of transport layer protocol states. The use of both stateful and stateless rules may be also found in a firewall configuration. For instance, a network administrator may add a rule in order to handle TCP connections which are used to transferring data in a FTP session. For instance, for a Netfilter firewall, we can manage this situation by adding a rule with the RELATED state parameter. This rule will be inserted in the other stateless rules which have been previously defined by the administrator. We then search for anomalies between stateful and stateless rules based on the specification of a transport layer protocol. Some application layer protocols use several TCP connections (or other transport layer protocol) during a session between two nodes. This applies for FTP, IRC or VoIP protocols which use related connections if needed. Let us further analyze the case of FTP. A typical FTP session consists in two steps:

1. The client begins the session with the FTP server on port 21; a TCP connection – the control connection – is established;
2. When the client wants to transfer data (file transfer, directory listing, etc.), two cases may occur:
 - After a control connection negotiation, the server initiates a new TCP connection for the transfer, from the port 20 to a client’s given port. This is the active mode.
 - Or the transfer connection is initiated by the client to a FTP server’s given port. This is the passive mode.

The FTP server’s firewall configuration may contain:

- A stateless rule for allowing TCP packets with the destination port 21;
- A stateful rule for allowing packets whose associated TCP connection is marked with a related connection in a FTP session. The destination port will be either 20 (active mode) or greater than 1024 (passive mode).

In this example, one issue consists in correctly handling the related TCP connections between two nodes which use an application layer protocol. We note the firewall shall understand the given application layer protocol which is concerned by the rules in order

to identify related connection packets. Netfilter especially has a module which allows FTP sessions tracking. Therefore a packet which belongs to a TCP transfer connection (according to the FTP terminology) has the RELATED status. This tracking of the application layer context allows the administrator to define stateful rules.

To automatically identify such anomalies for a given protocol, we assume knowing a full specification of possible scenarios of TCP connection used between two nodes during a session for the protocol. This specification explains how to initiate the connection and how it deals with the related connections during the session (order, number, ports, etc.).

The first step consists in searching the stateless rules which stand for the establishment of the protocol connection. In the case of FTP, we search a rule which matches the TCP packets with the destination port 21. If such rules are found, we consider the three following cases:

1. Stateful rules exist in the configuration to handle the possible related connections that may be used by the application layer protocol;
2. Stateless rules exist to handle these connexions;
3. No rule is defined to handle the related connexions.

The case 2 is too general because it does not take into account the inner logic of the protocol. An attacker may be able to initiate a TCP connection on a port which will be used only for a related connection of an application session. For example, a FTP connection on the server's port p will be allowed only if the server has previously initiated a FTP transfer on passive mode with a client on the port p . In the case 3, the application session may fail because the firewall will probably deny the related connections. The case 1 solves the encountered problem with the other ones and complies with the protocol specification. In a Netfilter firewall, such rules may have the RELATED state.

Definitions: Our algorithm aims at assisting the system administrator to detect and fix cases 2 and 3 of the aforementioned anomaly. We first provide the following definitions that will be used in the algorithm definition:

- L : set of stateless rules, such that every rule L_i (where i is a natural integer) is characterized by the following conditions $L_i[SourceAddr]$, $L_i[DestAddr]$, $L_i[SrcPort]$, $L_i[DestPort]$ and $L_i[Protocol]$ (such as TCP, UDP, or any other transport layer protocol).
- F : set of stateful rules, such that every rule F_i is characterized by the same conditions as the rules in L , plus the condition attribute $F_i[State]$. It is important to consider $F_i[Protocol]$ since the transport protocol of a given connection could be different from the protocol of the main connection. For instance, in VoIP scenarios, data transfer might be carried upon UDP, while the main connection is relayed via TCP.
- A : deterministic finite automaton that describes an application layer protocol. We rely on the use of the alphabet Σ of A , containing the set of operations that can be exchanged between hosts, e.g., remainder set of operations once the main connection of two FTP entities has been established. Q is the set of states, from which we identify the subset Q_2 . The elements q of Q_2 represent establishment of adjacent

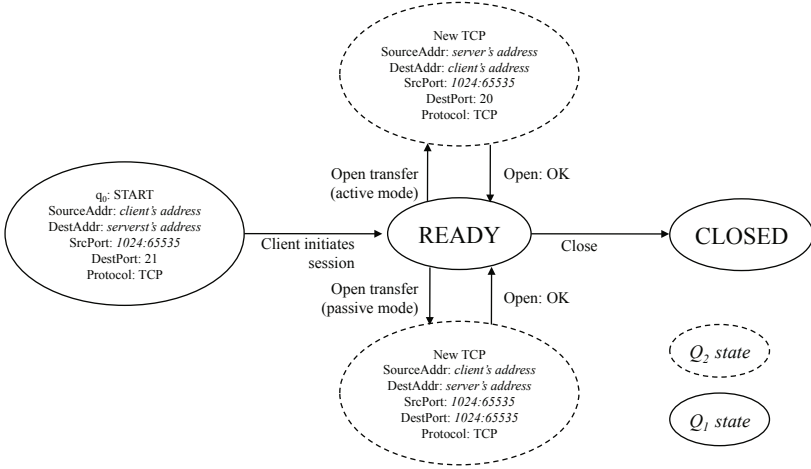


Fig. 3. Suggested automaton for the application layer protocol FTP

connections (such as TCP connections or from any other protocol type). The elements are characterized by the same set of conditions as the one in the rules (i.e., $q[SrcAddr]$, $q[DestAddr]$, etc.). Let us observe that $q[State]$ will highly rely on the specific firewall vendor (cf. following function definition, in which we define the way to link the specific state attribute of the automaton to the corresponding firewall device). Notice that $R_i[State]$ (i.e., the state defined in a given rule R_i) corresponds to the specific state as it is represented by the underlying firewall that contains the rule, not the state attribute of the automaton. In Netfilter, for instance, relevant connections are identified by the state attribute RELATED. If necessary, we can rely on extended features of Netfilter to provide a more fine-grained state management of some application layer protocols. $Q_1 = Q - Q_2$ contains the set of states that are independent from related connections, and for which the element $q[State]$ is not defined. Finally, the initial state q_0 of the automaton holds the following condition attributes: $q_0[SrcPort]$, $q_0[DestPort]$ and $q_0[Protocol]$ (corresponding to the transport layer protocol). Figure 3 depicts the example of an automaton based on our construction, for the FTP protocol.

- $stateFirewall(q)$: function that links a given state $q \in Q_2$ of the corresponding state automaton to the firewall. For instance, in the case of the FTP protocol and a firewall based on Netfilter, this function returns RELATED for those states in which the establishment of connections is called.
- $ruleExists(R, q)$: boolean function. R is a set of either stateless or stateful rules (but not both), q represents a state of the automaton A which belongs to Q_2 (the state corresponding to the establishment of related connections). If R contains stateless rules, then $ruleExists(R, q)$ is true iff there exists exactly one rule $R_i \in R$, such that $q[SourceAddr] \in R_i[SourceAddr]$, $q[DestAddr] \in R_i[DestAddr]$, $q[SrcPort] \in R_i[SrcPort]$, $q[DestPort] \in R_i[DestPort]$, and $q[protocol] = R_i[protocol]$. If R contains stateful rules, then $ruleExists(R, q)$ is true iff the previous conditions also hold and, moreover, $stateFirewall(q[State]) = R_i[State]$.

- *ruleExists*(L, q_0): boolean function. q_0 contains the initial state of the protocol, and L is a set of stateless rules. The function is true iff there exists a rule $L_i \in L$, such that $q_0[SrcPort] \in L_i[SrcPort]$, $q_0[DestPort] \in L_i[DestPort]$, $q_0[Protocol] = L_i[Protocol]$.

Algorithms: Algorithm 3 enables the verification of every state Q_2 of an automaton associated with a given protocol, in order to find rules that can be correlated. The algorithm specifies the appropriate corrections in accordance to the detection of anomalies, and following the three cases mentioned above (absence of rules, or misconfigured stateless or stateful rules). $A[Q_2]$ points out to the Q_2 set of the automaton.

Algorithm 4 allows detection and correction of anomalies between stateless and stateful rules, provided that a library of application layer protocols is given as input. Such a library must contain the corresponding automata for the protocols. Then, it verifies whether the firewall handles each of them, by looking at the initial state attribute q_0 of the corresponding automaton. In such a case, Algorithm 3 processes the specific anomalies associated with that protocol. $A[q_0]$ points out the initial state q_0 of every automaton.

The following example presents an extract from a Netfilter-based configuration. We can look at three rules that aim at granting authorization to FTP services, both in active and passive mode:

Notice that the sample contains two inter-state anomalies. Rule R_1 is a stateless authorization to control incoming higher port TCP connections targeting a range of FTP servers listening on port 21. Then, rules R_2 are R_3 expected to grant authorization

Algorithm 3. HandleInterRuleAnomalies(L, F, A)

```

/*A[Q2]: Q2 states for automaton A */
forall  $q \in A[Q_2]$  do
  if ruleExists( $F, q$ ) then
    | /*Move to following state */
    | continue;
  end
  if ruleExists( $L, q$ ) then
    | warning("stateless rule for state  $q$  of protocol  $A$ ")
  else
    | warning("missing rule for state  $q$  of protocol  $A$ ")
  end
end

```

Algorithm 4. HandleAllProtocols($L, F, Library$)

```

/*Library: automata library, containing
the list of supported application-layer
protocols */
forall  $A \in Library$  do
  if ruleExists( $L, A[q_0]$ ) then
    | HandleInterRuleAnomalies( $L, F, A$ )
  end
end

```

```
R1: iptables -A FORWARD -s $ANY -d $SERVERS -sport 1024:65535 -dport 21 -j ACCEPT
...
R2: iptables -A FORWARD -d $ANY -s $SERVERS -dport 1024:65535 -sport 20 -j ACCEPT
R3: iptables -A FORWARD -s $ANY -d $SERVERS -sport 1024:65535 -dport 1024:65535 -j ACCEPT
```

access to the data connection counterpart, i.e., outgoing TCP connection from servers to clients. However, these last two rules are stateless. They grant access to any connection targeting a TCP high port (i.e., the whole range 1024:65535). If we apply Algorithm 4 to the previous configuration, it will detect such a situation and suggest the administrator to handle it (e.g., by adding the Netfilter parameter `--state RELATED` to both rules).

6 Conclusion

Stateful firewalls are the predominant solution to guarantee network security. They provide an effective enforcement of access control rules at both network and transport layers, in order to protect incoming and outgoing interaction with the Internet. Nevertheless, the existence of anomalies in their configuration is very likely to degrade such a protection. While some anomalies may occur in rule sets that only contain stateful rules (*intra-state rule anomalies*), others affect rule sets that contain both stateful and stateless rules (*inter-state rule anomalies*). In this paper, we have presented new types of anomalies for each of these two categories, and have provided algorithmic solutions to handle them. General automata describing the stateful nature of the filtering process drive the discovering and correction functionality of our solutions. Perspectives for further work include the extension of our solutions towards multi-firewall scenarios, to handle distributed policy control.

Acknowledgements. This research was partially supported by the European Commission, in the framework of the ITEA2 Predykot project (Grant agreement no. 10035).

References

1. Preda, S., Cuppens-Bouahia, N., Cuppens, F., Garcia-Alfaro, J., Toutain, L.: Model-Driven Security Policy Deployment: Property Oriented Approach. In: Massacci, F., Wallach, D., Zannone, N. (eds.) ESSoS 2010. LNCS, vol. 5965, pp. 123–139. Springer, Heidelberg (2010)
2. Garcia-Alfaro, J., Bouahia-Cuppens, N., Cuppens, F.: Complete analysis of configuration rules to guarantee reliable network security policies. *Int. J. Inf. Sec.* 7(2), 103–122 (2008)
3. Adishesu, H., Suri, S., Parulkar, G.: Detecting and Resolving Packet Filter Conflicts. In: INFOCOM, Tel Aviv, Israel, pp. 1203–1212 (2000)
4. Al-Shaer, E., Hamed, H.: Discovery of Policy Anomalies in Distributed Firewalls. In: INFOCOM, Hong Kong, China (2004)
5. Yuan, L., Mai, J., Su, Z., Chen, H., Chuah, C., Mohapatra, P.: FIREMAN: A Toolkit for FIREwall Modeling and ANalysis. In: IEEE Symposium on Security and Privacy, Berkeley, California, USA, pp. 199–213 (2006)

6. Cheswick, W., Bellovin, S., Rubin, A.: *Firewalls and Internet Security: Repelling the Wily Hacker*, 2nd edn. Addison-Wesley (2003)
7. Gouda, M., Liu, A.: A model of stateful firewalls and its properties. In: *DSN, Yokohama, Japan*, pp. 128–137 (2005)
8. Buttyan, L., Pék, G., Thong, T.V.: Consistency verification of stateful firewalls is not harder than the stateless case. *Infocommunications Journal LXIV* (2009)
9. Garcia-Alfaro, J., Cuppens, F., Cuppens-Boulahia, N., Preda, S.: *MIRAGE: A Management Tool for the Analysis and Deployment of Network Security Policies*. In: *DPM/SETOP, Athens, Greece*, pp. 203–215 (2010)
10. Guttman, J.: Filtering postures: Local enforcement for global policies. In: *Proceedings, 1997 IEEE Symposium on Security and Privacy*, pp. 120–129. IEEE Computer Society Press (1997)
11. Bartal, Y., Mayer, A., Nissim, K., Wool, A.: *Firmato: A novel firewall management toolkit* (1999)
12. Cuppens, F., Cuppens-Boulahia, N., Sans, T., Miège, A.: A formal approach to specify and deploy a network security policy. In: *Formal Aspects in Security and Trust*, pp. 203–218 (2004)
13. Preda, S., Cuppens, F., Cuppens-Boulahia, N., Garcia-Alfaro, J., Toutain, L.: Dynamic deployment of context-aware access control policies for constrained security devices. *Journal of Systems and Software* 84(7), 1144–1159 (2011)
14. Hazelhurst, S., Attar, A., Sinnappan, R.: Algorithms for improving the dependability of firewall and filter rule lists. In: *DSN*, pp. 576–585 (2000)
15. Liu, A.X., Gouda, M.G., Ma, H.H., Ngu, A.H.: *Firewall Queries*. In: Higashino, T. (ed.) *OPODIS 2004*. LNCS, vol. 3544, pp. 197–212. Springer, Heidelberg (2005)
16. Mayer, A., Wool, A., Ziskind, E.: *Fang: A firewall analysis engine*. In: *IEEE Symposium on Security and Privacy*, pp. 177–187 (2000)
17. Baboescu, F., Varghese, G.: Scalable packet classification. In: *ACM SIGCOMM*, pp. 199–210 (2001)
18. Eppstein, D., Muthukrishnan, S.: Internet packet filter management and rectangle geometry, pp. 827–835 (2001)
19. Al-Shaer, E., Hamed, H.: Firewall policy advisor for anomaly discovery and rule editing. In: *Integrated Network Management*, pp. 17–30 (2003)
20. Alfaro, J.G., Cuppens, F., Cuppens-Boulahia, N.: Analysis of Policy Anomalies on Distributed Network Security Setups. In: Gollmann, D., Meier, J., Sabelfeld, A. (eds.) *ESORICS 2006*. LNCS, vol. 4189, pp. 496–511. Springer, Heidelberg (2006)
21. Alfaro, J.G., Cuppens, F., Cuppens-Boulahia, N.: Management of exceptions on access control policies. In: *SEC*, pp. 97–108 (2007)
22. Srinivasan, V., Suri, S., Varghese, G.: Packet classification using tuple space search. In: *Proc. of SIGCOMM*, pp. 135–146 (1999)
23. Fitzgerald, W., Foley, S., Foghlú, M.Ó.: Network access control interoperation using semantic web techniques. In: *WOSIS*, pp. 26–37 (2008)

A Framework for Threat Assessment in Access Control Systems

Hemanth Khambhammettu, Sofiene Boulares, Kamel Adi,
and Luigi Logrippo

Laboratoire de Recherche en Sécurité Informatique
Université du Québec en Outaouais, Canada

{hemanth.khambhammettu,bous42,kamel.adi,luigi.logrippo}@uqo.ca

Abstract. We describe a framework for threat assessment specifically within the context of access control systems, where subjects request access to resources for which they may not be pre-authorized. The framework that we describe includes four different approaches for conducting threat assessment: an object sensitivity-based approach, a subject trustworthiness-based approach and two additional approaches which are based on the difference between object sensitivity and subject trustworthiness. We motivate each of the four approaches with a series of examples. We also identify and formally describe the properties that are to be satisfied within each approach. Each of these approaches results in different threat orderings, and can be chosen based on the context of applications or preference of organizations.

Keywords: Security, Access control, Threat assessment.

1 Introduction

The “need to share” information in dynamic environments has prompted the development of risk-based access control systems [2, 6, 9]. Essentially, in order to facilitate information sharing, risk-based access controls extend traditional access control paradigms to provide support for flexible decision-making by specifying acceptable security risk, operational needs and situational conditions [5]. Risk-based access control mechanisms make access decisions by determining the security risk associated with access requests and weighing such security risk against operational needs together with situational conditions. Specifically, an access request will be *permitted* if the operational benefits outweigh the security risk of granting access to information, and *denied* otherwise.

Clearly, computing the security risk of access requests is an important aspect of risk-based access control systems. However, determining security risk is a complex task, which requires the consideration of a variety of factors, such as the trustworthiness of subjects (or users), sensitivity of data, type of access being requested, access history of subjects and objects, physical or logical location or device from which access to data is being requested as well as protection capabilities and robustness of the system that maintains data [5]. Furthermore,

the interpretation and computation of security risk might differ based on the context of applications or culture of organizations.

The NIST Special Publication (SP) 800-30 [7] is a well-known risk management standard for enterprise systems. In this standard, risk is computed as a product of threat likelihood and impact values.

We adapt the risk assessment function of NIST SP 800-30 to develop a risk assessment function for access control systems. Specifically, the risk of permitting a subject $s \in S$ to perform an action $a \in A$ on an object $o \in O$ is given by the following function:

$$\text{Risk}(s, o, a) = \text{Threat}(s, o) \times \text{Vulnerability}(o) \times \text{Impact}(o, a). \quad (1)$$

where $\text{Threat}(s, o)$ represents the threat that a subject (threat source) s may present towards an object (threat target) o , $\text{Vulnerability}(o)$ represents the weakness within the existing controls for protecting (threat target) o and $\text{Impact}(o, a)$ represents the adverse impact on the satisfaction of security objectives that results from successfully performing action a on o .

Note that the NIST SP 800-30 provides no concrete suggestions for estimating threats that subjects present towards data objects. Furthermore, none of the existing work on estimating risk of access requests [1, 3, 6, 9] has explicitly provided approaches to estimate the threats that subjects may present towards data objects, except [2] which will be discussed in Section 3.

In this paper, we focus on estimating the threats that subjects may present towards objects, which are needed to compute risk estimates of access requests. We offer different approaches to estimate such threats which could be applied based on the context of applications.

Consider, for example, a business-to-business scenario that enables organizations to successfully execute their missions, which require subjects (or users) from both intra-business and inter-business to access sensitive data.

- Assume that access requests for sensitive data objects have been initiated by subjects who are employees of the business that owns the requested data objects. In other words, access requests are initiated by subjects who are directly known to (and trusted to some degree by) the system. In such situations, data owners might be more concerned about the sensitivity of data being requested than the trustworthiness of subject. Hence, sensitivity of data objects may be more important than trustworthiness of subjects for estimating the threats posed by subjects towards objects.
- Alternatively, assume that access requests for sensitive objects have been initiated by subjects who are employed by business partners. In other words, subjects who initiate access requests may not be (directly) known to data owners. In such situations, data owners might be greatly concerned about the trustworthiness of subjects for granting access to the requested data objects. Consequently, trustworthiness of subjects may be more important than sensitivity of data objects while estimating the threats posed by subjects towards objects.

Towards this end, we have focused our efforts on developing a suite of threat assessment techniques by considering the sensitivity of objects and trustworthiness of subjects. Of course, as mentioned earlier, we may have to consider a variety of additional factors to compute threat metrics in a comprehensive manner. Nevertheless, even by only considering trustworthiness of subjects and sensitivity of objects, our framework provides significant insights into various ways for assessing the threats posed by subjects towards objects.

We believe that the threat assessment techniques presented in this paper can be used to compute risk metrics by using Formula 1 that is given above. Due to page limitations, we restrict ourselves in this paper to focus on describing approaches which assess threats posed by subjects towards objects and defer the risk assessment of access requests as a subject for our future work.

The following are the main contributions of this paper.

- We present a family of approaches for assessing the threats posed by subjects towards objects: an object sensitivity-based approach, a subject trustworthiness-based approach and two additional approaches which are based on the difference between object sensitivity and subject trustworthiness. We use a series of examples as a basis for developing and identifying the properties of our threat assessment approaches, which provide support for *qualitative* threat assessment of subject-object accesses. Each of these approaches results in different threat orderings, and can be chosen based on the context of applications or preference of organizations.
- We demonstrate that the order in which “general threat principles” (described in Section 2.2) are applied makes a difference in the resulting threat vectors.

The rest of the paper is organized as follows. Section 2 describes our threat assessment framework. In Section 3, we compare our work with notable works of the literature. We draw conclusions for this paper and outline opportunities for future work in Section 4.

2 Threat Assessment Approaches

In this section, we develop our framework for estimating the threats posed by subjects towards objects within the context of access control systems by considering object sensitivity and subject trustworthiness. Throughout this section, we present a series of examples for developing the conceptual underpinnings of our threat assessment approaches.

2.1 Assumptions

We assume the existence of the following entities within an access control system: a set of subjects S and a set of objects O . Furthermore, we assume that every object is associated with a sensitivity score that reflects the protection needs of the data it holds. Typically, sensitivity scores are assigned to data objects by

data owners. A function $ol : O \rightarrow [0, 100]$ formally represents the assignment of sensitivity scores to objects. We also assume that every subject is associated with a trustworthiness score that reflects the trust bestowed upon the subject by the organization that owns the data. Such trustworthiness scores of subjects may be computed either statically by referring to attributes of subjects or dynamically by referring to access histories of subjects or system. A function $sl : S \rightarrow [0, 100]$ represents the assignment of trustworthiness scores to subjects.

We assume a “risk-based” system where a subject labeled to a certain trustworthiness score is *always* permitted to access objects whose sensitivity score is up to the subject’s trustworthiness score. In other words, all accesses initiated by a subject $s \in S$ to an object $o \in O$, such that $sl(s) \geq ol(o)$, will be *permitted*. However, should $sl(s) < ol(o)$ then access decisions are made by the system by computing the risk of granting access to o for s and referring to the risk acceptance level specified within the access control policy. In particular, an access request initiated by a subject $s' \in S$ to object o , such that $sl(s') < ol(o)$, will be *permitted* if the risk of granting access to o for s' is lower than the specified risk acceptance level, and *denied* otherwise.

2.2 Defining “Threat”

As discussed earlier in Section [1](#) and also shown above in Formula [1](#), threat metrics are a pre-requisite to compute risk metrics. We take the point of view that permitting a subject s to access an object o , such that $sl(s) < ol(o)$, presents by itself a “measurable threat”, independently of what might happen to the information that is accessed. In this section, we define the notion of “threat” in the context of the right to access objects by subjects. In particular, threat of subject-object accesses is defined as follows.

Definition 1. *We say that there exists a threat if a subject $s \in S$ is able to access an object $o \in O$, such that $sl(s) < ol(o)$.*

In other words, any attempt by a subject s to access an object o , such that $sl(s) \geq ol(o)$ does not present a threat.

Intuitively, any measure of threat is affected by one or more of the following three general principles:

- **Principle 1:** Threat increases as object sensitivity score increases.
- **Principle 2:** Threat increases as subject trustworthiness score decreases.
- **Principle 3:** Threat increases as the difference between the object sensitivity score and subject trustworthiness score increases.

We define a function **Threat** : $S \times O \rightarrow [0, 1]$ that represents the threat value of a subject $s \in S$ accessing an object $o \in O$. We use relation \preceq^T to denote an ordering that represents threat on a set of subject-object accesses. In particular, \preceq^T can be defined in terms of subject-object accesses in the following way: $(s, o) \preceq^T (s', o')$ iff **Threat**(s, o) \leq **Threat**(s', o').

The relation \preceq^T allows threats to be compared, and “greater” and “lesser” threats assessed. We define $(s, o) \simeq^T (s', o')$ iff $(s, o) \preceq^T (s', o')$ and $(s, o) \succeq^T (s', o')$.

Subject	Trustworthiness Score
Alice	90
Bob	80
Carol	70
Dave	80

(a) Subjects and trustworthiness scores

Object	Sensitivity Score	Object Description
o	90	location of weapons
o'	100	launch codes of weapons

(b) Objects and sensitivity scores

Fig. 1. Configuration of the running example

(s', o') , and say (s', o') is a greater threat than (s, o) and write $(s, o) \prec^T (s', o')$ if $(s, o) \preceq^T (s', o')$ and $(s, o) \not\approx^T (s', o')$. We may write $(s', o') \succ^T (s, o)$ whenever $(s, o) \prec^T (s', o')$.

2.3 Running Example

In this section, we describe the setting of a scenario that is used in the rest of the paper for motivating our threat assessment approaches.

We assume the existence of the following four subjects: **Alice**, **Bob**, **Carol** and **Dave**. Figure 1(a) illustrates the trustworthiness scores of these four subjects.

Let us consider two objects o and o' within a military context, such that o maintains information regarding the **location** for nuclear weapons and o' maintains **launch codes** for nuclear weapons. It is reasonable to assume here that information regarding the **launch codes** for nuclear weapons is *more sensitive* than the **location** of nuclear weapons. Hence, we assume that object o is assigned a sensitivity score 90 and object o' is assigned a sensitivity score 100. Figure 1(b) shows the sensitivity scores of objects o and o' .

Recall that the objective of our work is to assess the threat of access requests, where a subject $s \in S$ who initiated the request may not be pre-authorized for the requested data object $o \in O$. Hence, throughout this paper, we only cite examples where $sl(s) < ol(o)$.

2.4 Object-Based Threat Assessment

It is possible that certain applications which maintain “highly” sensitive data, such as government or military systems, may understand or interpret threat in terms of access to such data. We now give examples that motivate our technique for threat assessment that primarily is based on the sensitivity score of objects.

Example 2. Suppose that **Alice** requests access to object o' , and **Bob** requests access to object o . We have the following from Figure 1: $sl(\text{Alice}) = 90$, $sl(\text{Bob}) = 80$, $ol(o) = 90$ and $ol(o') = 100$.

If we were to consider object sensitivity score to be the basic criteria for determining threat measures, then according to Principle 1 stated in Section 2.2 allowing **Alice** to access object o' is a greater threat than allowing **Bob** to access

object o . This is simply because the sensitivity score of object o' is higher than the sensitivity score of object o .

In the above example, we were able to understand which access poses a greater threat by simply comparing the sensitivity scores of those two objects. However, as we show below, such a technique is no longer sufficient when object sensitivity scores are the same.

Example 3. Let us extend Example 2 by considering an additional subject **Carol** whose trustworthiness score is given in Figure 1 as follows: $sl(\text{Carol}) = 70$. Suppose that **Carol** requests access to object o' , where $ol(o') = 100$. In other words, both **Alice** and **Carol** request access to object o' .

Now, if we were to determine which of these two accesses is a greater threat, then according to Principle 2 (see Section 2.2) one is likely to conclude that allowing **Carol** to access object o' is a greater threat than allowing **Alice** to access o' . This is because **Carol** who has a trustworthiness score of 70 is less trusted than **Alice** who has a trustworthiness score of 90.

Remark 4 (from Examples 2 and 3). A threat assessment technique that primarily is based on object-sensitivity scores should support the following:

1. always apply Principle 1 (that is, threat always increases as object sensitivity score increases),
2. whenever object sensitivity scores are the same, apply Principle 2 (that is, threat increases as subject trustworthiness score decreases).

Based on Remark 4, we obtain the following ordering of threat for the accesses which were considered in Examples 2 and 3: $(\text{Bob}, o) \prec^T (\text{Alice}, o') \prec^T (\text{Carol}, o')$.

It can easily be seen that we can construct a “priority order” of excessive accesses by subjects for objects, when sensitivity scores are higher than trustworthiness scores, in terms of their threat by adhering to the properties of Remark 4. Essentially, the properties of Remark 4 can be generalized as follows: $(s, o) \prec^T (s', o')$ if either

1. $ol(o) < ol(o')$ or
2. $ol(o) = ol(o')$ and $sl(s') < sl(s)$.

Within an object-based threat assessment approach, whenever object sensitivity scores are the same, unlike Remark 4 we may wish to apply Principle 3 as a secondary criterion. In other words, we may use “the difference of object sensitivity and subject trustworthiness scores” as a secondary parameter, rather than subject trustworthiness scores. Note however that whenever the object sensitivity score is fixed, the difference between object sensitivity and subject trustworthiness scores increases only if subject trustworthiness scores decrease. This means that the threat priority order remains the same irrespective of whether we apply Principle 2 or Principle 3 as a secondary criterion. Hence, we do not describe the subcase that applies Principle 3 as a secondary criterion.

2.5 Subject-Based Threat Assessment

As discussed earlier in Section 1, in certain scenarios (such as business-to-business environments), access requests could be initiated by subjects who may not be (directly) known to data owners. In such situations, trustworthiness of subjects may take higher preference than sensitivity of data objects while estimating access threats.

We now give examples that motivate our technique for threat assessment that, primarily, is based on trustworthiness scores of subjects.

Example 5. *Let us reuse the setting of Example 2 here. That is, we consider subjects Alice and Bob, and suppose that Alice requests access to object o' , and Bob requests access to object o .*

Now, should subject trustworthiness score be the basic criteria for conducting threat assessment, then according to Principle 2 (see Section 2.2) one is likely to conclude that granting access to Bob for object o is a greater threat than granting access to Alice for o' . This is because Bob, who has a subject trustworthiness score of 80, is less trusted than Alice, who has a subject trustworthiness score of 90.

Example 6. *Let us extend Example 5 by considering an additional user Dave where $sl(\text{Dave}) = 80$ (see Figure 7(a)). Now both Bob and Dave have the same trustworthiness score. Suppose that Dave is requesting access to object o' .*

Now, if we were to determine which one of the above two accesses of Bob and Dave poses a greater threat, then according to Principle 1 (see Section 2.2) we may reasonably say that granting access to Dave for o' is a greater threat than granting access to Bob for o . This is because—although both Bob and Dave have the same trustworthiness scores—Dave is requesting access to object o' which has a higher sensitivity score than object o that Bob is requesting access to.

Remark 7 (from Examples 5 and 6). *A threat assessment technique that primarily is based on subject-trustworthiness scores should support the following properties:*

1. *always apply Principle 2 (that is, threat increases as subject trustworthiness score decreases),*
2. *whenever subject trustworthiness scores are the same, apply Principle 1 (that is, threat increases as object sensitivity score increases).*

Based on Remark 7 we obtain the following ordering of threat for the subject-object accesses which were considered in Examples 5 and 6: $(\text{Alice}, o') \prec^T (\text{Bob}, o) \prec^T (\text{Dave}, o')$.

Essentially, the properties of Remark 7 can be generalized as follows: $(s, o) \prec^T (s', o')$ if either

1. $sl(s') < sl(s)$ or
2. $sl(s') = sl(s)$ and $ol(o') > ol(o)$.

It is important to note the effect of the basic criterion on threat orderings or metrics when subjects request access to objects, where sensitivity scores are higher than trustworthiness scores. In particular, should the sensitivity score of objects be the basic criterion for assessing threat, then $(\text{Bob}, o) \prec^T (\text{Alice}, o')$ (see Example 2). Whereas, if the trustworthiness score of subjects is considered as the basic criterion for assessing threat, then $(\text{Alice}, o') \prec^T (\text{Bob}, o)$ (see Example 5).

Note that, whenever subject trustworthiness scores are the same, unlike Remark 7 we may wish to apply Principle 3 as a secondary criterion. That is, we may wish to use “the difference of object sensitivity and subject trustworthiness scores” as a secondary parameter, rather than object sensitivity scores. However, note that whenever the subject trustworthiness score is fixed, the difference between object sensitivity and subject trustworthiness scores increases only if object sensitivity scores increase. This means that, within a subject-based threat assessment approach, the threat priority order remains the same irrespective of whether we apply Principle 1 or Principle 3 as a secondary criterion. Hence, we do not describe the subcase that applies Principle 3 as a secondary criterion.

2.6 Difference of Scores-Based Threat Assessment

In certain scenarios, we may not be directly concerned with either the object sensitivity scores or subject trustworthiness scores; however, our objective could be to understand threat simply in terms of the difference between object sensitivity and subject trustworthiness scores. Essentially, in such an approach, the degree of threat proportionally increases with the difference between object sensitivity and subject trustworthiness scores.

In this section, we adopt such a notion of threat (as described above) and develop two different techniques for threat assessment which, primarily, are based on the difference between the sensitivity scores of objects and subjects. We first give examples below for motivating our threat assessment techniques and then formalize their properties.

Example 8. *Let us reuse the setting from Examples 2 and 3 here. In particular, we consider user Bob from Example 2 and Carol from Example 3. As before, we suppose that Bob requests access for object o and Carol requests access for object o' .*

Now, should the basic criteria for determining threat measures be the difference between object sensitivity and subject trustworthiness scores, then according to Principle 3 (see Section 2.2) granting access to Carol for object o' is a greater threat than granting access to Bob for object o . This is because the difference between the sensitivity score of object o and trustworthiness score of Carol (which is $100 - 70 = 30$) is greater than the difference between the sensitivity score of object o' and trustworthiness score of Bob (which is $90 - 80 = 10$).

Note, in the above example, that the differences between object sensitivity and subject trustworthiness scores for the two accesses under consideration are not the same. Hence, we were able to compare the threat of granting accesses by

simply computing and comparing the difference between object sensitivity and subject trustworthiness scores of those two subject-object accesses.

We show in the following example that such a technique is no longer sufficient when the difference between object sensitivity and subject trustworthiness scores of subject-object accesses is the same.

Example 9. *Let us extend Example 8 by also considering subject Alice. As before, we suppose that Alice requests access for object o' , where $sl(\text{Alice}) = 90$ and $ol(o') = 100$ (see Figure 7).*

Note that the difference between the sensitivity score of object o' and trustworthiness score of Alice (which is $100 - 90 = 10$) is the same as the difference between the sensitivity score of object o and trustworthiness score of Bob (which is $90 - 80 = 10$). Hence, it is not immediately obvious which of the above two subject-object accesses poses a greater threat.

If we were to determine which of the two subject-object accesses considered in Example 9 is a greater threat, then we may choose between applying either Principle 1 or Principle 2 (see Section 2.2) yielding two different approaches, which consider different secondary parameters, for resolving the parity observed in Example 9. These two approaches are described below.

Difference Weighted by Object Sensitivity Score. In this approach, we consider object sensitivity scores as a secondary criterion and apply Principle 1 which says that threat increases with an increase in object sensitivity scores (see Section 2.2) for resolving the parity observed in Example 9.

This means that, in Example 9, granting access to object o' for Alice is a *greater threat* than granting access to object o for Bob, because $ol(o') > ol(o)$.

Remark 10. *A threat assessment technique that primarily is based on the difference between object sensitivity and subject trustworthiness scores, and that uses object sensitivity scores as a secondary criterion should support the following properties:*

1. *always apply Principle 3 (that is, threat increases as the difference between object sensitivity and subject trustworthiness scores increases),*
2. *whenever parity is observed on the difference between object sensitivity and subject trustworthiness scores, apply Principle 1 (that is, threat increases as object sensitivity score increases).*

Based on Remark 10, we obtain the following ordering of threat for the subject-object accesses which were considered in Examples 8 and 9 ($(\text{Bob}, o) \prec^T (\text{Alice}, o') \prec^T (\text{Carol}, o')$).

The properties of Remark 10 can be generalized as follows: $(s, o) \prec^T (s', o')$ if either

1. $(ol(o) - sl(s)) < (ol(o') - sl(s'))$ or
2. $(ol(o) - sl(s)) = (ol(o') - sl(s'))$ and $ol(o') > ol(o)$.

Difference Weighted by Subject Trustworthiness Score. In this approach, we consider subject trustworthiness scores as a secondary criterion and apply Principle 2 which says that threat increases with a decrease in subject trustworthiness scores (see Section 2.2) for resolving the parity observed in Example 9.

Recall from Example 9 that the difference between the sensitivity score of object o' and trustworthiness score of Alice (which is $100 - 90 = 10$)—*is the same as*—the difference between the sensitivity score of object o and trustworthiness score of Bob (which is $90 - 80 = 10$). In this approach, granting access to object o Bob poses a *greater threat* than granting access to object o' Alice, because $sl(\text{Bob}) < sl(\text{Alice})$.

Remark 11. *A threat assessment technique that primarily is based on the difference between object sensitivity and subject trustworthiness scores, and that uses the subject trustworthiness scores as a secondary criterion should support the following properties:*

1. *always apply Principle 3 (that is, threat increases as the difference between object sensitivity and subject trustworthiness scores increases),*
2. *whenever parity is observed on the difference between object sensitivity and subject trustworthiness scores, apply Principle 2 (that is, threat increases as subject trustworthiness score decreases).*

Based on Remark 11, we obtain the following ordering of threat for the subject-object accesses which were considered in Examples 8 and 9: $(\text{Alice}, o') \prec^T (\text{Bob}, o) \prec^T (\text{Carol}, o')$.

The properties of Remark 11 can be generalized as follows: $(s, o) \prec (s', o')$ if either

1. $(ol(o) - sl(s) < ol(o') - sl(s'))$ or
2. $(ol(o) - sl(s) = ol(o') - sl(s'))$ and $sl(s') < sl(s)$.

3 Discussion and Related Work

Cheng *et al* have proposed Fuzzy Multi-Level Security (Fuzzy MLS), which quantifies the risk of an access request in multi-level security systems as a product of the value of information and probability of unauthorized disclosure [2]. Fuzzy MLS considers that all subject-object accesses include a temptation to leak information and aims to quantify the risk of “unauthorized disclosure” of information by subjects.

In comparison with Fuzzy MLS, the aim of our framework is to assess the threat posed by subjects towards objects by referring to object sensitivity and subject trustworthiness scores. Although our framework considers simpler requirements than Fuzzy MLS, we have described four different approaches for assessing threat where each approach is biased towards a different set of criteria (unlike Fuzzy MLS whose temptation index is biased only towards object sensitivity scores).

Ni *et al* used fuzzy inference techniques as an approach for estimating access risks and developed an enforcement mechanism for risk based access control [6]. In comparison, we exclusively focus on threat assessment which is a pre-requisite for estimating access risks. Bartsch proposed a policy override calculus for qualitative risk assessment in the context of role-based access control systems [1]. In comparison with the work of Bartsch, our framework is developed in the context of generic access control systems by referring to the sensitivity of objects and trustworthiness of subjects. Diep *et al* described an access control model with context-based decisions that includes quantitative risk assessment [3]. However, there is no description for computing threat measures in [3].

Wang and Jin proposed a method to quantify access risk by considering need-to-know requirements for privacy protection within the context of health information systems [9]. This work exploited the concept of entropy from information theory to compute risk scores of access requests. We believe that our framework could be extended to also consider need-to-know requirements while assessing threats of subject-object accesses. Kandala *et al* developed a framework that captures various components and their interactions in order to develop “abstract models” for RADAC [4]. However, this work does not consider concrete details of assessing threat or risk.

4 Conclusions

The main contribution of this paper is a framework that includes a family of threat assessment approaches for subject-object accesses, which can be selected based on the context of applications or on the preference of organizations. Specifically, our framework includes four different ways of assessing the threat of subject-object accesses. Our first threat assessment approach, described in Section 2.4, primarily considers the sensitivity scores of objects, and thus gives more priority to the sensitivity of data. We have described another threat assessment approach in Section 2.5 that mainly considers trustworthiness scores of subjects, and thus gives more priority to subject trustworthiness than object sensitivity. A third approach that is based on the idea that threat can be calculated as the difference between object sensitivity and subject trustworthiness scores has been described in Section 2.6, and for this approach we have identified two different subcases by considering the object sensitivity scores and subject trustworthiness scores as secondary parameters.

We have demonstrated that the order in which the “three general threat principles” are applied makes a difference in the resulting threat vectors. This result is important because the approach adopted to assess the threat posed by subjects towards objects will subsequently affect the computation of risk metrics.

To the best of our knowledge, our work represents the first attempt in the literature to conduct a comprehensive study of several alternative approaches for threat assessment by considering object sensitivity and subject trustworthiness scores. We have presented several examples which justify our approaches in intuitive terms.

As mentioned in Section 1, our ultimate goal is to develop a framework for estimating risk of access requests by adopting the well accepted risk assessment function of the NIST SP 800-30. This requires us to extend the work reported in this paper in the following two directions. Firstly, in order to compute impact values of object-action pairs, we intend to exploit “data classification policies” [8]. Subsequently, we will use such impact values together with threat values for quantifying risk of (*subject, object, action*) triples based on Formula 1. We also aim to use real-world data sets in order to evaluate the efficacy of the threat assessment approaches described in this paper.

Acknowledgements. This research was funded in part by grants of the Natural Sciences and Engineering Research Council of Canada and CA Technologies. We thank Serge Mankovski of CA Technologies for having motivated our work.

References

1. Bartsch, S.: A calculus for the qualitative risk assessment of policy override authorization. In: Proceedings of the 3rd International Conference on Security of Information and Networks (SIN 2010), pp. 62–70 (2010)
2. Cheng, P.C., Rohatgi, P., Keser, C., Karger, P.A., Wagner, G.M., Reninger, A.S.: Fuzzy multi-level security: An experiment on quantified risk-adaptive access control. In: Proceedings of IEEE Symposium on Security and Privacy (SP 2007), pp. 222–230 (2007)
3. Diep, N.N., Hung, L.X., Zhung, Y., Lee, S., Lee, Y.-K., Lee, H.: Enforcing access control using risk assessment. In: Proceedings of the 4th European Conference on Universal Multiservice Networks (ECUMN 2007), pp. 419–424 (2007)
4. Kandala, S., Sandhu, R., Bhamidipati, V.: An attribute based framework for risk-adaptive access control models. In: Proceedings the 6th International Conference on Availability, Reliability and Security (ARES 2011) (2011)
5. McGraw, R.: Risk adaptive access control (RADAC). In: Proceedings of NIST & NSA Privilege Management Workshop (2009)
6. Ni, Q., Bertino, E., Lobo, J.: Risk-based access control systems built on fuzzy inferences. In: Proceedings of 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2010), pp. 250–260 (2010)
7. NIST. Risk management guide for information technology systems. National Institute of Standards and Technology, Special Publication (SP) 800-30 (2002)
8. NIST. Guide for mapping types of information and information systems to security categories. National Institute of Standards and Technology, Special Publication (SP) 800-60, volumes I & II (2008)
9. Wang, Q., Jin, H.: Quantified risk-adaptive access control for patient privacy protection in health information systems. In: Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2011), pp. 406–410 (2011)

Support for Write Privileges on Outsourced Data

Sabrina De Capitani di Vimercati¹, Sara Foresti¹, Sushil Jajodia²,
Stefano Paraboschi³, and Pierangela Samarati¹

¹ DTI - Università degli Studi di Milano, 26013 Crema, Italia

`firstname.lastname@unimi.it`

² CSIS - George Mason University, Fairfax, VA 22030-4444, USA

`jajodia@gmu.edu`

³ DIIMM - Università degli Studi di Bergamo, 24044 Dalmine, Italia

`parabosc@unibg.it`

Abstract. In the last years, data outsourcing has received an increasing attention by the research community thanks to the benefits that it brings in terms of data management. A basic requirement in such a scenario is that outsourced data be made accessible only to authorized users, that is, no unauthorized party (including the storing server) should have access to the data. While existing proposals provide a sound basis for addressing such a need with respect to data dissemination (i.e., enforcement of read authorizations), they fall short on the support of write authorizations.

In this paper we address such an open problem and present an approach to enforce write privileges over outsourced data. Our work nicely extends and complements existing solutions, and exploiting key derivation tokens, hashing, and HMAC functions provides efficient and effective controls.

Keywords: Data outsourcing, data protection, authorization management.

1 Introduction

Data outsourcing offers to end users and companies the opportunity to benefit from the lower costs, higher availability and larger elasticity that are offered by the rapidly growing market of cloud providers. The major obstacle to the adoption of cloud storage services is commonly recognized to be the uncertainty about a correct management of security requirements, with the user interested in robust guarantees about the confidentiality and integrity of the outsourced data. Several techniques have been recently proposed by the research community [5,10,16]. Most of these proposals have been developed with reference to scenarios where the data should remain confidential also to the external server storing them, which is considered *honest-but-curious*, i.e., trustworthy for managing resources but should not see or learn the resource content. To provide such confidentiality guarantee, these proposals (e.g., [5,10,16]) assume data to

be encrypted before being outsourced to the external server, and they associate with the encrypted data additional indexing information that can be used by the server to perform queries on the encrypted data. For efficiency reasons, encryption is based on symmetric keys. Earlier proposals typically consider data to be encrypted with a single key, assuming then all users to have complete visibility of the resources in the data collection or the data owner to mediate access requests to the data to enforce write authorizations. More recent proposals, addressing the problem of providing selective visibility over the data to the users (different sets of users can enjoy visibility over different resources), have proposed the application of a ‘selective encryption’ approach. Intuitively, different user views can be enforced by using different encryption keys for the resources and users will be able to have visibility on the resources depending on the keys they know. Proper modeling and derivation techniques have been devised to ensure limited key management overhead in such selective encryption.

While interesting and promising, however, all the solutions above assumed outsourced data to be read only. In other words, the owner can modify resources while all other users can only read them. Such an assumption can result restrictive in several scenarios where a data owner outsourcing the data to the external server may also want to authorize other users (again selectively) to write the resources it stores. We then build upon and complement previous proposals providing a solution for enforcing write authorizations on the encrypted outsourced data. Our solution is based on the same principles as previous proposals exploiting encryption itself for enforcing access control and having efficiency and manageability in mind. Our proposal enjoys compatibility and easy deployment with previous proposals, thus providing a natural and efficient extension to them for enforcing write authorizations.

The remainder of the paper is organized as follows. Section 2 introduces the basic concept on previous work on which our proposal is based. Section 3 illustrates our proposal for enforcing write authorizations. Section 4 discusses the control features of our proposal that allow a data owner to check the write operations executed and detect possible misbehaviors by the server or by the users. Section 5 discusses related work. Finally, Section 6 concludes the paper.

2 Basic Concepts

Our work builds upon and extends a previous proposal [8] for confidential data outsourcing. In this section, we briefly introduce the basic concepts of the original proposal that are useful to our treatment.

According to the proposal in [8], a data owner outsourcing data to a honest-but-curious server and wishing to provide selective visibility over them to other users encrypts resources before outsourcing them to the server and reflects the authorization policy in the encryption itself. Therefore, each resource o is encrypted with a key to be made known only to the users authorized to read o , that is, to users who belong to the access control list of o . Symmetric encryption is used and different keys are assumed: one for each user and one for each group

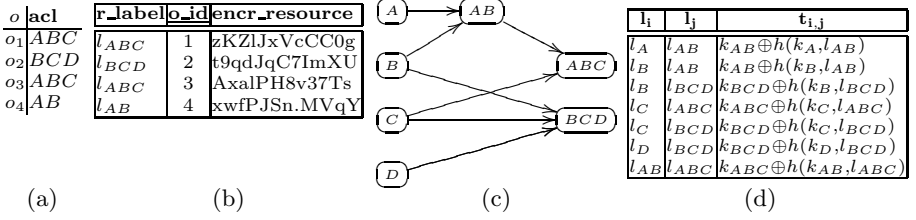


Fig. 1. An example of four resources with their acls (a), encrypted resources (b), key derivation graph (c), and tokens (d)

of users that corresponds to an access control list. The adoption of a *key derivation technique* based on public tokens permits users to access the system while having to manage only one key. Each key k_i is identified by a public label l_i . Given keys k_i and k_j , token $t_{i,j}$ is computed as $k_j \oplus h(k_i, l_j)$, with \oplus the bitwise xor operator, and h a deterministic cryptographic function. Token $t_{i,j}$ permits to derive key k_j from the knowledge of key k_i and public label l_j [2]. All keys with which resources are encrypted are then connected in a *key derivation graph*. A key derivation graph is a DAG whose nodes correspond to keys (of users and acls) and whose edges correspond to tokens that ensure that each user can - via a sequence of public tokens - derive the keys corresponding to the sets to which she belongs. Each users is then communicated the key of the node representing herself in the graph. Each resource is encrypted with the key corresponding to its acl. Encrypted resources as well as the tokens are outsourced to the server. In particular, for each resource o , the external server stores the encrypted version of the resource together with the resource identifier and the label of the key with which the resource is encrypted. A user authorized to read a resource (i.e., belonging to its acl) can, via the tokens available on the server, derive the key corresponding to the acl of the resource and decrypt it.

Example 1. Consider a system with four users $\mathcal{U}=\{A,B,C,D\}$ and four resources $\mathcal{O}=\{o_1,o_2,o_3,o_4\}$, whose acls are reported in Figure 1(a). Figure 1(b) illustrates the encrypted resources stored at the server, where: r_label is the label of the key used to encrypt the resource (i.e., the key associated with its acl); o_id is the resource identifier; and $encr_resource$ is the encrypted resource. Figure 1(c) illustrates the key derivation graph enforcing the authorizations. For simplicity and readability, in the key derivation graph we denote a key corresponding to a given acl U (i.e., a key with label l_U and value k_U) simply with U . Figure 1(d) illustrates the tokens corresponding to the key derivation graph in Figure 1(c).

3 Enforcement of Write Authorizations

The support of only read accesses may result limiting when considering emerging data sharing scenarios (e.g., document sharing), where users different from the data owner can be authorized to modify resources. Unfortunately, the keys associated with resources for regulating the read accesses to them cannot be used

for regulating write accesses as well. As a matter of fact, we can imagine that in many situations the set of users authorized to write a resource is different from (typically being a subset of) the set of users authorized to read the resource. A straightforward solution for enforcing write authorizations can be to simply outsource to the external server the authorization policy (for write privileges) as is and have the server to perform traditional (authorization-based) access control. This would involve user authentication and policy enforcement, with the drawback of requesting a considerable management overhead. Following the same spirit of the proposal in [8], and to the aim of providing an extension easily integrable with it, we aim at basing enforcement of write authorizations also on encryption. While simply encrypting a resource with a key known to all and only the users authorized to read the resource automatically ensures enforcement of read authorizations, enforcement of write privileges requires cooperation from the external server. Having resources being tied to access restrictions to them by means of cryptographic solutions provides more robustness and flexibility of the control, whose enforcement is less exposed to server misbehaviors and not affected by possible server allocation strategies.

The basic idea of our approach consists in associating each resource with a *write tag*, which is defined by the data owner, and encrypting it with a key to be known only to the users authorized to write the resource and to the external server (i.e., only the external server and the users authorized to write a resource can have access to the corresponding write tag). The server will accept a write operation on a resource when the user requesting it shows knowledge of the write tag. The key used for encrypting the write tag has to be shared among the server and the writers, and we leverage on the underlying structure already in place for regulating read operations to achieve this.

3.1 Key Derivation Structure

Elaborating the approach in [8], and adapting it to our context, we introduce a *set-based key derivation graph* as follows.

Definition 1 (Set-based key derivation graph). *Let \mathcal{U} be a set of users and $\mathcal{U}' \subseteq 2^{\mathcal{U}}$ be a family of subsets of users in \mathcal{U} such that $\forall u \in \mathcal{U}, \{u\} \in \mathcal{U}'$. A set-based key derivation graph is a triple $(\mathcal{K}, \mathcal{L}, \mathcal{T})$, with \mathcal{K} a set of keys, \mathcal{L} the set of corresponding labels, and \mathcal{T} a set of tokens, such that:*

1. $\forall U \in \mathcal{U}'$, there exists a key $k_U \in \mathcal{K}$;
2. $\forall u \in \mathcal{U}, \forall U \in \mathcal{U}' - \{u\}$, there exists a token $t_{u,U}$ or a sequence $\langle t_{u,U_1}, \dots, t_{u,U_n} \rangle$ of tokens in \mathcal{T} , with $t_{c,d}$ following $t_{a,b}$ in the sequence if $b = c$, and $n \geq 1$, iff $u \in U$.

Intuitively, the approach in [8], while not explicitly reporting the definition above, basically translates to it if we assume \mathcal{U}' to include, besides singleton sets of users, the read acls of resources.

Since in the scenario we consider each resource is associated with a write tag that must be encrypted with a key shared among the server and the authorized

```

INPUT
 $\mathcal{U}$  : set of users
 $\mathcal{S}$  : external server
 $\mathcal{U}' \subseteq 2^{\mathcal{U}}$  : family of subsets of users in  $\mathcal{U}$ 
 $\mathcal{U}'' \subseteq \mathcal{U}'$  : subset of  $\mathcal{U}'$ 

OUTPUT
 $(\mathcal{K}, \mathcal{L}, \mathcal{T})$  : key derivation structure

DEFINE_KEY_DERIVATION_STRUCTURE
1: /* Step 1: define the set-based key derivation graph */
2:  $\mathcal{K}' := \emptyset, \mathcal{L}' := \emptyset; \mathcal{T}' := \emptyset$ 
3: for each  $U \in \mathcal{U}'$  do /* generate a key for all  $U \in \mathcal{U}'$  (C1 in Def. 1) */
4:   generate a key  $k_U$  and a label  $l_U$ 
5:    $\mathcal{K}' := \mathcal{K}' \cup \{k_U\}, \mathcal{L}' := \mathcal{L}' \cup \{l_U\}$ 
6: /* define a set of tokens s.t.  $\forall U \in \mathcal{U}', \forall u \in U, k_U$  is derivable from  $k_u$  iff  $u \in U$  (C2 in Def. 1) */
7: for each  $U_j \in \mathcal{U}', |U_j| > 1$  do
8:    $cover_j := \{U_1, \dots, U_n \subseteq \mathcal{U}' \mid \bigcup_{i=1}^n U_i = U_j\}$ 
9:    $\mathcal{T}' := \mathcal{T}' \cup \{t_{U_i, U_j} = k_{U_j} \oplus h(k_{U_i}, l_{U_j}) \mid U_i \in cover_j\}$ 
10: /* Step 2: define a key derivation structure */
11: generate a key  $k_{\mathcal{S}}$  and a label  $l_{\mathcal{S}}$  /* generate a key for the external server (C1 in Def. 2) */
12:  $\mathcal{K} := \mathcal{K}' \cup \{k_{\mathcal{S}}\}, \mathcal{L} := \mathcal{L}' \cup \{l_{\mathcal{S}}\}$ 
13:  $\mathcal{T} := \mathcal{T}'$ 
14: for each  $U \in \mathcal{U}''$  do /* for each  $U \in \mathcal{U}''$ , compute  $k_{U \cup \{\mathcal{S}\}}$  as the hash of  $k_U$  (C1 in Def. 2) */
15:    $k_{U \cup \{\mathcal{S}\}} := h(k_U)$ 
16:   generate a label  $l_{U \cup \{\mathcal{S}\}}$ 
17:    $\mathcal{K} := \mathcal{K} \cup \{k_{U \cup \{\mathcal{S}\}}\}, \mathcal{L} := \mathcal{L} \cup \{l_{U \cup \{\mathcal{S}\}}\}$ 
18:    $\mathcal{T} := \mathcal{T} \cup \{t_{\mathcal{S}, U \cup \{\mathcal{S}\}} = k_{U \cup \{\mathcal{S}\}} \oplus h(k_{\mathcal{S}}, l_{U \cup \{\mathcal{S}\}})\}$  /* token from  $k_{\mathcal{S}}$  to  $k_{U \cup \{\mathcal{S}\}}$  (C2 in Def. 2) */
19: return $((\mathcal{K}, \mathcal{L}, \mathcal{T}))$ 

```

Fig. 2. Function that defines a key derivation structure

writers of the resource, we need to extend the set-based key derivation graph with the external server. The external server cannot however be treated as an authorized user of the system since it cannot access the plaintext of the outsourced resources. We then define a *key derivation structure* by extending the set-based key derivation graph to include also keys shared between authorized users and the server, which will be used to encrypt the write tags for enforcing write privileges (see Section 3.2). These additional keys can be derived both by the authorized users, applying a secure hash function to a key they already know (or can derive via a sequence of tokens), and by the external server, exploiting a token specifically added to the key derivation structure. Formally, a key derivation structure is defined as follows.

Definition 2 (Key derivation structure). *Let \mathcal{U} be a set of users, \mathcal{S} be an external server, $\mathcal{U}' \subseteq 2^{\mathcal{U}}$ be a family of subsets of users in \mathcal{U} such that $\forall u \in \mathcal{U}, \{u\} \in \mathcal{U}'$, \mathcal{U}'' be a subset of \mathcal{U}' , and $(\mathcal{K}', \mathcal{L}', \mathcal{T}')$ be a set-based key derivation graph. A key derivation structure is a triple $(\mathcal{K}, \mathcal{L}, \mathcal{T})$, with \mathcal{K} a set of keys, \mathcal{L} the set of corresponding labels, and \mathcal{T} a set of tokens, such that:*

1. $\mathcal{K} = \mathcal{K}' \cup \{k_{\mathcal{S}}\} \cup \{k_{U \cup \{\mathcal{S}\}} = h(k_U) \mid U \in \mathcal{U}'' \text{ and } h \text{ is a secure hash function}\}$;
2. $\mathcal{T} = \mathcal{T}' \cup \{t_{\mathcal{S}, U \cup \{\mathcal{S}\}} \mid U \in \mathcal{U}''\}$.

Figure 2 illustrates function **Define_Key_Derivation_Structure** that builds a key derivation structure. The function receives as input a set \mathcal{U} of users, an external server \mathcal{S} , and two families \mathcal{U}' and \mathcal{U}'' of subsets of users in \mathcal{U} , with $\mathcal{U}'' \subseteq \mathcal{U}'$. The function first defines a set-based key derivation graph $(\mathcal{K}', \mathcal{L}', \mathcal{T}')$

by leveraging on the algorithms in [8]. The function then extends the set-based key derivation graph by generating a key for the external server and for each set $U \cup \{\mathcal{S}\}$, with $U \in \mathcal{U}''$, and by inserting into \mathcal{T} a token that allows the derivation of $k_{U \cup \{\mathcal{S}\}}$ from $k_{\mathcal{S}}$. The following theorem, whose proof is omitted for space constraints, formally shows that function **Define_Key_Derivation_Structure** correctly computes a key derivation structure.

Theorem 1 (Correctness). *Let \mathcal{U} be a set of users, \mathcal{S} be an external server, $\mathcal{U}' \subseteq 2^{\mathcal{U}}$ be a family of subsets of users in \mathcal{U} such that $\forall u \in \mathcal{U}$, $\{u\} \in \mathcal{U}'$, and \mathcal{U}'' be subset of \mathcal{U}' . Triple $\langle \mathcal{K}, \mathcal{L}, \mathcal{T} \rangle$ computed by function **Define_Key_Derivation_Structure** in Figure 2 is a key derivation structure.*

Example 2. Consider a system with four users $\mathcal{U} = \{A, B, C, D\}$, a family $\mathcal{U}' = \{A, B, C, D, AB, CD, ABC, BCD\}$ of subsets of users, and a subset $\mathcal{U}'' = \{C, AB, CD\}$ of \mathcal{U}' . Figure 3(c) illustrates the key derivation structure computed by function **Define_Key_Derivation_Structure** in Figure 2. In the figure, continuous lines correspond to tokens forming the set-based key derivation graph, dotted lines correspond to tokens added to define the key derivation structure, and double lines correspond to hash-based derivation.

3.2 Resource Encryption, Write Tags, and Access Control Enforcement

We consider the case of a data owner outsourcing her resources, which can be read or modified by other users. Each resource o is then associated with two access control lists: *i*) a read access list $r[o]$ reporting the set of users authorized to read o , and *ii*) a write access list $w[o]$ reporting the set of users authorized to write o . We assume the users authorized to write a resource also to be able to read it, that is, $\forall o \in \mathcal{O}: w[o] \subseteq r[o]$. As mentioned at the beginning of this section, we use write tags for enforcing write authorizations. For each resource $o \in \mathcal{O}$, the data owner defines a write tag, denoted $tag[o]$, by using a secure random function. The consideration of a secure random function ensures the independence of the write tag from the resource identifier (since otherwise readers not authorized to write could infer it) as well as from its content (since again readers could infer it or the server could infer information on the resource content). The write tag associated with a resource o is then encrypted with the key corresponding to the set U of users authorized to write it (i.e., $U = w[o]$) plus the server \mathcal{S} , that is, with key $k_{U \cup \{\mathcal{S}\}}$. Each resource $o \in \mathcal{O}$ is then stored at the external server in encrypted form together with the following metadata.

- r_label:** it is the label of the key with which the resource is encrypted, which is the key associated with the set of users authorized to read o (i.e., $l_{r[o]}$).
- w_label:** it is the label of the key shared by the users authorized to write o and the server \mathcal{S} (i.e., $l_{w[o] \cup \{\mathcal{S}\}}$).

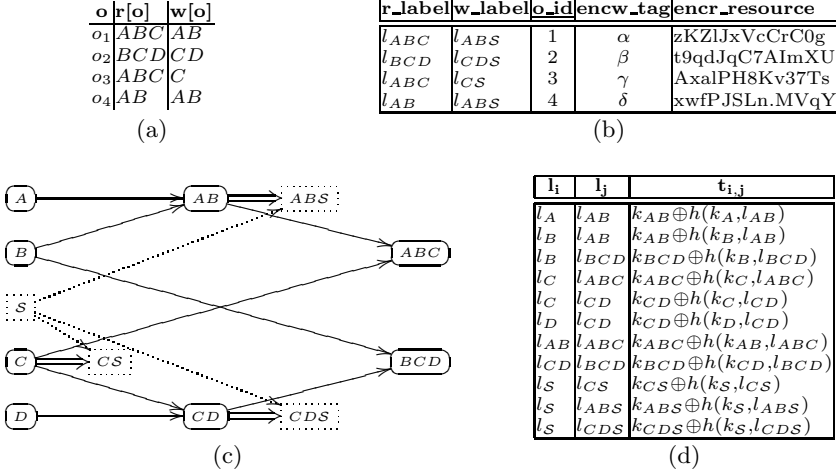


Fig. 3. An example of read and write acls (a), encrypted resources (b), key derivation structure (c), and tokens (d)

encw_tag: it is the write tag of the resource and is encrypted with the key identified by the label in w_label (i.e., $E(tag[o], k_{w[o] \cup \{S\}})$).

encr_resource: it is the encrypted version of the resource, encrypted with the key identified by the label in r_label (i.e., $E(o, k_{r[o]})$).

Let \mathcal{U} be the set of users and \mathcal{O} be the set of resources in the system, where each resource is associated with read and write access control lists as mentioned-above. To outsource her resources, the data owner must first compute keys and tokens forming the key derivation structure, by calling function **Define_Key_Derivation_Structure** in Figure 2. To this aim, she needs to define two families \mathcal{U}' and \mathcal{U}'' of subsets of users in \mathcal{U} . \mathcal{U}' corresponds to all the sets of users whose keys must be represented in the system for the correct enforcement of the authorizations. It then includes the singleton sets of users in \mathcal{U} , and the sets of users representing read and write access lists of resources in \mathcal{O} . \mathcal{U}'' is the subset of \mathcal{U}' representing those sets of users that have to share a key with the external server. It then includes all the sets corresponding to the write access lists of resources in \mathcal{O} . The data owner then:

- communicates to each user u key k_u and to the external server key k_S ;
- computes and stores at the external server the encrypted resources and the associated metadata as described above;
- stores at the external server all tokens in \mathcal{T} as a set of triples of the form $\langle l_i, l_j, t_{i,j} \rangle$ indicating that key with label l_j can be derived directly from key with label l_i through token $t_{i,j}$.

Example 3. Consider a system with four users $\mathcal{U}=\{A,B,C,D\}$ and four resources $\mathcal{O}=\{o_1,o_2,o_3,o_4\}$, and assume read and write acls of resources to be as in Figure 3(a) (read acls are the same as in Example 1). Figure 3(c) illustrates the

key derivation structure computed as described in Example 2. Figure 3(b) and Figure 3(d) illustrate the encrypted resources and associated metadata, and the public tokens outsourced to the external server, respectively.

Enforcement of read authorizations is automatically guaranteed as in the original proposal [8]: the key with which a resource is encrypted can be known only to users authorized to read the resource. Enforcement of write authorizations can be easily delegated to the server that will accept a write operation on a resource only if the user shows knowledge of the corresponding (plaintext) write tag. Since the write tag of the resource is encrypted with $k_{w[o] \cup \{S\}}$, besides the server only the authorized writers will be able to decrypt such an information. Also, since the server is assumed *honest-but-curious* (and therefore does not have interest to tamper with resources), this simple control allows us to enforce write authorizations. (More details in Section 4)

It is easy to see that our approach guarantees: *i*) the correct *read authorization enforcement*, because the content of a resource is visible only to users authorized to read the resource; *ii*) the correct *write authorization enforcement*, because the write tag of a resource is visible only to users authorized to write the resource; *iii*) the *write control* by the server, because the tag of a resource is visible to the server. This is formalized by the following theorem, whose proof is omitted for space constraints.

Theorem 2 (Correct enforcement of authorizations). *Let \mathcal{U} be a set of users, \mathcal{S} be an external server, \mathcal{O} be a set of resources with $r[o]$ and $w[o]$ the read and write access lists, respectively, of o . Our access control system satisfies the following conditions:*

1. $\forall u \in \mathcal{U}$ and $\forall o \in \mathcal{O}$, u can decrypt $encr_resource[o]$ iff $u \in r[o]$ (read authorization enforcement);
2. $\forall u \in \mathcal{U}$ and $\forall o \in \mathcal{O}$, u can decrypt $encw_tag[o]$ iff $u \in w[o]$ (write authorization enforcement);
3. $\forall o \in \mathcal{O}$, \mathcal{S} can decrypt $encw_tag[o]$ (write control).

4 Write Integrity Control and Resource Management

According to our reference scenario our complicating factor in policy enforcement is to ensure proper protection of the confidentiality of data, while the server can be assumed trustworthy to manage resources and delegated actions. Nevertheless, it is important to provide a means to the data owner to verify that server and users are behaving properly. Providing such a control has a double advantage: *i*) it allows detecting resource tampering, due to the server not performing the required check on the write tags or directly tampering with resources, and *ii*) it discourages improper behavior by the server and by the users since they know that their improper behavior can be easily detected, and their updates recognized as invalid and discarded. In this section, we illustrate our approach for providing the data owner with a means to verify that modifications to a resource

have been produced only by users authorized to write the resource. As discussed in the previous section, if the server performs the correct control on the write tags, such a property is automatically guaranteed. We therefore present how to perform a write integrity control to detect misbehavior (or laziness) by the server as well as misbehavior by users that can happen with the help of the server (not enforcing the control on the write tags since it is either colluding with the user or just behaving lazily) or without the help of the server (if the user improperly acquires the write tag for a resource by others).

A straightforward approach to provide such a write integrity control would be to apply a signature-based approach. This requires each user to have a pair (private,public) of keys and, when updating a resource, to sign the new resource content with her private key. The data owner can then check the write integrity by verifying that the signature associated with a resource correctly reflects the resource content and it is produced by a user authorized for the operation. Such an approach, while intuitive and simple, has however the main drawback of being computationally expensive (asymmetric encryption is considerably more expensive and therefore less efficient than symmetric encryption) and not well aligned with our approach, which - as a matter of fact - exploits symmetric encryption, tokens, and hash functions to provide efficiency in storage and processing. In the spirit of our approach, we then build our solution for controlling write integrity on HMAC functions [3].¹ In addition to the metadata illustrated in the previous section, we then associate with each resource three write integrity control fields:

- encw_ts:** it is the timestamp of the write operation, encrypted with the key $k_{w[o] \cup \{S\}}$ corresponding to the group including the server and all the users in the write access list of o (i.e., $E(ts, k_{w[o] \cup \{S\}})$);
- user_tag:** it is a HMAC computed with the key k_u of the user who performed the write operation over the resource concatenated with the user_tag of the resource prior to the write operation,² and the timestamp of the write operation (i.e., $\text{HMAC}(o||u_t'||ts, k_u)$);
- group_tag:** it is a HMAC computed with the key $k_{w[o]}$ corresponding to the write access list of o over the resource concatenated with the timestamp of the write operation (i.e., $\text{HMAC}(o||ts, k_{w[o]})$).

Figure 4 summarizes the information associated with each encrypted resource.

At time zero, when the data owner outsources her resources to the server, the values of the user_tag and group_tag are those computed by the owner with her own key for the user_tag, and with the key of the write access list of the resource (to which the owner clearly belongs) for the group_tag. Every time a user updates a resource, it also updates its user_tag and group_tag.

¹ For common platforms, the ratio between the execution times of digital signatures and of HMAC is more than three orders of magnitude.

² The reason for including the user_tag of the resource prior to the write operation is to provide the data owner with a hash chain connecting all the resource versions (we assume the server to never overwrite resources but to maintain all their versions).

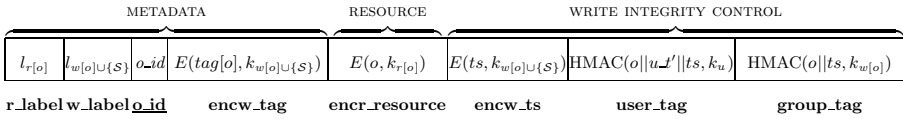


Fig. 4. Structure of outsourced resources

A `user_tag` is considered valid if it matches the resource content and it is produced by a user in the write access list of the resource. The `user_tag` provides write integrity (meaning the resource has been written by an authorized user) and accountability of user actions. In fact, since the data owner knows the key k_u of every user u (which she generated and distributed), she can check the validity of the `user_tag` and detect possible mismatches, corresponding to unauthorized writes. In addition, every write operation considered valid (according to the control on the `user_tag`) cannot be repudiated by the user u whose key k_u has generated the HMAC. The consideration of `group_tag` extends the ability of checking the validity of the write operations (i.e., write integrity) also to all the users in the write access list of the resource. To guarantee the integrity of the metadata associated with each resource and that no resource is dropped from the system, we adopt traditional approaches based on aggregate signatures [13] (i.e., the first four fields in Figure 4) are signed as a whole by the data owner, providing efficient and scalable integrity check for the structure with one signature).

While we assume the server to be trustworthy and therefore not interested in tampering with the resources, we note that the `user_tag` would allow also to detect possible tampering of the server with the resource (since not being an authorized writer, the server will not be able to produce a valid `user_tag`). The server could also tamper with the write authorizations, by decrypting the write tag and encrypting it with the key corresponding to a different write access list. However, the improper inclusion of a user in the write access list does not have any different effect than when the server does not perform the control, since the user improperly included in the write access list will not be able to produce a valid `user_tag`. Analogously, the improper removal of a user from the write access list has the same effects as when the server refuses its services.

Unauthorized write operations in the case of a well behaving server can only happen if a user has improperly acquired or received from other authorized users the write tag of a resource. Whichever the case, the user will be able to provide neither a valid `user_tag` nor a valid `group_tag` for the resource. Also, the data owner and any user authorized to write the resource will be able to detect the invalidity of the `group_tag`, since the key used to compute the HMAC will not correspond to the key of $w[o]$.

5 Related Work

In the last few years, several research efforts have been devoted to enable data owners to outsource the storage and management of their data to possibly non-fully trusted third parties [16]. Most proposals have addressed the problem of efficiently performing queries on outsourced encrypted data, without

decrypting sensitive information at the server side (e.g., [15,7,10,17]), or the problem of protecting data integrity and authenticity (e.g., [11,13,14,18]). The problem of enforcing an authorization policy on outsourced data is orthogonal to these issues and has recently received the attention of the research community (e.g., [8,12,19]). The proposal in [12] protects access to XML documents by encrypting different portions of the XML tree using different keys. The solution in [8] combines selective encryption and key derivation strategies for controlling accesses to outsourced resources. This approach also permits to delegate to the server the management of policy updates. More recently, attribute-based encryption has been proposed for providing system scalability [19].

All the approaches above consider read access privileges only and few works have addressed the issue of enforcing write privileges. Raykova et al. [15] adopt selective encryption to enforce read and write privileges on outsourced data. The authors introduce a two-layer access control model. The server restricts read/write accesses at the level of block. To enforce the authorization policy at the granularity of resources within blocks, the proposed system adopts asymmetric encryption and defines two key derivation hierarchies: one for private keys (to enforce read privileges) and one for public keys (to enforce write privileges). This solution cannot easily enforce policies where resources with the same read access control policy can be modified by two different sets of users. In this case, the approach in [15] can be adapted by associating different pairs of keys to the same group of users, thus increasing key management overhead. Zhao et al. [20] propose attribute-based encryption and attribute-based signature techniques to enforce read and write access privileges, respectively. This approach requires the presence of a trusted party for correct policy enforcement. Also, attribute-based techniques are computationally more expensive than traditional symmetric encryption.

6 Conclusions

In this paper, we presented an approach for supporting both read and write privileges on outsourced encrypted data. The proposed solution relies on the use of symmetric encryption, hashing, and HMAC functions for enforcing access control in an efficient and effective way. Our proposal performs then a step towards the development of solutions actually applicable to real-world scenarios where efficiency and scalability are mandatory.

Acknowledgements. This work was partially supported by the EC within the 7FP, under grant agreement 257129 (PoSecCo), by the Italian Ministry of Research within the PRIN 2008 project “PEPPER” (2008SY2PH4), and by the Università degli Studi di Milano within the project “PREVIOUS”. The work of Sushil Jajodia was partially supported by the National Science Foundation under grants CCF-1037987 and CT- 20013A.

References

1. Agrawal, R., Kierman, J., Srikant, R., Xu, Y.: Order preserving encryption for numeric data. In: Proc. of SIGMOD 2004, Paris, France (June 2004)

2. Atallah, M., Frikken, K., Blanton, M.: Dynamic and efficient key management for access hierarchies. In: Proc. of CCS 2005, Alexandria, VA, USA (November 2005)
3. Bellare, M., Canetti, R., Krawczyk, H.: Keying Hash Functions for Message Authentication. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 1–15. Springer, Heidelberg (1996)
4. Cimato, S., Gamassi, M., Piuri, V., Sassi, R., Scotti, F.: Privacy-aware biometrics: Design and implementation of a multimodal verification system. In: Proc. of ACSAC 2008, Anaheim, CA, USA (December 2008)
5. Damiani, E., De Capitani di Vimercati, S., Jajodia, S., Paraboschi, S., Samarati, P.: Balancing confidentiality and efficiency in untrusted relational DBMSs. In: Proc. of CCS 2003, Washington, DC, USA (October 2003)
6. Damiani, E., De Capitani di Vimercati, S., Paraboschi, S., Samarati, P.: Fine grained access control for SOAP e-services. In: Proc. of WWW 2001, Hong Kong, China (May 2001)
7. De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: A data outsourcing architecture combining cryptography and access control. In: Proc. of CSAW 2007, Fairfax, VA, USA (November 2007)
8. De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Encryption policies for regulating access to outsourced data. ACM TODS 35(2), 12:1–12:46 (2010)
9. Gamassi, M., Piuri, V., Sana, D., Scotti, F.: Robust fingerprint detection for access control. In: Proc. of RoboCare 2005, Rome, Italy (May 2005)
10. Hacıgümüş, H., Iyer, B., Mehrotra, S., Li, C.: Executing SQL over encrypted data in the database-service-provider model. In: Proc. of the SIGMOD 2002, Madison, WI, USA (June 2002)
11. Merkle, R.C.: A Certified Digital Signature. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 218–238. Springer, Heidelberg (1990)
12. Miklau, G., Suci, D.: Controlling access to published data using cryptography. In: Proc. of VLDB 2003, Berlin, Germany (September 2003)
13. Mykletun, E., Narasimha, M., Tsudik, G.: Authentication and integrity in outsourced databases. ACM TOS 2(2), 107–138 (2006)
14. Pang, H., Jain, A., Ramamritham, K., Tan, K.: Verifying completeness of relational query results in data publishing. In: Proc. of SIGMOD 2005, Baltimore, MA, USA (June 2005)
15. Raykova, M., Zhao, H., Bellovin, S.: Privacy enhanced access control for outsourced data sharing. In: Proc. of FC 2012, Bonaire (February-March 2012)
16. Samarati, P., De Capitani di Vimercati, S.: Data protection in outsourcing scenarios: Issues and directions. In: Proc. of ASIACCS 2010, China (April 2010)
17. Wang, H., Lakshmanan, L.V.S.: Efficient secure query evaluation over encrypted XML databases. In: Proc. of VLDB 2006, Seoul, Korea (September 2006)
18. Xie, M., Wang, H., Yin, J., Meng, X.: Integrity auditing of outsourced data. In: Proc. of VLDB 2007, Vienna, Austria (September 2007)
19. Yu, S., Wang, C., Ren, K., Lou, W.: Achieving secure, scalable, and fine-grained data access control in cloud computing. In: Proc. of INFOCOM 2010, San Diego, CA, USA (March 2010)
20. Zhao, F., Nishide, T., Sakurai, K.: Realizing Fine-Grained and Flexible Access Control to Outsourced Data with Attribute-Based Cryptosystems. In: Bao, F., Weng, J. (eds.) ISPEC 2011. LNCS, vol. 6672, pp. 83–97. Springer, Heidelberg (2011)

Malicious Users' Transactions: Tackling Insider Threat

Weihan Li, Brajendra Panda, and Qussai Yaseen

Department of Computer Science and Computer Engineering
University of Arkansas
Fayetteville, USA 72701
{wx1011, bpanda, qyaseen}@uark.edu

Abstract. This paper investigates the issues of malicious transactions by insiders in database systems. It establishes a number of rule sets to constrain the relationship between data items and transactions. A type of graph, called *Predictive Dependency Graph*, has been developed to determine data flow patterns among data items. This helps in foretelling which operation of a transaction has the ability to subsequently affect a sensitive data item. In addition, the paper proposes a mechanism to monitor suspicious insiders' activities and potential harm to the database. With the help of the Predictive Dependency Graphs, the presented model predicts and prevents potential damage caused by malicious transactions.

Keywords: Insider Threat, Transactions, Database, Security.

1 Introduction

Organizations face the continual possibility of outside (external) and inside (internal) attacks. In the current context that risks and chances of malicious attacks from intruders are limitless, it is very important that database systems that store critical data provide confidentiality, integrity and availability. Although a lot of work in security is related to the outsider threat problem, the growing reliance on technological infrastructures has made organizations increasingly vulnerable to threats from insiders. Identification of insider threats is a major challenge, because by definition [1], given a wide range of activities, insiders have been granted certain authority and trust. In addition, they have superior knowledge of the organization's inner control and security systems.

Although insiders can perform tasks as authorized users, they should not manipulate data items arbitrarily, but only in constrained ways that preserve or ensure the integrity of the data item. Since there are different sets of data items that are necessarily associated with a set of certain transactions permitted to manipulate them, even though the transactions could be from authorized users, any data item modifications should be constricted to the transactions assigned, and those transactions should be executed following the correct sequence [2].

This paper considers transactions that inject malicious attacks into the system and corrupt other transactions and data items. The threat mitigation technique presented in

this paper attempts to remove the malicious effects of an insider and to provide mitigation service. The contribution of this paper is summarized as follows. First, Normal Activity Rules are established to constrain the relationship between data items and transactions that depend upon different security levels. Second, a graph called Predictive Dependency Graph is developed to investigate different patterns of data flows among the data items. Third, The Labeling Mechanism is adopted to flag any unverified data items in order to predict and stop any malicious data flow. Fourth, the idea of a Real Event Analyzer (REA) is used to check the transactions' effect actions matching to real world. The advantage of using the REA is that it permits monitoring of the suspicious users' activities and potential harm to the database.

The rest of the paper is organized as follows. Section 2 discusses some related work. Section 3 introduces the insider threat problem and describes the proposed model. Finally, section 4 provides the conclusions and mentions some future work.

2 Related Work

The increasing awareness of the insider threat problem has led to significant interest in techniques which can effectively detect malicious insiders' activities. Unfortunately, the techniques for mitigating insider threat are relatively immature and have not gained wide acceptance or use.

A significant amount of research on intrusion detection systems has been conducted for almost twenty years [3][4][5][6][7][8][9], however, most of them focus on the capability of identifying intrusions by outsiders. In recent years, several works have been done in the area of detecting insider threats [10][11]. However, they are not adequate for protecting the database from "denial of service" and they tend to generate low "false alarms". Yaseen and Panda [12][13] discussed how insiders can use dependencies to infer sensitive information and make malicious modifications in sensitive data items. In addition, they proposed models to detect and prevent those types of insider attacks.

Database Activity Monitoring [14][15] is an emerging technology that monitors and analyzes individuals or organizations' databases for potential leaks and compromises. Database Activity Monitoring operates independently of the database management system and does not rely on any form of native (DBMS-resident) auditing or native logs such as trace or transaction logs. Database Activity Monitors capture and record, at a minimum, all Structured Query Language (SQL) activity in real time or near real time, including database administrator activity across multiple database platforms, and can generate alerts on policy violations. There are several DAM products on the market. This paper adopts the concept of DAM to propose a new Real Event Analyzer with similar functionality. The architecture technique, management policy and other features are beyond the discussion of this work.

3 Insider Threat and Malicious Transactions

The goal of this paper is to design models that can detect and stop malicious transactions submitted to the DBMS by a malicious insider masquerading as a

legitimate user. Since there can only be a finite set of tasks to be performed for inside users, this paper assumes that the “Normal Activity Rules” (discussed in the next section), which are predefined and established respectively by the application program and administrators /database experts, are stored separately for predicting any further damage. Before going further, let us introduce some definitions.

Definition 1. A *Critical Data Item (CDI)* is defined as a data item to which a change would induce significant effect. The CDIs therefore are usually the target of malicious attackers. Besides, any data items outside the CDI set are therefore termed as *Regular Data Item Set (RDI)*.

Definition 2. A *Predefined Transaction* is an authorized transaction that is allowed to update data items to which it is assigned. A Predefined transaction is categorized into two different types:

1. **Predefined Trusted Transaction (PTT)** is the transaction that has authorization to access both critical data items and regular data items, and it is executed by trusted users such as DB Administrators or other high-level employees.
2. **Predefined Normal Transaction (PNT)** is the transaction that can access and update only regular data items, and it is executed by normal-level users who might occasionally spread the potential damage, or a malicious normal-level user who intends to carry out malicious activities by breaching the system security.

For each transaction that has access to a mount of data items, whether *Critical Data Item Set* and/or *Regular Data Item Set*, those data items in that transaction are categorized with relation pairs of *Read Set* and *Write Set*. *Data dependency* is defined as follow.

Definition 3. A write operation $w_i[x]$ of a transaction T_i is *dependent on* a read operation $r_i[y]$ of T_i if $w_i[x]$ is computed using the value obtained by $r_i[y]$. A data value v_1 is dependent on another data value v_2 if the write operation that wrote v_1 was dependent on a read operation on v_2 . Notice that v_1 and v_2 may be two different versions of same data item.

The semantics information is insufficient; it is reasonable to presume that a write operation of a transaction depends on all read operations of the same transaction that precede the write operation in a particular pair in transaction T . For example, the Write Set containing write operation $w[x]$ is dependent on a set of data items in the corresponding Read Set if $x=f(\text{Read Set})$, the values of data items in that Read Set are used in calculating the new value of x (Previous value of x is the value before current operation). There are three cases for the set of data items in Read Set, which are as follows.

Case1: Read Set = \emptyset . This means that no data item is used in calculating the new value of x . Such an operation is denoted as a fresh write. If $w[x]$ is a fresh write and if

the previous value of x is updated maliciously, the value of x will be refreshed after this write operation.

Case2: $x \notin$ Read Set. Then $w[x]$ is called a blind write. If $w[x]$ is a blind write and if the previous value of x is updated by a malicious transaction and none of the data items in Read Set are updated maliciously, then the value of x will be refreshed after this write operation. If any data items in Read Set are modified, the value of x will be modified.

Case3: $x \in$ Read Set. If the previous value of x is updated maliciously, then x remains contaminated. Otherwise, if any other item in Read Set is contaminated, x is thus affected.

These actually show the data dependency between the data items in related Read Set and the Write Set. The dependency of different relations of Read and Write sets, defined as Normal Activity States, are identified and denoted like this: $\{(R_Set_1; W_Set_1), (R_Set_2; W_Set_2), \dots, (R_Set_n; W_Set_n)\}$. For instance, consider a transaction T with query statements in the given sequence as shown below:

UPDATE Table1 set x = a + b + x;
UPDATE Table1 set y = x + u;
*UPDATE Table1 set z = v * 0.9;*

After analyzing the transaction, the Normal Activity States of transaction T is as following:

$T: \{(\{a, b, x\}: \{x\}), (\{x, u\}: \{y\}), (\{v\}: \{z\})\}$

The transaction T consists of three related pairs. Each pair represents the data dependency that once any data in the Read Set is modified, all or parts of the data items in the Write Set of the same pair are affected. Consider that once data item x in the Read Set is about to be modified by a previous transaction T' . Both the data items y and x in the related Write Sets of T are considered to be potentially affected when the transaction T' is in the process of being executed. This is because both data items x and y are partially dependent on data item x . So if any one or more data item sets in Read Set of T is/are updated by a previous transaction T' , the value of at least one data item in the Write Set will be presumably influenced. This kind of situation is considered as potential damage to the critical data items in the database system.

Definition 4. The *Predictive Dependency Graph PDG* is a graph adopted to demonstrate the prediction of a particular transaction's impact or pathway in the database.

Figure 1 shows an example of a PDG. A rectangular node in the graph contains information such as transaction's ID and its operation sequence; an oval node in the graph denotes the data items in the Write Set of the transaction from which the arrows

come. A firm edge from transaction to data item represents that transaction updates the data item to which the arrow points. The predictable transactions which might probably be affected by data items being updated are denoted by a dash edge from data item to transaction. It is necessary to mention that the subscript for each transaction only means the transaction id in the Normal Activity States not the time sequence of transactions. As seen in Figure 1, the procedure starts from the rectangular node of transaction, and ends at rectangular nodes which represent all the affected transactions presumably determined by data dependencies.

Definition 5. *Average Time, $time_{avg}$* , is defined as the average time needed for the execution of a particular transaction, which is measured using the following formula:

$$time_{avg} = \frac{\sum_{i=1}^N (t_{i-exe} - t_{i-sch})}{N} \tag{1}$$

Where t_{i-exe} denotes the time when it was executed and t_{i-sch} represents the time when it was scheduled, N is the number of total transactions executed.

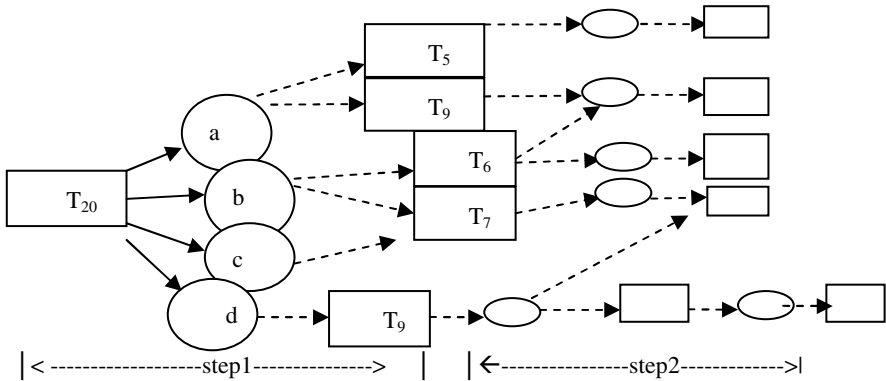


Fig. 1. Predictive Dependency Graph

Definition 6. *Assurance time t_A* is preset by the database experts and is defined to be used with the information of the average time of execution of particular transaction, which is measured using the following formula.

$$t_A = \frac{\sum_{i=1}^N (t_{i-exe} - t_{i-sch})}{N} * Avg_{Ts} * n_{STEP} \tag{2}$$

Where N is the number of steps used for generating the PDG graph and Avg_{Ts} represents the average number of potential affected transactions in each step.

Notice that in the proposed PDG graph, the one complete procedure of a step starts with the rectangular node of transaction and passes through the circular nodes of data items in its Write Set. Hence, in order to predict any further potential damage from an incoming transaction, the Predictive Dependency Graph is produced based on the

combined information of real execution time $time_{avg}$ and threshold number n . Hence, for each incoming transaction T , the possibility of potential damage (as mentioned on the previous page) is predicted and measured within the time period of t_A :

Definition 7. *Normal Activity Rules* is a predefined profile that contains a group of relation rules, each relation is of the form:

Rule, Transaction, Read Set, Write Set, Potential Damage, Assessment, Action,

where a transaction is categorized by different user's security levels, and Read and Write Sets are the constrained accessible data item sets that correspond to that security level.

Based on definition 1 and 4, there is a number of Preformed Transaction and Read/Write Set rules in the Normal Activity Rules Table in Figure 2. Predefined Trusted Transactions and Predefined Normal Transactions are denoted by PTTs and PNTs respectively. RDI and CDI represent Regular Data Item and Critical Data Item. We should mention here that *Normal Activity Rules* are constrained by application program or database administrator and is based on security issues and users' privileges.

For the Predefined Normal Transaction, it only has authorization over access to and update of the RDI Set. Since PNT is needed to be judged whether it is from an innocent insider or a masqueraded insider, PNT should be examined before scheduling it to commit. In the real world, however, thousands of PNT from different users can be executed per second in the database application program. In order to avoid slowing down the system, transactions are scheduled to be executed and examined simultaneously. Meanwhile, since the Predefined Trusted Transactions are considered from trusted user by default, any access or update merely within the range of CDI Set can be executed without any supervising action. However, any PTT update, involving both the Critical Data Item Set and at least one other data item from Regular Data Item Set, is needed to be checked carefully and thoroughly. It is the utmost

Rules	Transaction	Read Set	Write Set	Potential Damage	Action
1	PNT	RDI	RDI	indirectly access CDI	verify R Set, then predict damage
2	PNT	RDI	CDI	directly access CDI	not comply to rules/states, reject
3	PNT	RDI/CDI	CDI,RDI	directly access CDI	not comply to rules/states, reject
4	PNT	RDI, CDI	CDI/ RDI	directly access CDI	not comply to rules/states, reject
5	PTT	RDI	RDI	indirectly access CDI	verify R Set, then predict damage
6	PTT	RDI	CDI	directly access CDI	verify R Set
7	PTT	CDI	CDI,RDI	No	allowed
8	PTT	CDI	CDI/RDI	No	allowed
9	PTT	RDI,CDI	RDI	indirectly access CDI	verify R Set, then predict damage
10	PTT	RDI,CDI	CDI	directly access CDI	verify R Set

Fig. 2. Normal Activity Rules Table

requirement since before allowing any update, all data items to be read are verified to make sure that there are no damaged data from previous transactions. In addition, no more potential damage in future can result from the execution.

3.1 The Proposed Model

The goal of this model is to develop a Predictive Insider Threat Detection Mechanism that can be implemented within the database server and is capable of detecting any anomalous users' requests to a DBMS. Suppose that the Normal Activities States are as shown in Figure 3 (for each T_x , x is the transaction id, not the sequence of events). For instance, PNT Transaction T_5 has 2 pairs of Read Sets and Write Sets. Once a data item m is updated in the previous PNT Transaction T' , data items l and a are all considered to be potentially affected. On the other hand, once data item l in pair 2 is updated beforehand, only data item l in the related Write Set will be presumed to be the affected one. In PTT Transaction T_{200} , the data item z in bold is defined as a critical data item, and this PTT Transaction T_{200} follows the Rule 10 in the Table in Figure 2, all the other Transactions follow the Rule 1.

In Figure 4, suppose the incoming PNT Transaction T is the one that is to be scheduled to update data items a, b, d , and data item z is a critical data item. There are N steps after any updates on them. Similarly there are also countless transactions and data items that need to be checked. However, in this model, only 2 steps are analyzed for that particular Transaction T which might change data item a, b, d . From this graph, once the PNT Transaction T on data items a, b , and d is executed, there is a possibility that the critical data item z might be manipulated by the effect of previous events through T_7 to T_{200} .

T_2 : {{a}:{m,o}}
T_3 : {{a,f}:{c,f}}
T_4 : {{a,b}:{c,d}}
T_5 : {{m,k,l}:{l}}, {{m,k,l}:{l}}
T_7 : {{m,j}:{t}}
T_8 : {{p,o}:{s,n}}
T_{12} : {{b,d}:{p,q}}
T_{15} : {{a,d}:{d,q}}
T_{20} : {{q,l}:{g,l}}, {{q,l}:{m}}
.....
.....
T_{200} : {{t,z}:{z}}

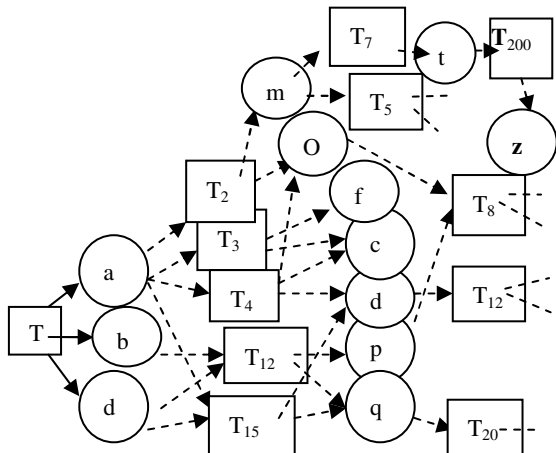


Fig. 3. Example of Normal Activities States

Fig. 4. Predictive Dependency Graph Model

However, since only the first 2 steps are checked in this graph, the PNT Transaction is considered to be safe in the time range we preset. If we further analyze it to 3 steps, T_{200} might be detected since it has indirect potential damage on critical data item z . In light of the situation of Transaction T_3 and T_{15} , for which the same data item appears in both the Read Set and Write Set, in order to keep the limited account of the considered affected transactions, the predictive dependency between the data item and the transaction which updates the same data item is not considered in the model.

3.2 The Labeling Mechanism

The data item is labeled and stored into the unverified data items set in two situations. The first situation is that the PNT Transaction whose Read Set is all verified and is still considered to be a threat. This is because it gets too close to CDI Set and all the following transactions needs to be alerted by labeling process before any direct damage to CDI. Accordingly, all of the data items in the Write Set of that PNT Transaction are initially labeled as unverified “dirty” data items and stored once it gets

executed. The second situation is when a damage spread might happen. In order to keep the both efficiency and availability of database and store the further impact of the to-be-verified data item, some of the data items in the Write Set whose corresponding Read Set are not verified are labeled as unverified “dirty” data items.

3.3 The Damage Predictor

The Damage Predictor is used to predict any potential impact on data items in the Critical Data Item Set within a certain time of period t_A . For all the transactions, PNTs or PTTs, which are submitted to the database application program, before scheduling them to be executed, the prediction of affected transactions and its data items can be made by using the Damage Predictor. First, the data items in the Read Set of that particular Transaction are checked and verified, and then, data dependency relation among other transactions in the Normal Activity States list is found. Second, a PDG graph can be produced with all the possible paths of the dataflow initialed from the incoming transaction. With enough information and in cooperation with the threshold time t_A , if the PDG graph of incoming transactions show indirect access to the CDI, or as also termed “potential damage”, the transaction might be considered to be from a user with malicious motivation, and thus, a more strict and harsh action will respond to it. For the PTT transaction that has more than one related pair, there might be more complicated situations that may hinder the Damage Predictor process. For example, since PTT has privilege over the CDI Set, the Read Set of PTT can either be the combination of CDI and RDI, or only contain the RDI. In this case, in order to simplify the process, the graph is checked using the same strategy instead of checking the pairs separately.

3.4 Real Events Analyzer

The Real Event Analyzer is used separately during the transaction activity-monitor process of the proposed Insider Detection System. In order to detect an attack, the

Real Event Analyzer will examine the real events that correspond to each query and judge whether the transaction is from a legitimate user based on the query information. If the presumed malicious transaction turns out to be innocent, all the data items in the Write Set of that transaction will be cleared and all the labels will be removed from the unverified data item set. Another function for the Real Events Analyzer is its ability to calculate the recovery time for the cleaning process of each data item x . So once potential damage is found in the PDG within the time period t_A , the average recovery time t_R should be checked:

$$t_R = \sum_{i=1}^n (t_{i-max} + t_{i-min}) * Avg_{OP} / 2 \tag{3}$$

where t_{i-max} and t_{i-min} are the maximum and minimum recovery time of labeled data items for a single step respectively, and Avg_{OP} is the average number of operations.

The recovery time t_R is calculated as average time length used for recovering data items during each step and t . If the recovery time t_R for the data item is beyond the assurance time t_A , which means that the unverified data items usually cannot be cleaned by the time it has spread the damage to the CDI indirectly, the transaction will be rejected. The advantage of the Real Event Analyzer lies in the fact that the examination process is arranged after the execution of the transaction, thus the speed of the DBMS system can be guaranteed. For those PTT Transactions whose Read_Set are not cleared, the transactions will be on hold unless all the data items are safe to use.

3.5 System Architecture

Figure 5 shows the architecture of the proposed approach. It has two main processing parts: the Damage Predictor and the Real Event Analyzer. Transactions are first compared with Normal Activities States (NAS) and sorted into PNT and PTT Transactions for further detection. The Damage Predictor processes transactions and generates Predictive Dependency Graph to predict potential damage and to report alerts to the Real Event Analyzer and the Scheduler. The Real Event Analyzer (REA) examines the executed transaction and verifies the data items, and then reports back to the scheduler.

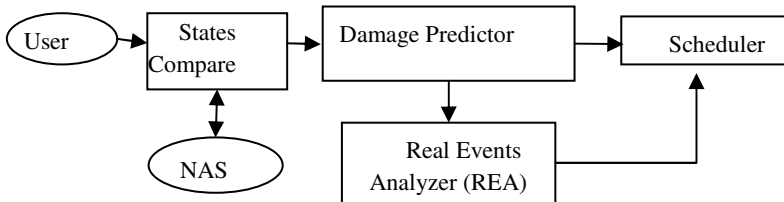


Fig. 5. System Architecture

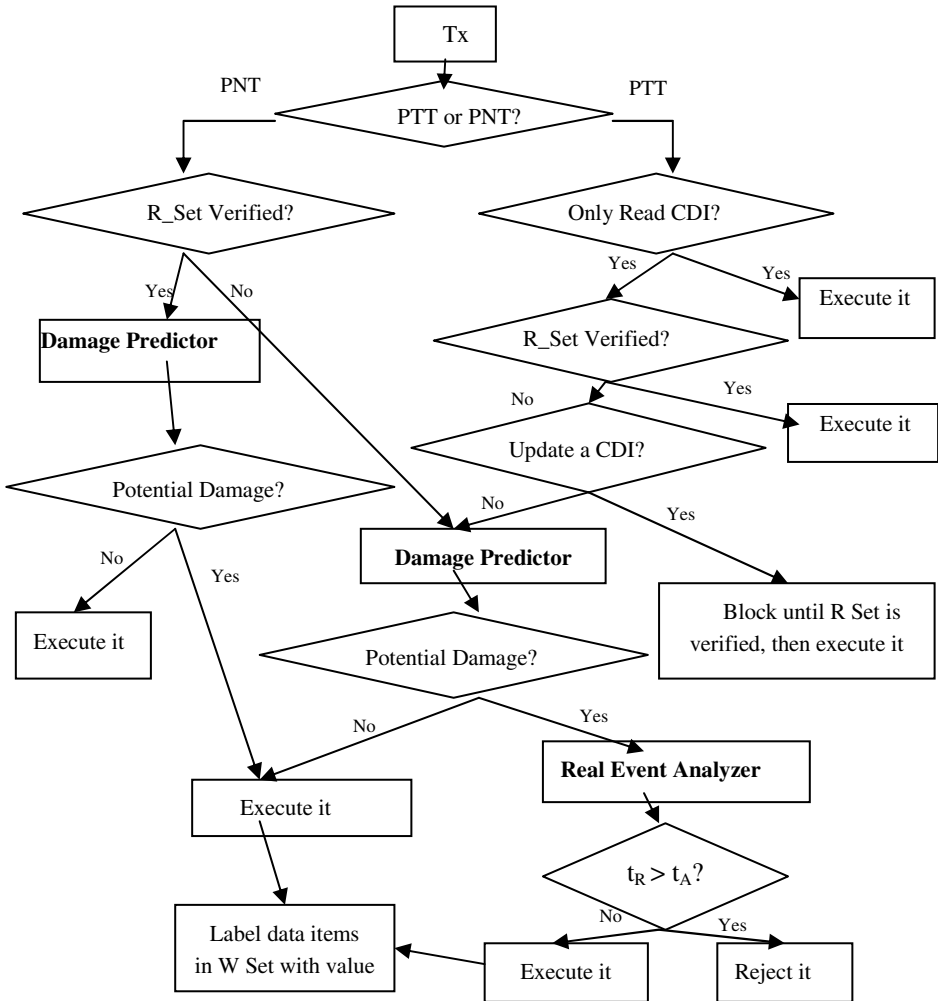


Fig. 6. Dataflow Chart of the Model

Figure 6 shows the flow chart incorporated with Damage Predictor and Real Events Analyzer. The Transaction Scheduler schedules PNT or PTT as to be executed or not. The ID mechanism is adopted to identify if the transaction is from a normal user or a trusted high level user. If the transaction is PNT, all Read Sets in R/W Set pairs are checked to see if there is any “bad” data item x which is damaged in previous executed transaction. If the system is not sure whether the transaction is from an innocent user, the labeling mechanism is initialized after the execution of T_x , so that the data items in the Write Sets will be cleaned during the next verification process. The time t_R and t_A are generated in the Damage Predictor and the Real Events Analyzer respectively. For instance, if threshold t_A is set to 15 seconds when the PDG is generated, and recovery time t_R is 12 seconds by consulting the individual recovery

time for data items, such transaction will be rejected since once data flow reaches the Critical Data Item Set, the verification is still in process and the value of the Critical Data Item will be compromised. In the case of the transaction from trusted user (PTT), if the transaction only reads data items from CDI (like the rules 7 and 8 in Figure 2), it will be executed without any check. If the transaction has direct access to CDI (like the rules 6 and 10 in Figure 2), the transaction should wait until all the labeled data items in Read Sets to be verified by REA. The system must be aware of the data flows between trusted users and suspicious users. Thus, the identity of the transactions and data items that T_x depends upon needs to be checked beforehand. Moreover, since Predefined Normal Transactions play an important role numerically in the database system, and any “block” action might significantly reduce the efficiency, labeling mechanism and the Real Events Analyzer are adopted here, instead of simply discarding the normal transaction. Meanwhile, on the other side of the Presumed Trusted Transaction, identification is also a mandatory step in the proposed approach. In addition, since critical data items always have extreme influence on regular data items and it is assumed that all the PTTs are from the trusted users that take more responsibility. CDIs should avoid any damage flow from RDI before any PTT is executed.

4 Conclusions

This paper has addressed the problem of malicious transactions by insiders in database systems. It has established a number of rule sets to constrain the relationship between data items and transactions, which are called Normal Activity Rules. Moreover, it has used a graph model, called Predictive Dependency Graph (PDG), to determine data flow patterns among data items. The PDG has been used to look for the data items that might be affected once an update to a particular set of data items occurs. That is, PDG has been used to detect any indirect damage to particular data items. The Labeling Mechanism and Real Event Analyzer have been used to detect and prevent malicious transactions. The Labeling Mechanism has been employed to flag any unverified data items in order to predict and stop any malicious data flow. The Real Event Analyzer has been used to permit monitoring of suspicious users' activities and potential harm to databases. As a future work, we plan to formally define and develop the Real Event Analyzer and analyze its performance.

Acknowledgments. This work has been supported in part by US AFOSR under grant FA 9550-08-1-0255. We are thankful to Dr. Robert. L. Herklotz for his support, which made this work possible.

References

1. Mills, R.F., Peterson, G.L., Grimaila, M.R.: Insider Threat Prevention, Detection and Mitigation. In: Knapp, K.J. (ed.) *Cyber Security and Global Information Assurance: Threat Analysis and Response Solution*. U.S. Air Force Academy, Colorado, USA (2009)

2. Clark, D., Wilson, D.: A comparison of Commercial and Military Computer Security Policies. In: IEEE Symposium on Security & Privacy (1987)
3. Chung, C.Y., Gertz, M., Levitt, K.: Demids: A misuse detection system for database systems. In: 14th IFIP WG11.3 Working Conference on Database and Application Security (2000)
4. Lee, S.Y., Low, W.L., Wong, P.Y.: Learning Fingerprints for a Database Intrusion Detection System. In: 7th European Symposium on Research in Computer Security (2002)
5. Kamra, A., Bertino, E., Terzi, E.: Detecting anomalous access patterns in relational databases. *The International Journal on Very Large Data Bases* 17(5), 1063–1077 (2008)
6. Hu, Y., Panda, B.: Design and Analysis of Techniques for Detection of Malicious Activities in Database System. *Journal of Network and System Management* 13(3), 111–125 (2005)
7. Lee, W., Stolfo, S.J.: A framework for constructing features and models for intrusion detection system. *ACM Transactions on Information and System Security* 3(4), 227–261 (2000)
8. Meng, Y., Liu, P., Zang, W.: Multi-Version Attack Recovery for Workflow Systems. In: Proceedings of the 19th Annual Computer Security Applications Conference (2003)
9. Srivastava, A., Surai, S., Majumbar, A.K.: Weighted Intra Transaction Rules Mining for Database Intrusion Detection. In: Proceedings of the Pacific-Asia Knowledge Discovery and Data Mining (2006)
10. Ray, I., Poolsappasit, N.: Using Attack Trees to Identify Malicious Attacks from Authorized Insiders. In: Proceedings of the 10th European Symposium on Research in Computer Security (2005)
11. Martinez-Moyano, I., Rich, E., Conrad, S., Anderson, D.F., Stewart, T.R.: A Behavioral Theory of Insider-Threat Risk: A System Dynamic Approach. *ACM Transactions on Modeling and Computer Simulation* 18(2) (2008)
12. Yaseen, Q., Panda, B.: Predicting and Preventing Insider Threat in Relational Database Systems. In: Samarati, P., Tunstall, M., Posegga, J., Markantonakis, K., Sauveron, D. (eds.) WISTP 2010. LNCS, vol. 6033, pp. 368–383. Springer, Heidelberg (2010)
13. Yaseen, Q., Panda, B.: Malicious Modification attacks by Insiders in Relational Databases: Prediction and Prevention. In: 2nd IEEE International Conference on Information Privacy, Security, Risk and Trust (2010)
14. Newman, A.: Database Activity Monitoring: Intrusion Detection & Security Auditing. DAM Whitepaper, http://www.appsecinc.com/presentations/DAM_wp82305.pdf
15. Mogull, R.: Understanding and Selecting a Database Activity Monitoring Solution, <http://www.securosis.com/reports/DAM-Whitepaper-final.pdf>

Privacy-Preserving Television Audience Measurement Using Smart TVs

George Drosatos, Aimilia Tasidou, and Pavlos S. Efraimidis

Dept. of Electrical and Computer Engineering, Democritus University
of Thrace, University Campus, 67100 Xanthi, Greece

{gdrosato, atasidou, pefraimi}@ee.duth.gr

Abstract. Internet-enabled television systems, often referred to as Smart TVs, are a new development in television and home entertainment technologies. In this work, we propose a new, privacy-preserving, approach for Television Audience Measurement (TAM), utilizing the capabilities of the Smart TV technologies. We propose a novel application to calculate aggregate audience measurements using Smart TV computation capabilities and permanent Internet access. Cryptographic techniques, including homomorphic encryption and zero-knowledge proofs, are used to ensure both that the privacy of the participating individuals is preserved and that the computed results are valid. Additionally, participants can be compensated for sharing their information. Preliminary experimental results on an Android-based Smart TV platform show the viability of the approach.

Keywords: Privacy, Television Audience Measurement (TAM), Smart TV, Privacy-preserving Data Aggregation, Economics of Privacy.

1 Introduction

Television is nowadays one of the dominant mediums for information and entertainment. Information about television audiences provide valuable insights to broadcasters and the advertising industry on recent trends. Television Audience Measurement (TAM) systems aim at calculating qualitative and quantitative TV audience measurements. For example, Nielsen¹, one of the leading companies in the field of media audience measurement, uses measurements from approximately 18,000 households² in the U.S.A. to create the estimates the TV networks use. The viewer data is collected by the special metering equipment installed on the TV sets of the participating households; this data is transferred directly to the company's servers. Apparently, the above measurement process raises important privacy issues for the participants. A person's viewing record can reveal sensitive information about the person's preferences and habits. A privacy-preserving method for creating accountable TAMs is needed, in order to

¹ <http://www.agbnielsen.net>

² February 2010.

utilize television ratings information, while protecting the participants' privacy. Additionally, since TAM data bring important financial benefits to the industry (broadcasters, advertising companies, commercial products and more), some kind of fair financial compensation should be offered to the users that provide their viewing records.

Advances in communication and entertainment technologies have recently led to the introduction of Smart TVs, which are Internet enabled devices that support standard computer functionality (i.e., calculations, application execution, etc). This combination of traditional TV functionality with computational and networking capabilities, makes Smart TV technology capable of a whole new set of applications.

In this work, we present PrivTAM, a system for privacy-preserving TAM using Smart TV technology. The core of PrivTAM is a privacy-preserving cryptographic protocol, which accepts as input the viewing records from users' Smart TVs and performs secure multi-party computations [23] to calculate the TAMs. PrivTAM satisfies the following requirements for a reliable, privacy-preserving, TAM:

- Privacy - all records must be secret.
- Completeness - all valid records must be counted correctly.
- Soundness - dishonest records cannot disrupt the measurement process.
- Unreusability - no user can submit their record more than once.
- Eligibility - only those who are allowed to participate can submit their records.
- Verifiability - nobody can falsify the result of the TAM process.

The above requirements are a subset of the typical requirements of e-voting systems [15] and thus, our system borrows techniques from this field [4,9,12]. In addition, functionalities for the financial compensation of the participants are supported. The computations of PrivTAM are performed between software agents, which are located at the participants' Smart TVs, and a Trusted Authority (TA). Each Smart TV has an agent which continuously collects the viewing records of its owners.

The Trusted Authority coordinates the computation, verifies the validity of the records, collects the encrypted results and provides the compensation to the participants. This process is performed using encrypted viewing records, hence the record contents are never revealed to the Trusted Authority. Finally, we develop a prototype implementation and perform experiments that confirm the feasibility of the approach.

Some of the advantages of our approach in comparison to traditional TAM systems are:

- Preserving the privacy of participants' viewing records.
- More reliable measurements can be achieved, since a practically unrestricted number of participants can produce the PrivTAM results.
- Supports fine grained measurements which can be automatically calculated in small time intervals as well as specific one-time queries.

- Reducing the cost for conducting a TAM. No specialized equipment is required and only the participants in a calculation need to be compensated.
- Supporting measurements using records from any Internet-enabled broadcasting medium (e.g., Broadcast TV, Cable TV, IPTV and Satellite TV).

Our solution requires Smart TV's to have permanent Internet access, a requirement which is satisfied by default. Moreover, the computational and networking requirements of PrivTAM can be easily fulfilled by modern embedded Android-based platforms.

Related Work. To our knowledge this is the first attempt at creating a privacy-preserving TAM system, particularly one that supports an arbitrarily large amount of participants. In general, TAMs are products of aggregation operations and therefore our work is related to common privacy-preserving aggregation systems. For example, in [17], privacy-preserving data aggregation in people-centric urban sensing systems is discussed. A market for personal data, supporting anonymous data aggregation operations is presented in [3]. The economic aspects of personal privacy are discussed in [22,1]. The fact that individuals need to be in control and be compensated when their personal information is used for commercial purposes, is discussed in [11,18]. The sensitivity of the viewing records is stressed by both the Video Privacy Protection Act [20] and the Cable TV Privacy Act [19].

Overall, we consider that PrivTAM lies between privacy-preserving aggregation systems and e-voting systems, offering verifiable, privacy-preserving, aggregation functionalities. Additionally, PrivTAM takes into account the economic aspects of privacy and supports compensation functionalities for the measurement subjects.

2 The PrivTAM System

An overview of the PrivTAM system architecture, built on top of Smart TV technology, is shown in Figure 1. The main parts of the architecture are the participating Smart TVs, the Television Audience Measurement Service (TAM Service) and the Trusted Authority (TA). Every Smart TV contains a software agent that collects and stores its viewing records and maintains a set of demographic elements, such as gender, age and educational level of the viewers. The agent manages the viewers' personal data, provides controlled access to the data, and has the ability to participate in distributed protocols and computations.

The TAM Service collects the measurements and is responsible for coordinating the distributed key generation [13] for the public-key cryptosystem between itself and a group of L TV agents. These L agents are chosen with a verifiable random selection [7] and participate in both the public-key creation phase and the decryption of the results phase. The TA is responsible for coordinating the PrivTAM computation process. Time is divided into consecutive intervals, and for each interval, an aggregate result is periodically calculated using input from the participating Smart TV's. The TAM Broker is used to facilitate the

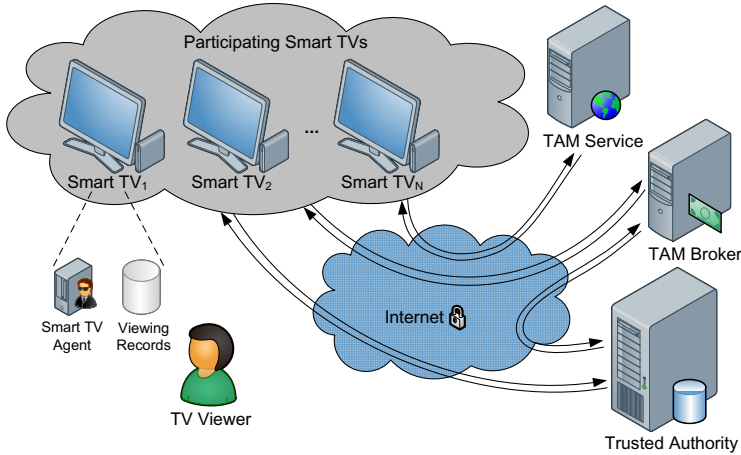


Fig. 1. The general architecture of our system

(optional) payment functionality described in Section 3.2. Each Smart TV agent encrypts its viewership vector with the public-key of the measurement and sends it to the TA for verification. Following a successful verification, the TA adds this vector to the current encrypted result of the measurement. The final encrypted TAM is transmitted to the participants in the distributed key generation for decryption and announcement of the result.

3 The PrivTAM Protocol

In this section, we present the cryptographic protocol used in PrivTAM. The communication between the entities in our protocol is performed over secure sockets (SSL/TLS) with both server and client authentication enabled. Our protocol is secure in the Malicious Model, assuming that the TA and the TAM Service are Honest-But-Curious (HBC) (see Section 4). During the calculation the actual users' personal data are not disclosed in any stage of the process, but only the final results are revealed at the end.

3.1 Problem Definition

We define the PrivTAM problem for verifiable, privacy-preserving TAM. A PrivTAM problem instance consists of:

- **N Smart TVs** - TV_1, TV_2, \dots, TV_N and the viewing records of their owners.
 - **Input:** The viewership vector of each owner.
 - **Output:** The TAM for the participating viewership vectors.

We assume that one viewership vector is submitted per Smart TV. We do not consider user identification issues within family members, as this is an existing issue in TAM systems and is out of the scope of this work.

3.2 Outline of the Computation

The computation consists of three main phases. In Figure 2 the participating entities of each phase are illustrated. The full descriptions of the three phases are given in the following paragraphs.

- In Phase 1 a distributed key generation for a Threshold Paillier Cryptosystem is performed.
- In Phase 2 the privacy-preserving TAM calculation takes place.
- In Phase 3 the final encrypted TAM is forwarded for decryption and the result is announced.

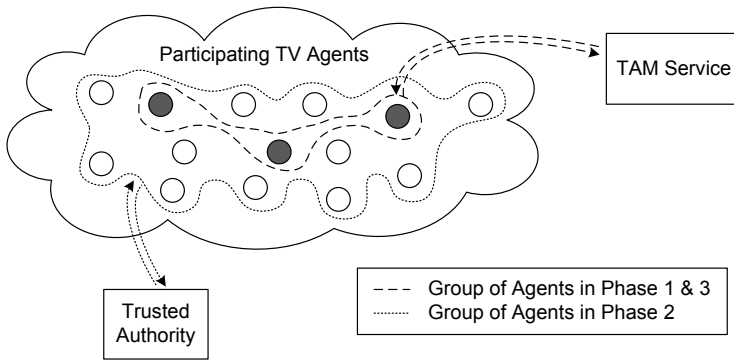


Fig. 2. Illustration of protocol participants

Phase 1. During Phase 1 the TAM Service selects an L -sized subset of the N -sized set of all the participating TV agents with a verifiably random procedure. An example of a publicly verifiable random selection process is described in [7]. This technique prevents the TAM Service from making a biased or impeachable group selection. Then, the TAM Service and the L selected TV agents execute a cryptographic protocol for the distributed key generation of the Threshold Paillier Cryptosystem [6]. We use the following Threshold Decryption Model, which is an adaptation of the corresponding definition in [4] to our needs, so that the distributed key generation can be performed without a trusted dealer [13].

Definition 1 (Threshold Decryption Model). *In a threshold cryptosystem, instead of merely decrypting the encrypted message, we use n parties P_i with their secret keys, so that at least t parties, where $t \leq n$, are required to decrypt the message. The decryption process includes the following players: a combiner (can be one of the n parties), a set of n parties P_i , and users. We consider the following scenario:*

- In an initialization phase, the parties use a distributed key generation algorithm to create the public key PK of their private keys SK_i . Next the parties publish their verification keys VK_i .

- To encrypt a message, any user can run the encryption algorithm using the public key PK .
- To decrypt a ciphertext c , we forward c to the combiner and n parties. Using their secret keys SK_i and their verification keys VK_i , each party runs the decryption algorithm and outputs a partial decryption c_i with a proof of validity of the partial decryption $proof_i$. Finally, the combiner uses the combining algorithm to recover the cleartext, provided that at least t partial decryptions are valid.

In PrivTAM, we use the Paillier public key generated in Phase 1 for the encryption of the viewership vectors and utilize the Pailler Cryptosystem’s homomorphic property in Phase 2. In addition, we specify that t is equal to n in our Threshold Decryption Model, meaning that all the parties are required to decrypt a message. Setting $t = n$ is important to ensure that the final result cannot be decrypted without the active participation of the TAM Service. Phase 1 should be repeated occasionally, to renew the keys and the set of L agents.

Phase 2. During this phase, the TA coordinates the voting process, and collects and verifies the encrypted viewership vectors of the participants. Upon successful verification, the TA adds the submitted viewership vector to the current TAM result, and sends the compensation to the participant.

In detail, Phase 2 begins with the TV agents that hold viewing records for the particular time period, creating their viewership vectors (Figure 3). Each such vector is submitted to the TA for the verification. The verification process is based on a zero-knowledge proof that an encrypted message lies in a given set of messages [4]. This way, when encrypting a message, it is possible to append a proof that the message lies in a public set $S = \{m_1, \dots, m_p\}$ of p messages without revealing any further information. This proof is described in detail in Section 4.

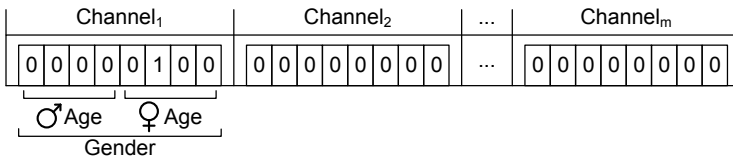


Fig. 3. Example of a viewership vector

In Figure 3, the viewership vector for m TV channels is illustrated. The vector section for each channel consists of a number of ciphertexts, which result from the number of demographic elements used in the vector. In our example, these elements are the age group and the gender of the viewer. The gender categories are male and female and the age groups are “ $Age_1 \leq 24$ ”, “ $25 \leq Age_2 \leq 40$ ”, “ $41 \leq Age_3 \leq 55$ ” and “ $Age_4 > 55$ ”. Consequently, a combination of 8 ciphertexts is created to represent these elements. In order to indicate the channel the viewer was watching, the representation of their demographic elements are added

to the viewership vector section for the corresponding channel. For example, in our case the gender is female, the age is between 25 and 40 years old and she was watching $Channel_1$. All the values in the viewership vector lie in the public set $S = \{0, 1\}$ and they are encrypted using the public key that is generated in Phase 1. Every participant should prove that their vector is valid so that the TA can avoid any malicious behavior from them. More specifically, the participants should prove that:

1. Every ciphertext in the viewership vector should lie in the set $S = \{0, 1\}$.
2. The multiplication of the ciphertexts in every channel should lie in the set $S = \{0, 1\}$.
3. Finally, the multiplication of all ciphertexts in the viewership vector should equal to “1”. This means that the participant was watching TV.

The multiplication of the ciphertexts in the above proofs utilizes the homomorphic property of the Paillier Cryptosystem [14]. Using homomorphic encryption one can perform a specific algebraic operation on the plaintext by performing a (possibly different) algebraic operation on the ciphertext. The additive homomorphic property of the Paillier cryptosystem, if the public key is modulus m , is shown in the following equation:

$$\mathcal{E}(x_1) \cdot \mathcal{E}(x_2) = \mathcal{E}(x_1 + x_2 \bmod m)$$

Once the viewership vector is confirmed by the TA, the vector is multiplied, using the homomorphic property, with the current TAM result. We assume that the TA only logs the participants in a measurement in order to ensure unreusability of the vectors. However, even if the vectors were stored, the TA would not be able to reveal their contents, unless all the participants of the threshold decryption are malicious and collude towards this purpose. The final result of Phase 2 is the encrypted TAM of the particular query, which ensures k-anonymity (see Section 4).

Payments in PrivTAM. The PrivTAM system can support functionalities for the compensation of participants, either in the form of financial payments or in the form of vouchers or points. The requirement for a participant to be compensated is that they provide a valid viewership vector to the computation. After the successful verification of a participant’s viewership vector, the TA sends the compensation to the participant.

In case of financial compensation, the payment scheme within PrivTAM needs to be efficient enough to facilitate large numbers of small amount payments, without entailing substantial transaction costs. Therefore, we draw techniques along the lines of micropayments, as proposed in [16]. The main actors in micropayment schemes are Brokers, Vendors and Users. A User becomes authorized to make micropayments by the Broker. A Vendor receives micropayments from authorized users and redeems them through the Broker. Relationships of Users and Vendors with the Broker are long term. In PrivTAM, the Smart TV owners can act as Vendors and the TA can act as a user making micropayments. The

TAM Broker is introduced in the architecture to facilitate the payments (Figure 1). A micropayment scheme suitable for PrivTAM is Payword, presented in [16]. Payword is a credit-based scheme, based on chains of hash values (called Paywords) and the Broker does not need to be online in order for a transaction between a User and a Vendor to take place. Due to lack of space, the Payword protocol is not presented here.

Alternatively, non-monetary compensation, including points that can be redeemed with participating companies, can be offered to participants. The amount of compensation for each PrivTAM calculation is fixed for simplicity, but methods for providing different pricing could be introduced into the system. It is important to stress that the collected points of each participant are not recorded in a profile by a centralized service, but are kept at the participant's side.

Phase 3. In Phase 3, the final encrypted result of Phase 2 is forwarded to the L selected TV agents of Phase 1 and the TAM Service. The L agents perform partial decryptions and send the results to the TAM Service which acts as the final participant and combiner of the threshold decryption. This way, only the TAM Service can see the final result of the calculation, which is acceptable if the TAM Service is considered honest and reports accurately the decrypted result. In order for the PrivTAM calculation to be protected from inaccurate reporting of the results from the TAM Service, a verification mechanism can be introduced to validate the announced results. This verification could be accomplished by using multiple combiners in the threshold decryption, to confirm the announced results from the TAM Service.

4 The Protocol's Security

In this section, we show that the PrivTAM protocol achieves the requirements described in the introduction, i.e., privacy, completeness, soundness, unreuseability, eligibility, and verifiability. The security model holds for Malicious viewers, with the assumption that the TA and the TAM Service are Honest-But-Curious (HBC). Malicious users can submit any value as input to the computation or even abandon the protocol at any step. See the definition of the Malicious Model given in [10] or the more detailed description in [8]. An Honest-But-Curious party (adversary) [2] follows the prescribed protocol properly, but may keep intermediate computation results, e.g. messages exchanged, and try to deduce additional information from them other than the protocol result.

The security of the Threshold Paillier cryptosystem and its homomorphic property ensures that the viewing records are never disclosed and cannot be associated with any particular participant. To prove the privacy attribute of the protocol, we show that it satisfies the criterion of k -anonymity [5]. In the context of this work, k -anonymity means that no less than k individual users can be associated with a particular personal viewing record.

The following zero-knowledge proof illustrates the steps of the verification process in Phase 2. The security of this zero-knowledge proof is shown in [4].

Proof that an encrypted message lies in a given set of messages [4]. Let N be a k -bit RSA modulus, $\mathcal{S} = \{m_1, \dots, m_p\}$ a public set of p messages, and $c = g^{m_i r^N} \pmod{N^2}$ an encryption of m_i where i is secret. In the protocol, the prover P convinces the verifier V that c encrypts a message in \mathcal{S} .

1. P picks at random ρ in \mathbb{Z}_N^* . He randomly picks $p - 1$ values $\{e_j\}_{j \neq i}$ in \mathbb{Z}_N and $p - 1$ values $\{v_j\}_{j \neq i}$ in \mathbb{Z}_N^* . Then, he computes $u_i = \rho^N \pmod{N^2}$ and $\{u_j = v_j^N (g^{m_j}/c)^{e_j} \pmod{N^2}\}_{j \neq i}$. Finally, he sends $\{u_j\}_{j \in \{1, \dots, p\}}$ to V .
2. V chooses a random challenge e in $[0, A[$ and sends it to P .
3. P computes $e_i = e - \sum_{j \neq i} e_j \pmod{N}$ and $v_i = \rho r^{e_i} g^{(e - \sum_{j \neq i} e_j) \div N} \pmod{N}$ and sends $\{v_j, e_j\}_{j \in \{1, \dots, p\}}$ to V .
4. V checks that $e = \sum_j e_j \pmod{N}$ and that $v_j^N = u_j (c/g^{m_j})^{e_j} \pmod{N^2}$ for each $j \in \{1, \dots, p\}$.

We note that r is the random number which was used for the encryption of message m_i and $a \div b$ is the quotient in the division of a by b . According to Theorem 2 of [4], it holds that t iterations of the above protocol is a perfect zero-knowledge proof (against an honest verifier) that the decryption of c is a member of \mathcal{S} , for any non-zero parameters A and t such that $1/A^t$ is negligible.

The main security features of the protocol are:

- The TA cannot obtain information about the contents of the viewership vector, since the ciphertexts are encrypted with the Paillier encryption.
- In case the TA stores the viewership vector, the contents cannot be revealed unless all the participants in the threshold decryption are malicious and collude towards this purpose.
- The participants cannot submit invalid viewership vectors and disrupt the calculation, due to the verification process.
- At the end of the protocol, only the aggregate TAM result is revealed. As a result, no individual can be associated with the viewership vector that they submitted. Consequently, the proposed protocol preserves k -anonymity for $k = N$, where N is the number of all the participants who take part in the measurement.
- In order to be protected from inaccurate result reporting from the TAM Service, multiple combiners can be introduced in Phase 3, to confirm the announced results.

5 Experimental Results

To evaluate our solution, we developed a prototype that implements the Priv-TAM calculation. The prototype can be separated into two main parts, the first being the application on the Smart TVs and the second the application on the TA. The Smart TV application is implemented using the Google TV

platform³ and the Java for Android 3.1 SDK. The application on the TA is also implemented in Java. Both applications use the cryptographic primitives of the Paillier Threshold Encryption Toolbox [21]. In this library, a centralized mechanism (with a trusted dealer) for threshold key generation [6] is implemented, instead of a distributed Paillier key generation [13]. In our view, this is enough for this prototype implementation.

The TV agents use production-ready cryptographic libraries and employ 1024 bits RSA X.509 certificates. The communication between agents is performed over secure sockets (SSL/TLS) with both client and server authentication. At this stage, the full functionalities of the TV agents described in our proposed system are not implemented, rather, we only implement the privacy-preserving cryptographic TAM computation.

We performed an experiment of the PrivTAM calculation, where 6 TV agents, the TA and the TAM Service participated and four channels exist. Each agent generated random values for the submitted viewing record, as well as for the gender and the age of the viewer. Initially, the TAM Service randomly chooses two of the participating TV agents ($L = 2$), $TVAgent_2$ and $TVAgent_5$, for the first phase of the protocol. Therefore, the final encrypted measurement will be decrypted from $TVAgent_2$, $TVAgent_5$ and the TAM Service ($n, t = 3$ parties).

Next, each TV agent encrypts the viewership vector and transmits it to the TA for verification. This process in our experiments takes less than 8 seconds. Once the viewership vector is verified, the TA multiplies it with the current encrypted TAM result. In Table 1 the values used to create the viewership vector of each agent are shown, along with the resulting current encrypted measurement after the submitted viewership vector is calculated by the TA.

Table 1. Example of a PrivTAM.

TV Agents Values				Current Encrypted TAM			
Agent	Channel	Gender	Age	$Channel_1$	$Channel_2$	$Channel_3$	$Channel_4$
$TVAgent_1$	$Channel_3$	Male	23	0000 0000	0000 0000	1000 0000	0000 0000
$TVAgent_6$	$Channel_1$	Female	45	0000 0010	0000 0000	1000 0000	0000 0000
$TVAgent_2$	$Channel_1$	Male	32	0100 0010	0000 0000	1000 0000	0000 0000
$TVAgent_4$	$Channel_4$	Female	29	0100 0010	0000 0000	1000 0000	0000 0100
$TVAgent_3$	$Channel_3$	Female	53	0100 0010	0000 0000	1000 0010	0000 0100
$TVAgent_5$	$Channel_3$	Female	22	0100 0010	0000 0000	1000 1010	0000 0100

At the end of the computation, the TA sends the encrypted results to $TVAgent_2$, $TVAgent_5$ and the TAM Service. The TAM Service collects the partial decryption results from $TVAgent_2$ and $TVAgent_5$, and combines the partial decryption results. The decrypted TAM result, is shown in the last row of Table 1, where $Channel_3$ has the highest audience (50%) and the 66.66% of viewers were women. A snapshot of the application during the execution of the experiment is shown in Figure 4.

³ <http://www.google.com/tv/>

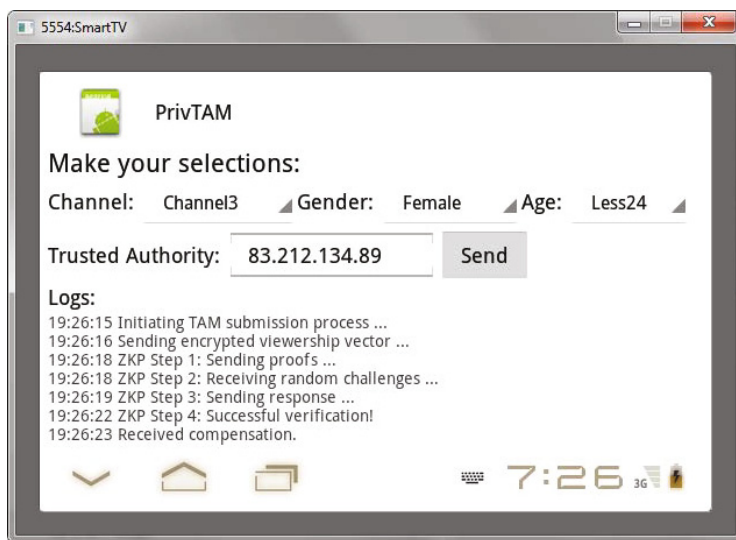


Fig. 4. A snapshot of the *TV Agents*

6 Conclusions

The introduction of Internet connectivity and computation capabilities to contemporary television systems, opens the possibility of conducting TAMs using larger samples of viewers. In this work we design an efficient protocol for privacy-preserving TAMs and test the applicability of the proposed solution. The accuracy and trustworthiness of the produced results act as strong incentives for TAM Services to adopt the PrivTAM system. From the viewers' perspective, PrivTAM offers the privacy assurance necessary for them to participate in a TAM system, while fair compensation can be offered for their participation, returning some of the economic benefits of TAMs back to the viewer. Additionally, PrivTAM can support alternative kinds of measurements, providing interesting information about audiences to the TV industry. These results are achieved without using any specialized equipment and can take into account data from multiple broadcasted sources.

Acknowledgments. This work was performed in the framework of and partially funded by the GSRT/CO-OPERATION/SPHINX Project (09SYN-72-419) (<http://sphinx.vtrip.net>).

References

1. Acquisti, A.: Privacy and security of personal information: Technological solutions and economic incentives. In: Camp, J., Lewis, R. (eds.) *The Economics of Information Security*, pp. 165–178. Kluwer (2004)

2. Acquisti, A., Gritzalis, S., Lambrinouidakis, C., De Capitani di Vimercati, S.: Digital privacy. Auerbach Publications. Taylor & Francis Group (2008)
3. Adar, E., Huberman, B.: A market for secrets. *First Monday* 6(8) (2001)
4. Baudron, O., Fouque, P.A., Pointcheval, D., Stern, J., Poupard, G.: Practical multi-candidate election system. In: *PODC 2001*, pp. 274–283. ACM, New York (2001)
5. Ciriani, V., Capitani di Vimercati, S., Foresti, S., Samarati, P.: κ -anonymity. In: *Secure Data Management in Decentralized Systems. Advances in Information Security*, vol. 33, pp. 323–353. Springer, Heidelberg (2007)
6. Damgård, I., Jurik, M.: A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System. In: Kim, K.-C. (ed.) *PKC 2001*. LNCS, vol. 1992, pp. 119–136. Springer, Heidelberg (2001)
7. Eastlake, D.: Publicly Verifiable Nominations Committee (NomCom) Random Selection. RFC 3797 (Informational) (June 2004)
8. Goldreich, O.: *The Foundations of Cryptography*, vol. 2. Cambridge University Press (2004)
9. Karlof, C., Sastry, N., Wagner, D.: Cryptographic voting protocols: a systems perspective. In: *14th USENIX Security Symposium*, pp. 33–50 (2005)
10. Kissner, L., Song, D.: Privacy-Preserving Set Operations. In: Shoup, V. (ed.) *CRYPTO 2005*. LNCS, vol. 3621, pp. 241–257. Springer, Heidelberg (2005)
11. Kleinberg, J., Papadimitriou, C.H., Raghavan, P.: On the value of private information. In: *TARK 2001*, pp. 249–257. Morgan Kaufmann Publishers Inc. (2001)
12. Kremer, S., Ryan, M., Smyth, B.: Election Verifiability in Electronic Voting Protocols. In: Gritzalis, D., Preneel, B., Theoharidou, M. (eds.) *ESORICS 2010*. LNCS, vol. 6345, pp. 389–404. Springer, Heidelberg (2010)
13. Nishide, T., Sakurai, K.: Distributed Paillier Cryptosystem without Trusted Dealer. In: Chung, Y., Yung, M. (eds.) *WISA 2010*. LNCS, vol. 6513, pp. 44–60. Springer, Heidelberg (2011)
14. Paillier, P.: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In: Stern, J. (ed.) *EUROCRYPT 1999*. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)
15. Pieprzyk, J., Hardjono, T., Seberry, J.: *Fundamentals of computer security. Monographs in theoretical computer science*, ch. 15. Springer (2003)
16. Rivest, R.L., Shamir, A.: PayWord and MicroMint: two simple micropayment schemes. In: *CryptoBytes*, vol. 2, pp. 69–87 (1996)
17. Shi, J., Zhang, R., Liu, Y., Zhang, Y.: PriSense: Privacy-preserving data aggregation in people-centric urban sensing systems. In: *IEEE INFOCOM 2010*, pp. 1–9. IEEE (March 2010)
18. Tasidou, A., Efraimidis, P.S., Katos, V.: Economics of personal data management: Fair personal information trades. In: *Next Generation Society. Technological and Legal Issues*. LNICST, vol. 26, pp. 151–160. Springer, Heidelberg (2010)
19. US Government: The cable tv privacy act of 1984. In: 47 USC, Chapter 5, Subchapter V-A, Part IV, Sec. 551. U.S. Gov. Printing Office, Washington, DC (1984)
20. US Government: Video privacy protection act. In: 18 USC, Part I, Chapter 121, Sec. 2710, Pub.L. 100-618. U.S. Gov. Printing Office, Washington, DC (1988)
21. UTD Data Security and Privacy Lab: Paillier threshold encryption toolbox (November 2011), <http://www.utdallas.edu/~mxk093120/cgi-bin/paillier/>
22. Varian, H.: Economic aspects of personal privacy. In: *Privacy and Self-regulation in the Information Age*. NTIA, Washington, DC (1997)
23. Yao, A.C.C.: Protocols for secure computations (extended abstract). In: *FOCS 1982*, pp. 160–164. IEEE, Los Alamitos (1982)

Tracking Users on the Internet with Behavioral Patterns: Evaluation of Its Practical Feasibility

Christian Banse¹, Dominik Herrmann², and Hannes Federrath²

¹ Fraunhofer AISEC, Garching b. München, Germany
`christian.banse@aisec.fraunhofer.de`

² University of Hamburg, Department of Informatics, Germany
`{herrmann,federrath}@informatik.uni-hamburg.de`

Abstract. Traditionally, service providers, who want to track the activities of Internet users, rely on explicit tracking techniques like HTTP cookies. From a privacy perspective *behavior-based tracking* is even more dangerous, because it allows service providers to track users passively, i. e., without cookies. In this case multiple sessions of a user are linked by exploiting characteristic patterns mined from network traffic.

In this paper we study the *feasibility* of behavior-based tracking in a real-world setting, which is unknown so far. In principle, behavior-based tracking can be carried out by any attacker that can observe the activities of users on the Internet. We design and implement a behavior-based tracking technique that consists of a Naive Bayes classifier supported by a cosine similarity decision engine. We evaluate our technique using a large-scale dataset that contains all queries received by a DNS resolver that is used by more than 2100 concurrent users on average per day. Our technique is able to correctly link 88.2% of the surfing sessions on a day-to-day basis. We also discuss various countermeasures that reduce the effectiveness of our technique.

1 Introduction

Tracking users on the Internet is perceived as a serious infringement of their privacy, especially when it is done *without their consent*. Nevertheless, tracking cookies, which allow content providers to link multiple sessions of a user, are used on many popular websites today [2]. This allows ad networks such as Google Analytics, Doubleclick or Facebook to create usage profiles from the websites a user interacts with over time. Service providers that require authentication can also track the actions of their users. Apart from the Internet Service Provider this applies to, among others, commercial anonymization services that rely on single-hop proxies or VPNs (e. g., anonymizer.com and ipredator.se) and the providers of browser toolbars (e. g., *Alexa* and *Web of Trust*). We call this practice *explicit tracking*, because it uses well-known techniques, relies on the cooperation of the web browser or the user and can be detected rather easily.

Instead of explicit tracking we are interested in the feasibility of **behavior-based tracking** techniques, which rely on characteristic patterns within the

activities of the users. In contrast to explicit tracking it can be used to link multiple Internet sessions of a user *passively*, i. e., without the user’s cooperation and even in the absence of unique identifiers or tracking cookies. Behavior-based tracking constitutes a considerable privacy threat, because it can not only be carried out *without consent*, but it is also *imperceptible*, i. e., it cannot be detected. As the recent discussions regarding the Do-not-Track initiative revolved mainly around tracking cookies, behavior-based tracking may come as a surprise even for well-informed users. For example, users of an alternative DNS resolver (like OpenDNS or Google DNS)¹, a service provider that cannot use explicit tracking for technical reasons, do probably not expect that their resolver can track their activities. As our results show, DNS resolvers are in a good position for behavior-based tracking, though.

Behavior-based tracking of users is “challenging” because many Internet Service Providers assign their customers dynamic, periodically changing IP addresses [25][24]. Recent research on user profiling techniques may help curious service providers to overcome this hurdle: given a database with traffic profiles of a set of users, data mining techniques can be used to map traffic with unknown origin to one of the users from the database [15][26][13]. The mapping becomes more difficult for larger numbers of users and for smaller amounts of training data [26]. The actual privacy threat of behavioral profiles is unknown so far, because previously published results were obtained with a limited number of concurrent users (up to 100 in [26]) in a closed-world evaluation setting. Therefore, we address the following research question in this paper: *To what degree is behavior-based tracking feasible in practice and how can users protect themselves against it?* This question can be tackled by applying techniques that are known to work well for behavioral profiling to traffic of a large number of Internet users. Existing profiling techniques are not suitable for our tracking problem, though (cf. Sect. 2). Therefore, based on previous work we start out by devising a tracking technique and establish its suitability for behavioral profiling. We then analyze the real-world feasibility within an evaluation setup that incorporates the challenges to be faced in practice.

Our Contribution. We design a tracking technique that exploits behavioral characteristics, implement a scalable evaluation environment with a *MapReduce* framework and evaluate our technique in a realistic setting with a large-scale dataset. Given more than 2100 concurrent users on average per day, we find that behavior-based tracking is feasible with very high accuracy for a third of the users and with moderate accuracy for the remaining ones. On overall 88.2% of all session are linked correctly. The high accuracy is achieved by resolving ambiguous results caused by fluctuating users. To the best of our knowledge we are the first to assess large-scale behavior-based tracking in a real-world setting.

The paper is structured as follows. We present related work in Sect. 2 before modeling the tracking problem in Sect. 3 and describing our dataset in Sect. 4. Our tracking technique is illustrated in Sect. 5 and evaluated in Sect. 6. Finally, we discuss countermeasures in Sect. 7 before we conclude in Sect. 8.

¹ cf. <http://www.opendns.com> and <http://code.google.com/speed/public-dns>

2 Related Work

In the following we outline related research efforts that analyze the possibility to re-identify users based on their traffic or behavioral profiles. They differ from our work in terms of application, techniques and evaluation methodology as well as the size of their dataset.

Yang [19,27,26] analyzes visitation patterns of web users as an additional authentication factor to tackle identity theft and fraud in e-commerce applications. She assumes that an e-commerce provider (e.g., a web shop) has access to user profiles built from multiple surfing sessions of its users. Each time a user logs into the web shop the provider is supposed to retrieve the most recent surfing sessions from the user in order to confirm the user's identity. To this end Yang constructs user profiles from the hostnames of the web sites visited within a surfing session. She proposes two profiling techniques based on *support* and *lift*, which are common measures in the field of *association rule mining*. The effectiveness of the techniques is evaluated and compared with the J4.8 decision tree classifier from Weka [12] and with Support Vector Machines [8]. In a scenario with 100 concurrent users the predictive accuracy of her techniques reaches 87% for profiles built from 100 training sessions. The accuracy drops to 62%, if only one training session is available, which corresponds to our behavior-based tracking scenario. Yang's results demonstrate that characteristic behavior can be exploited for user re-identification in her e-commerce scenario with a limited number of users. Its feasibility for large-scale re-identification problems or behavior-based tracking with many users cannot be deduced from her results, though.

Kumpost's user re-identification technique relies on the destination IP addresses for HTTP(S) and SSH connections of each user [15]. His user profiles consist of sparse access frequency vectors that contain the number of connections to each destination IP address. This is similar to our hostname-based profiles. Kumpost's re-identification methodology employs the *inverse document frequency* transformation and the *cosine similarity metric*. Kumpost evaluates the effectiveness of the technique using *monthly aggregated* NetFlow logs. Unfortunately, he fails to provide statistics regarding the size of the dataset, i.e., the number of concurrent users. Although monthly profiles should contain a considerable amount of information, Kumpost reports rather high false-positive rates of 68% for HTTP traffic and 21% for SSH traffic.

In a previous publication we proposed a user re-identification technique based on the Multinomial Naïve Bayes classifier from Weka [13]. The classifier was evaluated using the HTTP traffic of 28 volunteering users. Given one training session per user, up to 73% of sessions were classified correctly. In comparison to [13] the tracking technique presented in Sect. 5 utilizes an improved feature extraction technique with *n-grams* and incorporates an additional *cosine similarity decision engine*. Moreover, the evaluation in Sect. 6 is carried out on a much larger dataset.

3 Modeling Behavior-Based Tracking

Building on the promising results of previous studies described in Sect. 2, our behavior-based tracking scheme solely relies on the web sites visited within a surfing session. Order of requests and exact timings are neglected in this model.

We assume that each user is represented by a dynamic IP address that is replaced by a different one after a fixed amount of time, i. e., there are epochs $e_i \in E$ of constant length (e. g., 24 hours, beginning at 4:00 am each day). Moreover we assume that the service provider that wants to use behavior-based tracking can record the interactions of its users with a set of destination hosts H . Each interaction is stored as a triplet (e, s, h) consisting of the epoch e it was observed in, as well as the corresponding source address s and the destination host h (in our case: a hostname). A *user session* is then obtained by aggregating all events that share the same e and s to obtain the access frequency of each destination h . Each user session can be mapped to an $|H|$ -dimensional vector, where the i -th component corresponds to the access frequency of the i -th destination according to a totally ordered list of all observed destinations H . Figure 1 summarizes the transition from the real-world view of the service provider to our model.

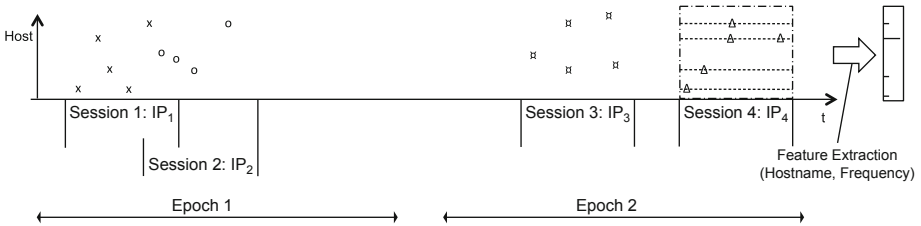


Fig. 1. Behavior-based tracking scenario

4 Dataset

We evaluate the feasibility of behavior-based tracking using *DNS queries*, i. e., from the viewpoint of a curious DNS resolver. Our dataset was obtained in cooperation with the computer center of the University of Regensburg, Germany. It contains all DNS requests, which were received by the two DNS resolvers of the university network between February 1, 2010 and June 30, 2010. In order to maintain a high level of privacy, every source IP address s was replaced by a pseudonym $u = h(s|r_{\text{const}})$ using a hash function h before the record was stored. The actual salt value r was not disclosed to us and thrown away once data collection was completed. Nevertheless, this practice cannot ensure a satisfactory level of privacy for all users: some users may issue personally

identifying queries (e. g., for their weblog at *username.blogspot.com*), which undermine our pseudonymization efforts. Therefore, we limit access to the dataset to a small group of researchers.

In total the dataset contains 2,645,748,393 queries for 77,662,943 unique hostnames issued by 18,904 unique source IP addresses. To maintain a high data quality we only consider queries originating from the student housing network segment in this paper. Each network device must be personally registered by its resident and is assigned a *unique, static* IP address². The majority of the 4153 considered users in our dataset are students of the University of Regensburg, which use their Internet access for research and studying as well as private browsing. On average 2123 users were active per day.

During the evaluation of the tracking technique we simulate *dynamically changing IP addresses*, i. e., *a priori* our classifier (cf. Sect. 5) does not know which sessions belong to a given user. Of course, due to the static IP addresses used within the campus network, we *do* know all the sessions that belong to a given user. Thus, we can compare the predictions of our tracking technique with the actual set of sessions of each user to measure its accuracy.

5 Behavior-Based Tracking Technique

5.1 Multinomial Naïve Bayes Classifier

The behavior-based tracking problem can be formulated as a classification problem [18]. Every session corresponds to a single *instance* $x \in X$ which is represented by its attribute vector (cf. Sect. 3). Each instance belongs to a class $c \in C$, which corresponds to a certain user u (cf. Sect. 4). A set of *training instances*, whose class assignments are known (Sessions 1 and 2 in Fig. 1), is used to build a predictive model. Based on the model the *classifier* assigns the most probable class to each *test instance*, whose class is supposed to be unknown in the experiment (Sessions 3 and 4 in the example).

For the *Multinomial Naïve Bayes classifier* (**MNB classifier**) [18] used in this paper the probability that a test instance x belongs to a certain class c_i is defined as

$$P(c_i|x) \sim \prod_{h \in H} P(h|c_i)^{f_{h,x}}.$$

The classifier assigns the test instance x to the class with the highest probability $P(c_i|x)$. In our case $P(h|c_i)$ represents the probability that a hostname h is to be found in the training instances of class c_i . $f_{h,x}$ specifies the frequency with which h occurs in the given test instance x . The rationale behind this formula is: the more often frequently accessed hosts seen during training of a class do appear in the given test instance, the more likely does the test instance belong to that class.

² Each address is supposed to be exclusively used by one resident; in some rare cases devices may be used by multiple users, though.

The selection of the MNB classifier for behavior-based tracking may seem unintuitive at first sight. After all, it is usually only applied to text categorization problems (e. g., in spam filters) [18]. One aspect that motivates the application of the MNB classifier for the problem at hand is the fact that the features used in our model (access frequencies of hostnames) and the features used for text categorization (term frequencies) have an important similarity: both of them are subject to power law distributions [1,29]. Another argument in favor of the MNB classifier is its little computation complexity in comparison to more sophisticated classifiers such as *Support Vector Machines* (SVMs) [8]. In Yang’s experiments the application of SVMs increased classification runtimes by 300 % [26].

5.2 Transformation of Frequency Vectors

The text categorization discipline makes use of numerous transformation techniques in order to boost classification accuracy. We implemented the most promising techniques as described below.

The first technique consists of a transformation of the individual access frequencies $f_{h,x}$ of an instance vector. In our case $f_{h,x}$ specifies how often a hostname h occurs in a session x . The frequencies are scaled down by a sub-linear transformation $\log(1 + f_{h,x})$ in order to minimize the influence of extremely large frequency values [23, p. 311]. Additionally, the frequency vectors of all classes are normalized to a uniform Euclidean length [23, p. 310]. The application of the two transformations is indicated by the use of **TFN** in this paper.

Additionally, the frequencies $f_{h,x}$ can be multiplied by the *Inverse Document Frequency* $idf_h = \log \frac{N}{n_h}$ [23, p. 311], where N is the number of all instances and n_h the number of all instances containing the hostname h . The application of the *IDF* reduces the weight of common hostnames that are accessed by a large number of different users. The combination of the *IDF* and *TFN* transformation is designated by **TFIDFN**.

Finally, *n-grams* are known to increase the accuracy for many text mining problems [3,6,9]. They have also been used for traffic analysis problems with success [21]. This technique combines the hostnames of n adjacent events to obtain a composite hostname. We denote the usage of *n-grams* by appending a suffix to the configuration name, e. g., **TFN-1+2** indicates that – in addition to the original hostnames – bi-grams are added to the instances.

5.3 Resolving Ambiguous Predictions with Cosine Similarity

According to our model only a single user can issue queries from a certain IP address within an epoch (cf. Sect. 3). Therefore, not more than one instance should be mapped to each class within an epoch. If the classifier *does* assign more than one test instance to a given class, we resort to an additional decision criterion to resolve the ambiguous prediction, i. e., to single out the instance that fits best (cf. Sect. 6.3). For this purpose we employ the *cosine similarity metric*.

The cosine similarity is a [0;1]-normalized similarity measure between two vectors \mathbf{x} and \mathbf{y} that reflects the angle θ which is spanned by them. A smaller θ

corresponds to a larger value of the cosine similarity. The similarity s is equal to the quotient of the dot product of \mathbf{x} and \mathbf{y} and the product of their magnitudes:

$$s_{\mathbf{x},\mathbf{y}} = \cos \theta_{\mathbf{x},\mathbf{y}} = \frac{\mathbf{x} \cdot \mathbf{y}}{|\mathbf{x}||\mathbf{y}|}$$

Ambiguous predictions are resolved by computing the similarity values between the training instance of the class and all the test instances that have been assigned to it. The test instance that achieves the highest similarity value is assigned to the class, all other instances are discarded.

6 Evaluation

Due to the large size of the dataset we did not use off-the-shelf data mining tools such as *Weka* [12] for the evaluation. Instead, we ported the MNB classifier based on the *Weka* source code to Apache Hadoop [22], a *MapReduce* [10] framework implemented in Java. We obtained the results presented in this paper on a cluster of 12 quad-core machines (Intel Core i5, 3.1 GHz, 8 GB RAM). The running time of individual experiments was below 24 hours most of the time, which enabled us to try out many parameters. In the following we will report our evaluation methodology and the most significant results.

6.1 Cross Validation

Firstly, we analyze the effectiveness of the classifier and the transformations presented in Sect. 5.2 with a series of *cross validation* experiments. Cross validation ensures that the results obtained are not biased due to selection of peculiar training instances [14]. In order to obtain meaningful results, a *stratified* dataset is used, which contains the same number of instances for all classes.

For the cross validation experiments we focused on the very active period between March 1, 2010 and June 30, 2010, to include as many users as possible. We randomly selected 3000 users $u \in U$, which were able to contribute at least 20 instances of 24 hours length in the mentioned period. 20 instances were randomly selected from each user resulting in a cross validation dataset D of 60,000 instances in total.

In the following we illustrate the evaluation procedure for the case of *10-Fold Stratified Cross Validation*: first of all, the 20 instances of each class are split randomly, yet equally to obtain 10 disjoint sets (folds): $D = D_1 \cup D_2 \cup \dots \cup D_{10}$. The evaluation takes place in 10 runs $k \in 1, \dots, 10$. In the k -th experiment the instances $D_{\text{train}} = D \setminus D_k$ are used to train the classifier. D_{train} contains $\frac{9}{10}$ of all instances of every user u . With I_{train}^u denoting the training instances of one user, we have $D_{\text{train}} = I_{\text{train}}^{u_1} \cup \dots \cup I_{\text{train}}^{u_{3000}}$. Based on the supplied training data the classifier has to predict the appropriate classes for the instances in D_k . The overall accuracy is finally obtained as average of all 10 runs.

For each experiment we compute the average *precision* and *recall* values, two metrics, which are used to analyze the prediction accuracy of data mining

techniques. Both metrics are bound to the interval $[0;1]$. Assuming the classifier assigned n instances to some class c_i , where $m \leq n$ of the instances actually do belong to c_i , then the precision of this result is given by the ratio $\frac{m}{n}$, i. e., it expresses the pureness of the result. On the other hand, the recall expresses whether the classifier “found” all test instances of c_i : given l test instances of c_i the recall value is equal to $\frac{m}{l}$. Thus, a service provider that tries to track a user as extensively as possible over a period of time is primarily interested in high recall values. If the objective is to minimize the number of false mappings, a high precision value is of high importance.

For ease of exposition we only report the average recall values in the following. In all experiments the average precision values were constantly above the recall values by about 3 percentage points. Figure 2a shows the results for different configurations of the classifier with $|I_{\text{train}}| = 18$ training and 2 test instances per user. For 1-grams we obtain a recall of 83.1 % using the *TFN* transformation. The remaining transformations increase the recall further. The configuration *TFIDFN-1+2* achieves the best result with an average recall of 89.0 %. Higher-order n-grams have no more positive effects.

According to the results the selected text mining techniques are suitable for behavioral user profiling. The high accuracy obtained in the previous experiment is of little relevance for behavior-based tracking in practice, though, because we do not expect the service provider to have $|I_{\text{train}}| = 18$ training instances for each user at his disposal. Hence, we repeated the cross validation experiments with a smaller amount of training data. As expected, the recall values of these experiments show that with a decreasing number of training instances it is increasingly difficult for the classifier to assign the test instances to the correct class. In case of a *single training instance* the recall drops to 69.2 %. Even for a large user group the behavior of most Internet users seems to exhibit sufficiently characteristic attributes, which are present in a large share of their sessions.

Exploited Characteristics. We manually reviewed a sample of the trained model to understand the patterns the classifier relies on. As expected the classifier mainly exploits patterns in the web surfing behavior of the users, which are reflected by the DNS queries pretty well; only repeated queries for the same hostnames cannot be observed due to caching. For some users the classifier also relies on patterns introduced by the installed applications on their machine (visible due to update checks) or environmental peculiarities (e. g., the Windows Network Browser looking up its neighbors).

6.2 Evaluation in Real-World Setting

After having demonstrated the fundamental suitability of our technique in Sect. 6.1, we proceed with experiments that apply it to the actually recorded traffic. This will allow us to study its feasibility in a real-world setting. To this end, we have implemented a fully-automated experimental environment with the Hadoop framework that allows us to simulate a service provider that tries to track all users from one day to the following day, i. e., the *epochs* (cf. Sect. 6.2)

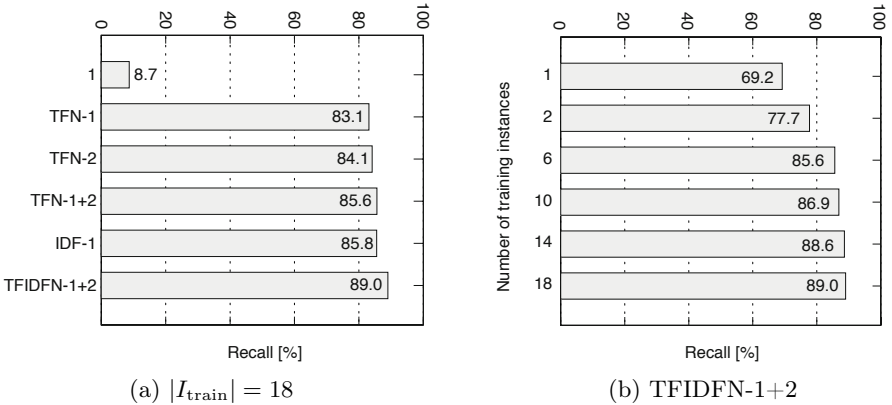


Fig. 2. Cross validation: observed recall values for 3000 users

start at midnight and last 24 hours³. The simulation iterates over all days t in chronological order. On a given day t a class c_i is set up for each active user u_i and the MNB classifier is trained with all instances x present on that day. The learned model is used to predict the most probable classes for all instances from the following day $t + 1$, i. e., to link the sessions of the two consecutive days.

In the case that user u_i is active on two consecutive days, the simulated observer scores a *correct mapping* (C) within an iteration, if the classifier predicts that the instance of user u_i on day $t + 1$ belongs to the class c_i , which was set up using the instance of u_i from the previous day, *and* if no other instance from $t + 1$ is assigned to c_i . If the classifier assigns *exactly one* instance x_j , which belongs to a different user $u_j, j \neq i$, to c_i , we record a *non-detectable error* (E_1). We record a *detectable error* (E_2) for ambiguous results, i. e., if instances from multiple users (possibly including u_i) are assigned to c_i . In the case that user u_i is not active on day $t + 1$ any more, a *correct mapping* is scored, if no instances are assigned to c_i at all. The overall *accuracy* of the classifier is given by counting all mappings of an experiment and computing $|C|/(|C|+|E_1|+|E_2|)$.

We evaluate the MNB classifier with this scheme for the month of May, which sports the largest number of active users. As we want to report results that are significant for typical users we ranked the users according to their daily query volume and removed the instances of the top 5% and bottom 5% of the list. Accuracy decreases by about 2.5 percentage points if this step is skipped.

Using the *TFIDFN-1+2* configuration the classifier achieves an average accuracy of 76.6%. Thus, tracking users in the real-world setting works better than the recall value of 69.2% obtained for the cross validation experiment with a single training instance suggests. This counterintuitive result can be explained by the average number of active users per day (average: 2412, standard deviation: 695) being smaller than the 3000 concurrent users used in Sect. 6.1.

³ We found that the actual time of day at which the epochs start has a negligible effect on the accuracy.

Robustness of profiles. We also studied the effect of the age of the trained profiles. To this end we randomly drew tuples with two dates $(t_i, t_{i+\delta})$, which were exactly $\delta \in \{5, 15, 30, 60, 90\}$ days apart. The instances on t_i were used for training and the classifier had to predict the classes of the instances on $t_{i+\delta}$. Accuracy values degraded rather slowly: from 78.3% for $\delta = 5$ to 44.7% for $\delta = 90$. The results suggest that the behavior of many users is rather stable, i. e., tracking is also possible with outdated profiles.

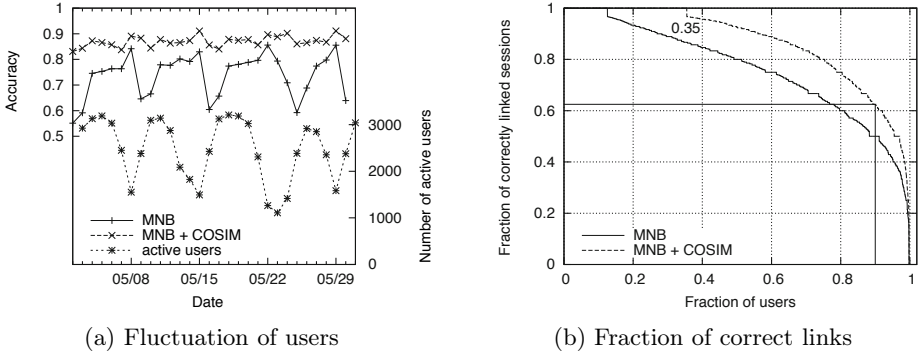


Fig. 3. Tracking in the real-world setting

6.3 Dealing with Fluctuating Activity

The real-world setting is more challenging than the cross validation experiment because the classifier is faced with user fluctuation. The classifier will encounter instances, for which no class has been trained on the previous day, because the respective user was inactive on that day. Nevertheless, the default implementation of the MNB classifier will assign such an instance to the most likely class, which causes a (non-)detectable error. In our dataset the fluctuation is caused by students that leave the city on weekends. Once they become active again accuracy drops to about 60%, while it reaches more than 80% during the week (cf. Fig. 3a, *MNB* graph). Further analysis of the results of the initial experiment from Sect. 6.2 reveals the extent of this problem: ambiguous mappings account for 13.6% of all cases (44.6% of all errors), and in 90.4% of the ambiguous mappings the correct test instance was in fact part of the set of assigned test instances. Therefore, resolving ambiguous results with the cosine similarity is promising (cf. Sect. 5.3). The effect of this optimization is shown in the graph labeled *MNB+COSIM* in Fig. 3a. On overall the accuracy increases to 88.2%, i. e., about 85% of the ambiguous results are resolved correctly.

Not all users exhibit enough characteristic patterns for the behavior-based tracking to be effective. Figure 3b shows the accuracy levels that can be obtained for a certain fraction of users. The *MNB+COSIM* configuration is able

to establish a correct mapping for all sessions of 35% of the users. For 90% of the users at least 60% of their sessions were linked correctly.

6.4 Future Work

According to our results tracking students based on their DNS queries works quite well. A globally distributed service provider, such as Google's DNS resolver, is faced with additional challenges, though. The queries of multiple users may be issued from a single source IP in case they are hidden behind a NAT gateway, a typical scenario for domestic broadband connections. Moreover, users may be located in different time zones, i. e., queries must be assigned to epochs depending on the location of the respective user. In future work we will study the impact of these issues on the feasibility of our technique.

Another important area of work is the evaluation of our technique in different attack scenarios. Especially concerning is the question whether third-party tracking networks such as Doubleclick, Google Analytics or Facebook can exploit behavioral analysis to overcome privacy-enhancing techniques. Today, visits of different websites within a session can be mapped to a specific user due to a tracking cookie. If users delete their cookies between sessions or if they enable the Private Browsing mode, recurring visits in different sessions cannot be linked any more. With our behavior-based technique third-party trackers may be able to overcome this limitation, though.

7 Countermeasures

In this section we present initial findings from our search for countermeasures that potentially mitigate the effectiveness of behavior-based tracking. The results reported below are obtained by repeating the experiment described in Sect. 6.3.

A generic protective measure is to use **anonymizers** like Tor or AN.ON (JonDonym) [114] that hide communication patterns from eavesdroppers. We will not discuss anonymizers any further in this paper, as we are interested in countermeasures that directly target the behavior-based tracking technique.

If the data requested by the users is valid for a certain amount of time, which is the case for many DNS records, a **caching system** may be a viable countermeasure to hide repeated queries from the observer. Thus, the user profiles would consist of queries for data that hasn't been requested before and queries for expired entries only. Motivated by the fact that many DNS records have a time-to-live (TTL) value of 24 hours, we can simulate a cache, whose entries expire at the end of each day. As a result instance vectors do not contain the actual number of queries per hostname any more. The effect of such a cache is very limited, though: accuracy only drops from 88.2% to 80.5%. While keeping records in the cache for a longer time may further decrease accuracy, this practice may introduce usability and security issues due to stale information.

Another countermeasure that reduces the amount of information available to the classifier may be to **change IP addresses frequently**. We study the effect of shorter sessions by pretending that our users change their IP addresses multiple times per day. This strategy proves to be quite effective: for sessions with a length of three hours we observe an accuracy of 60.4%, which decreases further to 49.5% for a length of one hour. Time-based address changes have the drawback that all existing connections are terminated, though. The large address space of IPv6 may allow the design of more sophisticated countermeasures in the future [20]. The extreme case would be to issue each query from a different IP address (each having a different IPv6 address prefix) to minimize linkability.

Finally, we consider the effectiveness of **Range Queries**, a technique that has been proposed to protect the privacy of DNS queries [5,28,17]. It is based on the principle of Private Information Retrieval [7,16] and aims to achieve privacy by hiding each query of a user within a set of n random dummy queries. We implemented a Range Query engine, which adds dummy queries to the instances, and found that the effectiveness of Range Queries largely depends on the values of various parameters, among them the number of dummies n , the size of the pool of hostnames to choose the dummies from (N), and the actual hostnames used as dummies. As a first result we can report that accuracy can be reduced to as low as 10%, if $n = 5$ dummies are used per query and dummies are drawn from a set of $N = 5000$ randomly selected hostnames. In future work we will search for configurations that offer a good trade-off between security and bandwidth.

8 Conclusion

The behavior-based tracking scheme studied in this paper enables service providers, which have access to the (web) requests of users, to link multiple sessions without cookies or other explicit identifiers. Our analysis shows that behavior-based tracking is feasible in a real-world setting for a large user group: in our DNS query log, in which more than 2000 users are active each day, we correctly link up to 88.2% of all sessions on a day-to-day basis. Our design manages to achieve this high accuracy, because it is able to resolve ambiguous classification that are caused by fluctuating numbers of users.

Daily changing IP addresses, which are sometimes cited to be important for users' privacy, offer only limited protection against behavior-based tracking. While there are no practical countermeasures so far, we see the introduction of IPv6 as an opportunity to design effective and lightweight techniques that help to prevent profiling and tracking users without their consent.

Acknowledgments. We thank Martin Wimmer, head of the computing center of the University of Regensburg and his employees, who enabled the compilation of our dataset. We are grateful to the anonymous reviewers, to Arvind Narayanan, to Christopher Pioceny and to our colleagues Karl-Peter Fuchs and Christoph Gerber for their critical feedback and constructive comments.

References

1. Adamic, L., Huberman, B.: Zipf's Law and the Internet. *Glottometrics* 3(1), 143–150 (2002)
2. Ayenson, M., Wambach, D.J., Soltani, A., Good, N., Hoofnagle, C.J.: Flash Cookies and Privacy II: Now with HTML5 and ETag Respawning (2011), <http://ssrn.com/abstract=1898390>
3. Beesley, K.R.: Language identifier: A computer program for automatic natural-language identification of on-line text. In: *Language at Crossroads: Proceedings of the 29th Annual Conference of the American Translators Association*, pp. 12–16 (1988)
4. Berthold, O., Federrath, H., Köpsell, S.: Web MIXes: A System for Anonymous and Unobservable Internet Access. In: Federrath, H. (ed.) *Anonymity 2000. LNCS*, vol. 2009, pp. 115–129. Springer, Heidelberg (2001)
5. Castillo-Perez, S., García-Alfaro, J.: Evaluation of Two Privacy-Preserving Protocols for the DNS. In: *Proceedings of the Sixth International Conference on Information Technology: New Generations*, Washington, DC, USA, pp. 411–416 (2009)
6. Cavnar, W.B., Trenkle, J.M.: N-Gram-Based Text Categorization. In: *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pp. 161–175 (1994)
7. Chor, B., Kushilevitz, E., Goldreich, O., Sudan, M.: Private Information Retrieval. *J. ACM* 45(6), 965–981 (1998)
8. Cortes, C., Vapnik, V.: Support-Vector Networks. *Machine Learning* 20(3), 273–297 (1995)
9. Damashek, M.: Gauging Similarity with n-Grams: Language-Independent Categorization of Text. *Science* 267(5199), 843–848 (1995)
10. Dean, J., Ghemawat, S.: MapReduce: Simplified data processing on large clusters. *Communications of the ACM* 51(1), 107–113 (2008)
11. Dingledine, R., Mathewson, N., Syverson, P.F.: Tor: The Second-Generation Onion Router. In: *Proceedings of the 13th USENIX Security Symposium*, pp. 303–320 (2004)
12. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. *SIGKDD Explor. Newsl.* 11, 10–18 (2009)
13. Herrmann, D., Gerber, C., Banse, C., Federrath, H.: Analyzing Characteristic Host Access Patterns for Re-identification of Web User Sessions. In: Järvinen, K. (ed.) *NordSec 2010. LNCS*, vol. 7127, pp. 136–154. Springer, Heidelberg (2012)
14. Kohavi, R.: A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In: *International Joint Conference on Artificial Intelligence*, vol. 14, pp. 1137–1143. Morgan Kaufmann (1995)
15. Kumpošt, M., Matyáš, V.: User Profiling and Re-identification: Case of University-Wide Network Analysis. In: Fischer-Hübner, S., Lambrinouidakis, C., Pernul, G. (eds.) *TrustBus 2009. LNCS*, vol. 5695, pp. 1–10. Springer, Heidelberg (2009)
16. Kushilevitz, E., Ostrovsky, R.: Replication is Not Needed: Single Database, Computationally-Private Information Retrieval. In: *Proceedings of the 38th annual IEEE Symposium on Foundations of Computer Science*, pp. 364–373. IEEE Computer Society (1997)
17. Lu, Y., Tsudik, G.: Towards Plugging Privacy Leaks in the Domain Name System. In: *Proceedings of the Tenth International Conference on Peer-to-Peer Computing (P2P)*, pp. 1–10. IEEE (2010)

18. Manning, C.D., Raghavan, P., Schütze, H.: *Introduction to Information Retrieval*. Cambridge University Press, Cambridge (2008)
19. Padmanabhan, B., Yang, Y.: Clickprints on the Web: Are there signatures in Web Browsing Data? (October 2006), <http://knowledge.wharton.upenn.edu/papers/1323.pdf>
20. Raghavan, B., Kohno, T., Snoeren, A.C., Wetherall, D.: Enlisting ISPs to Improve Online Privacy: IP Address Mixing by Default. In: Goldberg, I., Atallah, M.J. (eds.) *PETS 2009*. LNCS, vol. 5672, pp. 143–163. Springer, Heidelberg (2009)
21. Rieck, K., Laskov, P.: Language Models for Detection of Unknown Attacks in Network Traffic. *Journal in Computer Virology* 2(4), 243–256 (2007)
22. White, T.: *Hadoop – The Definitive Guide: Storage and Analysis at Internet Scale*, 2nd edn. O’Reilly (2011)
23. Witten, I.H., Frank, E.: *Data Mining. Practical Machine Learning Tools and Techniques*. Elsevier, San Francisco (2005)
24. Xie, Y., Yu, F., Abadi, M.: De-anonymizing the internet using unreliable IDs. In: *Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication*, pp. 75–86. ACM, New York (2009)
25. Xie, Y., Yu, F., Achan, K., Gillum, E., Goldszmidt, M., Wobber, T.: How dynamic are IP addresses? In: *Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM 2007)*, pp. 301–312. ACM, New York (2007)
26. Yang, Y.: Web user behavioral profiling for user identification. *Decision Support Systems* 49, 261–271 (2010)
27. Yang, Y., Padmanabhan, B.: Toward user patterns for online security: Observation time and online user identification. *Decision Support Systems* 48, 548–558 (2008)
28. Zhao, F., Hori, Y., Sakurai, K.: Analysis of Existing Privacy-Preserving Protocols in Domain Name System. *IEICE Transactions* 93-D(5), 1031–1043 (2010)
29. Zipf, G.K.: *The psycho-biology of language. An introduction to dynamic philology*, 2nd edn. M.I.T. Press, Cambridge (1968)

Smartphone Forensics: A Proactive Investigation Scheme for Evidence Acquisition

Alexios Mylonas¹, Vasilis Meletiadiis¹, Bill Tsoumas¹, Lilian Mitrou^{1,2},
and Dimitris Gritzalis¹

¹Information Security and Critical Infrastructure Protection Research Laboratory
Dept. of Informatics, Athens University of Economics and Business (AUEB)
76 Patission Ave., Athens, GR-10434 Greece
{amylonas,meletiadiisv,bts,dgrit}@aueb.gr

²Dept. of Information & Communication Systems Engineering,
University of the Aegean, Samos 83200, Greece
l.mitrou@aegean.gr

Abstract. Smartphones constantly interweave into everyday life, as they accompany individuals in different contexts. Smartphones include a combination of heterogeneous data sources, which can prove essential when combating crime. In this paper we examine potential evidence that may be collected from smartphones. We also examine the available connection channels for evidence transfer during a forensic investigation. We propose a Proactive Smartphone Investigation Scheme that focuses on ad hoc acquisition of smartphone evidence. We also, take into consideration the legal implications of the proposed scheme, as it is essential that the scheme includes prevention mechanisms, so as to protect individuals from misuse by investigators or malicious entities.

Keywords: Smartphones, Forensics, Digital Evidence.

1 Introduction

Smartphones, as ubiquitous devices, merge with a person's everyday life. As a recent report¹ points out, smartphone sales outnumbered these of feature phones, thus, acquiring a significant user base. Smartphones are characterized by mobility, context-awareness, and diversity on the data sources that they integrate.

In a crime investigation context, the aforementioned characteristics can be used for forensic purposes, not only after a crime, but even proactively. For instance, in some crimes, which the in place legal and regulatory context regards as 'severe' (e.g. crimes against public or the state, pedophilia, etc.), proactive acquisition of smartphone data may be required. Currently, in a Lawful Interception (LI) of cell phones, Law Enforcement Agencies engage via the carrier's infrastructure, the interception of specific data, such as

¹ International Data Corporation (IDC). Smartphones Outstrip Feature Phones for First Time in Western Europe as Android Sees Strong Growth in 2Q11, September 2011.

phone calls, messaging services, and network data traffic [3]. The need for direct and in-time access to forensics data is also present in other technological contexts.

In the organizational context, Grobler et al. [5] refer to proactivity, as “creating or controlling a situation rather just responding to it”. They stress that a Proactive Digital Forensics (ProDF) as a process, ensures the facilitation of the investigation in a successful and cost-effective manner for enterprise systems. ProDF refers, for example, to the deployment of *forensic readiness* processes [15], which aim to maximize an environment’s ability to collect credible digital evidence and, at the same time, minimize the forensics cost during incident response. These processes incorporate tools and techniques for active/live forensics that acquire volatile data for detecting criminal activity [14].

In traditional, i.e. post mortem, forensics the need for *forensic triage* [9], [13] emerges. Forensic triage refers to the notion of *on-site forensics*, which allows an investigator to directly assess a crime scene. Assessment is feasible for a subset of the available digital evidence. Consequently, delays on results, stemming from time-consuming processes in the forensics lab, are avoided. Furthermore, in on-going crime investigations, forensic triage can assist an investigator to determine critical issues, such as subjects in immediate need or the suspect’s call activity [17], and act as appropriate. Also, mobile live memory forensic techniques focus on the acquisition of volatile data, which reside in device’s memory [16].

In this paper, we examine the technological aspects of proactive smartphone digital forensics. A smartphone can provide additional information, which exceeds communication data. The multitude, variety and ‘context awareness’ of smartphone data may constitute crime evidence beyond the scope of current LI systems, both in technical and legal terms. Thus, we propose a Proactive Smartphone Investigation Scheme that incorporates misuse avoidance of proactive evidence collection.

The paper is organized as follows. In Section 2, a taxonomy of smartphone evidence and evidence transport channels is presented. Section 3 proposes an Investigation Scheme for proactive smartphone forensics. Section 4 discusses legal considerations for proactive evidence acquisition on smartphones. Section 5 concludes the paper.

2 Smartphone Evidence

Smartphones host a plethora of heterogeneous data generated from hardware or software sources. This section associates smartphone data sources with evidence types and then correlates these sources with smartphone evidence transport channels. These associations will be used in the sequel for smartphone ad hoc evidence acquisition.

2.1 Smartphone Evidence Taxonomy

In order to associate smartphone data to evidence types, a data-oriented analysis was followed. The analysis used a smartphone data taxonomy presented in [12]. Data are categorized, with respect to their source, as: a) *Messaging Data*, i.e. the content and metadata (e.g. sender, delivery time, etc.) from messaging services (e.g. Short

Message Service (SMS), email etc.), b) *Device data*, i.e. data that are stored in the device storage media and are not related to any application (e.g. multimedia files, software and hardware identifiers etc.), c) *(U)SIM Card Data*, that reside in a (Universal) Subscriber Identity Module, such as IMSI² and MSIN³, d) *Usage History Data*, i.e. user logs (e.g. call logs, browsing history, etc.) and system logs kept for monitoring and debugging, e) *Application Data*, i.e. permanent or temporal data that are used during application execution (e.g. flat files, databases, etc.), f) *Sensor Data*, which are created by sensors that are found in most devices (e.g. camera, microphone, GPS), motion sensors (accelerometer, gyroscope), or environment sensors (magnetometer, proximity, light, temperature, etc.) and g) *User Input Data*, i.e. data from keystrokes, gestures, etc., which are processed on the fly, or stored in a keyboard cache for performance reasons.

The above mentioned data sources were organized in a *taxonomy of evidence types*, which derived from a set of questions - i.e. {*who, where, when, what, why, how*} [20]. These questions are being used in digital forensics literature [6], [8], [1] for evidence examination and analysis, as well as for evidence presentation in courts of law.

- *Identity Evidence*. Data identify subjects that are part of an event.
- *Location Evidence*. Data define the approximate or exact location, where an event takes place.
- *Time Evidence*. Data can be used to infer the time that an event takes place.
- *Context Evidence*. Data provide adequate context, such as user actions and activities for an event description [7], or the event nature.
- *Motivation Evidence*. Data can be used to determine event motivation.
- *Means Evidence*. Data describe the way that an event took place, or the mean that were used.

Even if we both assume that smartphone data are generated in a deterministic and undisturbed way and their acquisition is always feasible, this does not necessarily mean that evidence will always be present. Hence, we use three levels of association of *direct* relationship between a data source and an evidence type, namely: a) *Strong Correlation* to represent that data sources always (or in most cases) provide potential evidence, b) *Weak Correlation* to represent that data sources may provide potential evidence according to their state, and c) *No Correlation* to represent lack of relationship between a data source and evidence type. In this point we note that motivation evidence may be deduced *indirectly* via the combination of the other evidence types.

The rest of this section presents the correlation of the evidence types with the data sources.

- *Messaging Data*. Both traditional messaging services (e.g. SMS) and modern ones, e.g. email, often constitute potential evidence. Specifically, external communication data reveal the subjects and the communication time and, as a result, this source is strongly

² The International Mobile Subscriber Identity (IMSI) identifies the subscriber to the network.

³ Mobile Subscriber Identification Number (MSIN) is the 10-digit phone subscriber number.

correlated with time and identity evidence. In addition, the communication content may reveal location evidence, motive evidence, etc., therefore this source is weakly correlated with them as well.

- *Device Data*. Data such as system identifiers (e.g. IMEI) can be used to determine a subject from the provider's records, therefore a strong correlation with identity evidence exists. File system metadata can be used to determine time (e.g. file access time). Hence, a strong correlation with time evidence exists. Data created by the user (e.g. multimedia) may reveal motive or means evidence. In addition, device data may include sensor data as metadata (e.g. in geo-tagging). This provides a weak correlation with location, context, motivation or means of an incident.
- *(U)SIM Card Data* includes identifiers (e.g. ICCID⁴, IMSI, etc.) that uniquely identify a device owner, thus, a strong correlation exists. Other data that may be stored in this source (e.g. contact entries, SMS and LAI⁵) [8] can be used to deduce the rest evidence types, and, in this case a weak correlation exists.
- *Usage History Data* can infer the event time and the means used by a digital incident, or they can even reconstruct user events, thus implying a strong correlation with the respective evidence types. Furthermore, under certain circumstances, the wireless connection history (i.e. access point MAC logs) may be used to infer the device location⁶, implying a weak correlation with this evidence type. Finally, Bluetooth pairing logs can be used to infer whether the user is in a crowded area and, in some cases, identify subject evidence via the device Bluetooth id.
- *Application Data*. In some cases, private application data stored on the device may lead to potential evidence about an event and, thus, weak associations with the corresponding evidence exist. For instance: a) cached maps in navigation applications determine location evidence, b) cached social networking application data can be used to infer all the other evidence types depending on their content, etc.
- *Sensor Data* provide weak correlations with all evidence types, since they can be used to infer the device context. For instance, a microphone can be remotely enabled [12] and harvest speech data related to all evidence types. Nonetheless, in the case of location evidences, the correlation is strong due to the popularity and location accuracy of GPS sensors. .
- *User Input Data* can be used to identify a subject via keystroke analysis and, as a result, a weak correlation with this evidence type exists. Often, the keystroke cache content may reveal other evidence types, thus, a weak correlation with them exists.

Table 1 depicts the correlations between data sources and potential evidence types, where: (✓) stands for strong correlation, (~) for weak and (✗) for no correlation.

⁴ Integrated Circuit Card ID (ICCID) is the serial number of the (U)SIM card.

⁵ Simply put the Location Area Identity (LAI) identifies the cell where the device is in and as a result can be used to get an approximation of device location.

⁶ Via public mapping of MAC addresses to GPS coordinates, such as <http://samy.pl/mapxss/>

Finally, the above evidence types can be combined to form evidence chains when they include valid timestamps. For instance, call logs combined with cached data of a navigation application can be used to reveal or confirm a subject's alibi.

Table 1. Correlation between evidence types and data sources

Data sources	Evidence Types					
	Identity	Location	Time	Context	Motivation	Means
Messaging Data	✓	~	✓	~	~	~
Device data	✓	~	✓	~	~	~
(U)SIM Card Data	✓	~	~	~	~	~
Usage History Data	✗	~	✓	~	✗	✓
Application Data	~	~	~	~	~	~
Sensor Data	~	✓	~	~	~	~
User Input Data	~	~	~	~	~	~

2.2 Evidence Transport Channels

Smartphones can use four data transport channels (or interfaces) that provide different transport services. This section discusses their ability to support evidence transfer during a proactive forensic investigation.

1. *GSM Messaging interface* (e.g. SMS, etc.) provides a remote channel appropriate for small volume data transfers, which is nearly always available. Apart from the restriction in volume, another refers to cost. Increased cost a) may limit the messaging service availability, thus, large data may not be transferrable and b) may alert suspects, who thoroughly check their carrier bills in the case a proactive investigation taking place.
2. *Personal Area Network (PAN) interface* (e.g. Bluetooth, IrDA, etc.) provides a cost free, ad hoc, remote data channel, appropriate if the data collector is in the smartphone's proximity. By not relying on any base station existence, it avoids network monitoring mechanisms, such as Intrusion Detection System (IDS). Furthermore, as this channel requires no cost, it is stealthier than others. Potential shortcomings are: a) distance constraint between the device and the collector, which increases attack complexity (e.g. Bluetooth range is 10 meters), b) the average transfer speed and c) the requirement for a pairing bypass without alerting the smartphone user.
3. *WLAN interface* (e.g. Wi-Fi) provides a fast, remote channel that is appropriate for any data volume often without a cost. Due to the general popularity of Wi-Fi, availability is considered high. A shortcoming is that data transfer speed and availability rely on the distance from a base station. This distance, though, is considerably larger than the PAN's requirement, thus, it does not add considerable complexity on evidence collection.
4. *Cellular network (CN) interface* provides a data transport channel of variable speed, which is dependent on the supported carrier network technology (e.g. GPRS, HSDPA, etc.). A CN channel is not restrained by the antenna range. It

provides greater mobility than any of the aforementioned channels, since the smart-phone may travel through cells. Nonetheless, this channel is not considered suitable for large data volume, as: a) it suffers from connection drops, b) the network speed may vary as the user moves inside a cell or visits others, and c) this channel use has considerable cost, and thus it may be discovered by the owner.

Table 2 summarizes each channel’s ability to effectively transfer each source data volume (hereto referred as ‘volume’). For the association notation three symbols were used, i.e.: 1. (?) transfers small subset of the data source, 2. (~) transfers most data in this data source and 3. (✓) can transfer source type. Obviously, the WLAN channel is able to transfer all data sources. The rest of the associations are listed below:

- *Messaging data* are not volume intense, thus, they can be transferred by all channels.
- *Device data* include data with different volume requirements, ranging from small files to high definition videos. Hence, the GSMMe channel can only convey a subset of this source, whereas PAN and CN can convey, in general, this data source. Nonetheless, the latter are limited by time and transfer cost respectively.
- *(U)SIM Card Data* is not volume demanding, and can be transferred by all channels.
- *Application Data* volume requirements range from few Kbytes up to hundreds of Mbytes. As a result, PAN and CN (provided that available quota exist) are able to transfer this data type. GSMMe can only transfer application data that are not volume intense.
- *Usage History Data* are not always volume demanding (e.g. recent call logs) and, hence, can be transferred by GSMMe. Nonetheless, this data type includes OS logs, which are data volume intense and are transferable by PANs and CNs (if available bandwidth quota exists).
- Similarly to *Usage History Data*, *Sensor data* can be a) either not volume intense (e.g. GPS coordinates) and, thus, transferrable by GSMMe, or b) volume intense (e.g. accelerometer data) that can be transferred by PAN and CN.
- *User Input Data* volume requirements range from a few Bytes, which are transferable even by GSMMe, up to several Mbytes (e.g. keystroke dictionaries) that are transferable by PANs and CNs.

Table 2. Correlation between transport channels and data sources

Data type	Transport channel	GSMMe	PAN	WLAN	CN
Messaging Data		✓	✓	✓	✓
Device data		?	~	✓	~
(U)SIM Card Data		✓	✓	✓	✓
Application Data		?	✓	✓	~
Usage History Data		?	✓	✓	~
Sensor Data		?	✓	✓	~
User Input Data		?	✓	✓	✓

3 Proactive Smartphone Forensics Investigation

This section focuses on proactive forensics, where ad hoc acquisition of smartphone evidence takes place. A proactive smartphone forensics investigation may take place for the investigation of crimes considered ‘severe’ by the legal and regulatory context in place (e.g. crimes against public or the state, pedophilia, etc.). In such cases, the creation and defence of an investigation hypothesis⁷ may use the aforementioned associations of smartphone sources with transport channels and evidence types. In this context, an investigation scheme is presented and outlined.

3.1 Proactive Smartphone Forensics Scheme

The proposed scheme consists of three entities (Fig. 1), namely: a) subject(s) carrying out a proactive smartphone forensic investigation (hereinafter ‘investigator’), b) an Independent Authority (IA), and c) the investigation’s subject(s) (hereinafter ‘suspect’).

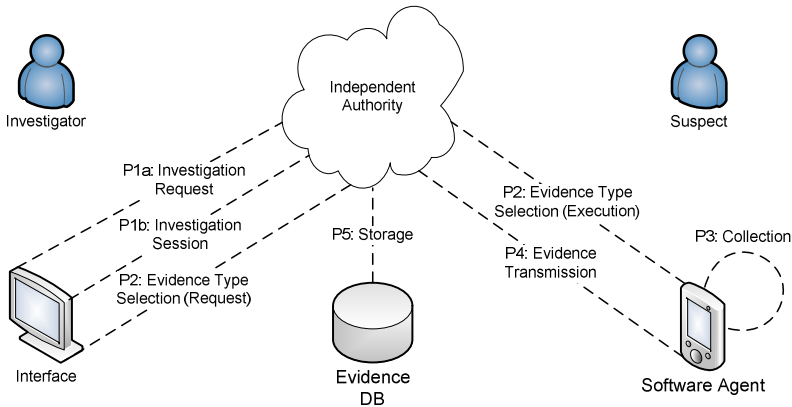


Fig. 1. Proactive Smartphone Forensics Scheme

As depicted in Fig. 1, the *IA* is the scheme’s corner stone, as it: a) controls evidence collection from the Software Agent (SA) that resides in the suspect’s smartphone, b) handles evidence storage for a time period compliant with existing laws and regulations, and c) authorizes investigator’s requests for a proactive forensic investigation against individuals.

This architecture was selected so as to hinder investigators, or other individuals, from misusing the evidence collection mechanism for profiling and intelligence gathering reasons (see Section 4).

Hence, it is assumed that the *IA* allows a proactive digital forensic investigation to take place only in suspected ‘severe’ crimes. It is also assumed that the *IA* maintains a

⁷ Hypothesis is a report based on the examination and the analysis of collected evidence that are admissible to court.

database, where evidence data are stored and protected, in terms of forensic soundness and confidentiality.

The *investigator's* role in this scheme is to create a hypothesis, i.e. collect adequate evidence suitable for use in courts of law. An investigator may request from the IA authorization for a proactive smartphone forensics investigation to take place when: a) other mechanisms for data collection are either incapable to gather the required data (e.g. cases where the suspect's context is required), or they collect data of reduced accuracy, inadequate for evidence presentation in a court of law (e.g. the location accuracy of the GPS sensor is greater than the approximate location provided by the cell phone provider), and b) the suspected crime is considered 'severe' by laws and regulations.

In this scheme, it is assumed that a Software Agent (SA), controlled by the IA, is present in the *suspect's* smartphone. Also, the IA has the capability to commence the collection of evidence types from selected smartphone sources. The scheme's architecture is depicted in Fig. 1, while its processes are described in the following section.

3.2 Scheme Processes

The proposed scheme consists of six building blocks - processes, namely: 1) *investigation engagement*, 2) *evidence type selection*, 3) *evidence collection*, 4) *evidence transmission*, 5) *evidence storage* and 6) *investigation completion*.

The first process takes place once per suspect investigation, while the remaining processes are iterative and incremental. The last process (*Process 6: Investigation Completion*) is completed when the investigation's requirements are satisfied, i.e. the hypothesis can be created and its validity can be defended in the courts of law, or when the IA's granted permission becomes invalid. The other processes are described in detail in the sequel.

Process 1: Investigation Engagement. It consists of two sub processes that define an investigation's details:

- *Process 1.1: Investigation Request (IR).* This is the formal request of investigation authorization, addressed to the IA. The request contains a definition of investigation specific parameters, such as: a) the investigator who creates the hypothesis, b) the suspects(s), c) the nature of the examined crime, d) the expected investigation period, e) the required data sources that will be collected as potential evidence, and f) the required channels for evidence transmission.
- *Process 1.2: Investigation Session.* Once an Investigation Request is submitted to the IA, a non-automated process determines the investigation's permission level upon the suspect's smartphone. This permission level is determined by the IR details and the evaluation criteria of each IA, which are dependent on the regulatory context. If the IR is accepted, an investigation session is provided to the investigator that is used for the following processes.

Process 2: Evidence Type Selection. This process is initiated after the investigation engagement process. It refers to the selection of evidence types and the corresponding transport channels by the investigator. This selection is carried out based on the two associations presented previously: (1) between smartphone data and digital evidences, and (2) between smartphone data and transport channels. For instance, an investigator may compile a configuration request for the SA evidence collection process. This configuration request specifies the desired data sources (e.g. motion sensors data) and transport channels (e.g. WLAN), and it is forwarded to the IA. Then, the IA executes the request by acquiring evidence from the SA, only if the configuration request parameters are conformant with the current investigation session permission level. In this way, misuse of evidence collection is avoided. Finally, this activity takes place every time the investigator makes a new request for potential evidence in the context of the same investigation session.

Process 3: Evidence Collection. It is triggered every time the SA configuration is altered. Based on configuration attributes (i.e. data source, transfer channel, interception period and duration, etc.) the SA harvests potential evidence, applies integrity mechanisms and forwards them to the evidence transmission process.

Process 4: Evidence Transmission. The transmission of evidence takes place, when the collection process ends. It is assumed that an Evidence Transmission Protocol (ETP) is applied between IA and SA. This ETP must include messages for the collection of all smartphone data sources and support the various smartphone transport channels. Furthermore, the ETP must impose security properties, such as message authenticity, integrity, liveness, and confidentiality.

Process 5: Evidence Storage. It refers to the storage of potential evidence that is received from the SA in the IA's infrastructure. The preservation of potential evidence in the forensics database must ensure their forensic soundness via employing integrity mechanisms, as well as, their confidentiality. The stored evidence is bound to be revisited during an investigation, so as to be further examined and analyzed before being presented in court. Thus, limited access (e.g. read-only) is provided to the investigator via an interface.

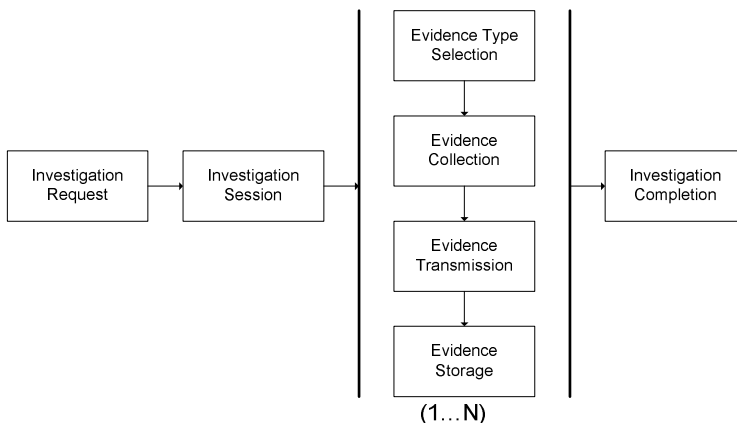


Fig. 2. Processes of the Investigation Scheme

4 Legal Considerations

A smartphone combines the features of a cell phone along with PC-like functionalities. It permits access to saved and sent messages and files and provides tools for accessing data not presently stored on the device.

Even if some Courts (like the Fifth Circuit) do not recognize any conceptual differences between searching a person's body and searching electronic equipment that this person possesses or carries with him, a smartphone stores and reveals apparently and tremendously more information, thereby providing law enforcement with access to information that a person would never carry in her pocket [4].

A smartphone itself, alone or in its technical networking, can contain personal data to such a degree and in such diversity that it may provide a revealing picture of her personality, and/or facilitate insight even into the core area of private life of a person. With the increasing use of such devices for every kind of communications, including social networking, the importance of digital evidence gathering is increasing as well. A smartphone offers law enforcement authorities "a window into their suspect" not only via hard evidences but also through the character and habit information it may provide [11].

Accessing, searching, and using as evidence the data communicated, accessed or stored by a smartphone, poses new challenges to courts and legislators that are far reaching and go far beyond the secrecy of telecommunications. Neither regulatory regimes concerning the monitoring and recording of communication content, nor rules providing for the retention of external communications data (traffic data) is deemed appropriate and/or sufficient to deal with evidence acquisition for smartphone forensics, as interception refers to the surveillance of a communication taking place between communication partners.

Sensor data or User Input Data could be deemed as biometric data, enabling the collecting of sensitive data and the profiling of the person concerned. Different legal standards and requirements apply to the lawful interception of communications by law enforcement authorities than to the recovery of data stored on a smartphone.

Especially if the data provided by such a device are remotely searched or are not presently stored with the confines of the device, it appears that the acquisition evidence appears to be comparable to a search of premises [19]. The need for review of existing regulatory concepts and converged regulatory regimes in order to face modern converged communication and IT systems in a consistent way is obvious.

The German Constitutional Court has recently (2008) placed strict limits upon the ability law enforcement authorities to remotely access computers, PDAs and mobile phones. The Court has specified the rights to "informational self-determination" and "absolute protection of the core area of the private conduct of life" and has lifted "security and integrity of information systems" as a fundamental right of the user, expressing a right to the unhampered development of her personality in the information era [18]. On the other side it is not clear if the Fourth Amendment of the US Constitution affords reasonable expectation of privacy and protection against unreasonable searches and seizures to a person who uses a smartphone.

Evidence acquisition and surveillance in the digital world need to be adapted to keep pace with technological progress. Legislators have to constantly struggle to keep up with technology and to new risks and challenges. The proposed smartphone investigation scheme is valuable for combating crime and security threats through proactiveness.

Proactive, routine data and preservation reflects the transformation from the traditional constitutional model of gathering conclusive evidence of wrongdoing of suspect individuals toward a model of intelligence gathering where information is collected at random on all users [10].

Both legislators and IT-designers should take care to prevent the deployment of technology that treats all users as potential criminals or suspects without any cause [2].

Therefore, the proposed proactive forensic investigation scheme has been conceived and designed with protective mechanisms in place that hinder investigators or other malicious individuals, from misusing it, and, in a way that allows the acquisition and preservation of data, without infringing fundamental rights.

5 Conclusions

Proactive digital forensics is, per se, an interdisciplinary task, requiring the cooperation of computer and law scientists. In this paper, we examined the technological aspects of proactive smartphone digital forensics.

We studied associations between the smartphone data sources and evidence types, as well as the available connection channels in order to deliver potential evidence from a smartphone.

Then, a proactive smartphone forensics scheme was developed. By using this scheme, the appropriate person can collect smartphone data that are essential for combating 'severe' crimes.

We also examined the current legal and regulatory framework concerning ad hoc data acquisition from smartphones. We stressed that even though our proposed investigation scheme can be an essential tool in combating specific types of crime, its application and legality lies clearly with the presence of strong protection mechanisms that ensure its lawful use. Otherwise, this scheme could be used as a tool for profiling or intelligence gathering, thus violating fundamental rights of individuals.

Future work will include the implementation of the proposed scheme with a focus on the Software Agent and the Evidence Transmission Protocol. We also plan to evaluate the implementation performance in real world scenarios, explore alternative misuse avoidance mechanisms and add stealthiness techniques to the agent. Finally, we plan to elaborate more on the proposed taxonomies by identifying subgroups in each smartphone data source and their respective correlations.

Acknowledgements. This research has been co-funded by the European Union (ESF) and Greek national funds, through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (Program HERACLEITUS II: Investing in knowledge society through the European Social Fund).

The authors would like to thank Marianthi Theoharidou (AUEB) for her valuable contribution.

References

1. Blackwell, C.: An Investigative Framework for Incident Analysis. In: Peterson, G., Sheno, S. (eds.) *Advances in Digital Forensics, Part VII*, pp. 23–34. Springer, Heidelberg (2011)
2. Brown, I.: Regulation of Converged Communications Surveillance. In: Nyeland, D., Goold, B. (eds.) *New Directions in Surveillance and Privacy*, pp. 39–73. Willan (2009)
3. European Telecommunications Standards Institute (ETSI): Lawful Interception; Requirements of Law Enforcement Agencies. Technical Specification 101: 331 (2009)
4. Gershowitz, A.: The iPhone meets the Fourth Amendment (2008), http://works.bepress.com/adam_gershowitz/3/
5. Grobler, C., Louwrens, C., Von Solms, S.: A Multi-component View of Digital Forensics. In: Aleksey, M., Ghernaouti-Helie, S., Quirchmayr, G. (eds.) *International Conference on Availability Reliability and Security (ARES 2010)*, pp. 647–652 (2010)
6. Jeong, S.C.R.: FORZA – Digital forensics investigation framework that incorporate legal issues. *Digital Investigation* 3(suppl.1), 29–36 (2006)
7. Lane, N., Miluzzo, E., Ly, H., Peebles, D., Choudhury, T., Campbell, A.: A survey of mobile phone sensing. *IEEE Communications Magazine* 48(9), 140–150 (2010)
8. Jansen, W., Ayers, R.: Guidelines on cell phone forensics. NIST Special Publication 800: 101 (2007)
9. Mislan, R.P., Casey, E., Kessler, G.C.: The growing need for on-scene triage of mobile devices. *Digital Investigation* 6(3-4), 112–124 (2010)
10. Mitrou, L.: Data Retention: a Pandora Box for Rights and Liberties? In: Acquisti, A., De Capitani di Vimercati, S., Gritzalis, S., Lambrinouidakis, C. (eds.) *Digital Privacy: Theory, Technologies and Practices*, pp. 410–433. Auerbach Publications (2008)
11. Morrissey, S.: *IOS Forensic Analysis: for iPhone, iPad and iPod Touch*. Apress (2010)
12. Mylonas, A.: *Smartphone spying tools*. M.Sc. Thesis, Royal Holloway, University of London (2008)
13. Rogers, M., Goldman, J., Mislan, R., Wedge, T., Debrot, S.: Computer forensics field triage process model. In: *Proceeding of the Conference on Digital Forensics Security and Law*, pp. 27–40 (2006)
14. Sutherland, I., Evans, J., Tryfonas, T., Blyth, A.: Acquiring Volatile Operating System Data Tools and Techniques. *SIGOPS Operating System Review* 42(3), 6–73 (2008)
15. Tan, J.: Forensic readiness. @ Stake Technical Report (2001)
16. Thing, V., Ng, K.-Y., Chang, E.-C.: Live memory forensics of mobile phones. *Digital Investigation* 7(suppl.1), 74–82 (2010)
17. Walls, J.: Forensic Triage for Mobile Phones with DEC0DE. In: *USENIX Security Symposium* (2011)
18. Wiebe, A.: The new Fundamental Right to IT Security - First evaluation and comparative view at the U.S., *Datenschutz und Datensicherheit*, pp. 713–716 (2008)
19. Wiebke, A.: Agents, Trojans and tags: The next generation of investigators. *International Review of Law, Computers & Technology* 23(1-2), 99–108 (2009)
20. Zachman, J.A.: A framework for information systems architecture. *IBM Systems Journal* 26(3), 276–292 (1987)

Modeling Social Engineering Botnet Dynamics across Multiple Social Networks^{*}

Shuhao Li^{1,2,3}, Xiaochun Yun^{1,2,**}, Zhiyu Hao¹,
Yongzheng Zhang¹, Xiang Cui^{2,3}, and Yipeng Wang^{1,3}

¹ Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China

² Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
yunxiaochun@iie.ac.cn

³ Graduate University of Chinese Academy of Sciences, Beijing, China

Abstract. In recent years, widely spreading botnets in social networks are becoming a major security threat to both social networking services and the privacy of their users. In order to have a better understanding of the dynamics of these botnets, defenders should model the process of their propagation. However, previous studies on botnet propagation model have tended to focus solely on characterizing the vulnerability propagation on one infection domain, and left two key properties (cross-domain mobility and user dynamics) untouched. In this paper, we formalize a new propagation model to reveal the general infection process of social engineering botnets in multiple social networks. This proposed model is based on stochastic process, and investigates two important factors involved in botnet propagation: (i) bot spreading across multiple domains, and (ii) user behaviors in social networks. Furthermore, with statistical data obtained from four real-world social networks, a botnet simulation platform is built based on OMNeT++ to test the validity of our model. The experimental results indicate that our model can accurately predict the infection process of these new advanced botnets with less than 5% deviation.

Keywords: network security, social network, social engineering attack, botnet, propagation model.

1 Introduction

Botnets, which are considered to be the platforms of cyberattacks, have the ability to send spams, launch DDoS attacks and steal sensitive information. They have become a growing risk with the potential to severely impact important infrastructure components like communication systems. In recent years specifically, the rapid development of social networking services (SNS) [1] and the diversification of social engineering attacks (SEA) [2] demonstrate that new widely spreading botnets have become an emerging threat to social networking services

^{*} This work was supported by "The National High Technology Research and Development Program of China" (863 programs: No.2007AA010501) and "The National Natural Science Foundation of China" (No.60703021, No.61003261 and No.61070185).

^{**} Corresponding author.

and personal privacy in social networks (SN). Comparing to traditional botnets, the evolved variety have the ability to not only steal and abuse social network users' digital identities, but also to disseminate a large number of advertising messages in social networks. In addition, they can also employ SEA to spread bots across multiple social networks. Therefore, the infection scope is significantly expanded by these new botnets. For example, the Koobface botnet [3], which was initially launched in the scope of Facebook and MySapce, is one of the most popular botnets in social networks. Up to now, it has generated 56 variants and infected more social networks than originally anticipated ones.

We believe that new advanced SEA botnets (similar to Koobface, but more destructive) will emerge in large numbers, resulting in wide-scale damage to social networks. This belief is based on the following observations:

1) These botnets have the ability to capture bots by using SEA, which is much easier than other intrusion means. SEA greatly reduces attacking difficulty for hackers, although it requires the participation of users to complete the injection of bot programs.

2) With the characteristic of cross-domain propagation, the botnets are able to disseminate bot programs from one social network to another, which makes the infection rate of the bot programs much wider and faster.

In order to know more about the evolution and dynamics of botnets and how to mitigate them effectively, defenders should first be able to measure and predict the size of botnets. Therefore, modeling botnet propagation has become an important research direction. In this paper, we propose a propagation model based on stochastic process, to describe the complex infection process of new botnets in social networks. In order to develop a more accurate description than those in previous attempts, both the properties of botnet cross-domain spreading and user dynamics (the variation of active users and users' responses) in social networks are investigated in our model. We believe the proposed model in this paper could be a practical and effective tool for social network administrators and botnet defenders, who are currently struggling against such botnets (*e.g.*, Koobface).

Our contributions are summarized as follows:

1) We have formalized a new propagation model to predict the general infection process of new social engineering botnets in social networks based on stochastic process. Two factors (the cross-domain spreading and user dynamics) found to significantly influence the infection of botnets, are well investigated in our proposed model. It was able to predict the infection process of these new botnets with a small deviation.

2) We have provided a botnet simulation platform based on OMNeT++ [15] with the statistical data obtained from real-world social networks. It was applied to simulate and evaluate the propagation process of these new botnets. In the proposed platform, multiple behaviors of social network users are simulated with a guarantee of efficacy, which is considered to be superior to other botnet simulators [5] in the ability of generalization.

The remainder of this paper is organized as follows: Some related work is discussed in Section 2; and Section 3 is dedicated to the background on these new botnets; the proposed propagation model is introduced in Section 4; the simulation evaluation and experimental results are provided in Section 5; several limitations of our work are discussed in Section 6. Finally, we conclude the paper while highlighting the scope of future work.

2 Related Work

Several propositions in the literature exist for modeling malware propagation. In [11], Zou *et al.* presented an email worm model, which revealed the significance of user behaviors by considering email checking time and the probability of opening email attachments. In [12], Yan *et al.* identified some parameters that are related to malware propagation in social networks. In [13], Cheng *et al.* proposed an equation-based model to analyze the mixed behaviors of delocalized and ripple based propagation, which is related to hybrid malware in generalized social networks. According to our knowledge, most of previous work introduced the propagation models based on a single infection domain, leaving cross-domain spreading and user dynamics unconsidered. However, there are a few exceptions. For example, Dagon *et al.* [14] used time zones to model the propagation process of some botnets, arguing that victims turn their computers off at night, which leads to diurnal properties in botnet activity. But their study represents only one factor involved in user dynamics.

3 Scene Analysis

3.1 Background

An SNS is an online service, platform, or site which focuses on the building and reflecting of social networks or social relations among people. There are many popular social networks (*e.g.*, Facebook, Twitter, Gmail and Tencent QQ) with hundreds of millions of registered users connected through friendship links. The friendship link is generally a binary relationship, representing two users as friends. Usually, social network users connect, express themselves, or share information with social network messages. These messages may have different forms in different social networks, but typically have a similar function. Meanwhile, social network messages can be also exploited for the purpose of spreading malwares by attackers, especially social engineering hackers. These malwares can launch SEA, which relies on social disguises, cultural ploys and psychological tricks to persuade computer users in assisting hackers with their illegal intrusion or use of computer systems and networks [2].

To demonstrate the efficacy of our proposed model, we deployed several SNS crawlers to capture the statistical data of social network users' behaviors, which includes access time, online duration, and the interval time prior to users' responses. Finally, based on the social network generation algorithm in [4], we built a simulation platform based on multiple virtual social network domains, which allows us to evaluate our propagation model in a controlled environment.

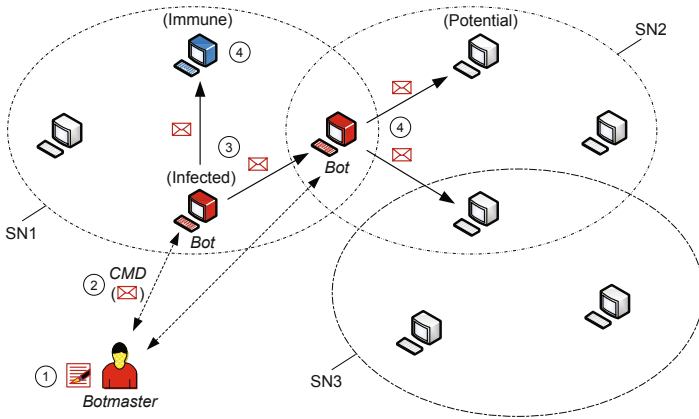


Fig. 1. The general process of SS-botnets' propagation

In our experiment, we designed the user model according to real statistics obtained from our SNS crawlers, and then simulated the whole infection process of these new botnets. Our experimental results show that our model can yield an accurate prediction on the size of the new botnets, on the condition of the topology information obtained from related social networks and the information of the initial infected nodes detected in the social networks.

3.2 The SS-Botnet Propagation Process

To more clearly describe this issue, we first provide two key definitions:

Definition 1. A *trap message* is a social network message which is constructed with either bot programs already embedded within, or with a means of accessing them (e.g., download URL). This message is forged with a social engineering approach, and designed to appear attractive to victims.

Definition 2. The botnet making use of trap messages for the purpose of spreading bots in one or more social networks is referred to as a **Social network & Social engineering botnet (SS-botnet for short)**.

It can be seen that SS-botnets represent a class of advanced widely spreading botnets. Generally, the propagation process employed by SS-botnets can be divided into four steps. Fig. 1 illustrates the infection process of an SS-botnet across three social networks.

Step 1. Botmasters fabricate a trap message, which is similar to normal social network messages and attractive to receivers.

Step 2. Botmasters construct the command (marked as *CMD* in Fig. 1) with the trap message, and send *CMD* to the bots under the control of them.

Step 3. The target bot extracts the trap message from *CMD*, and then impersonates the victim user to send the trap message to all the friends of the user. We set this time to t_s .

Step 4. The victim's friends start to deal with the trap message after a uncertain time interval (assume this time is t_r). Generally, they have two choices: believing the trap message (by activating the bot program within the trap message), or avoiding it (by recognizing it as a threat). In the former case, if the antivirus software can't detect this threat, their computers will be infected and become bots, attacking other computers (Step 3). In the latter case, the users' computers become immune to the botnet infection due to the users' security awareness and exercise in judgement.

For the initial stage of SS-botnets' propagation, botmasters can anonymously register several accounts in one or more social networks to disseminate trap messages and capture the first group of initial bots. In addition, if the propagation is blocked by an effective defense, botmasters have the ability to design new bot programs and trap messages for the purpose of bypassing this defense. As such, the SS-botnet can continue with high-speed infection, and maintain their ability to attack.

3.3 Affecting Factors

We believe that there are two key factors affecting the propagation of SS-botnets as follows.

1) Cross-Domain Spreading

A *cross-domain node* is a social network node with the ability to allow its user to use more than one social networking service. Since the users of cross-domain nodes have access to multiple social networking services, they typically have different friends in different social networks. To illustrate the threat involved with this, envision the following scenario: botmasters design the bot program to disseminate the trap message from one social network to another. This will make the infection radius of SS-botnets wider and the rate faster. Therefore, cross-domain spreading should be taken into full account while developing the SS-botnet propagation model.

2) User Dynamics

The Variation of Active Users. Intuitively, if a social network node is not active (*i.e.*, its user isn't online in any social network at this time), it is impossible to infect others or to be infected by others. Therefore, the time of users' visiting social networks, and how long they remain active on each network, have a great impact on the infection process of SS-botnets.

Users' Responses. The general process of the SS-botnet propagation requires users' responses to the trap messages propagated by bot programs, although the response behaviors are often unintentional. It would seem that when the receivers deal with the trap message and the probability that users are deceived by trap messages also have a great effect on the infection process of SS-botnets.

4 Modeling Methodology

4.1 Related Notations

The related notations of our propagation model are provided in Table [1](#).

Table 1. The notations in the proposed model

Notation	Explanation
n_i	The node with index i
V	The set of all nodes under consideration
K	The number of social network domains
D_k	The social network domain with index k
$e_{i,j,k}$	The variable indicating the relationship between n_i and n_j in D_k
$p_{i,t}$	The probability that n_i is infected at time t
I_t	The set of bots at time t
tw_i	The time interval before n_i 's response
$\alpha_{ij,t}$	The connectivity between n_i and n_j at time t
$Act(i,k,t)$	The function indicating whether n_i is active in D_k at time t
β_i	The immunity to bots for n_i
S_0	The set of the original infected nodes

4.2 Theoretical Description

We assume an SS-botnet can spread bots in K social network domains and the topologies of these domains are known. The set V is given by

$$V = \bigcup_{k=1}^K D_k \quad (1)$$

There are cross-domain nodes in the set of V , which belong to more than one domain. We can formulate this case as follows:

$$\exists k_1, k_2 \in [1, K] \rightarrow D_{k_1} \cap D_{k_2} \neq \emptyset \quad (2)$$

A cross-domain node has the ability to be active in more than one domain simultaneously, a single domain, or no domain at any time. In the propagation process of SS-botnets, it is possible that a node is in the offline state at a given time (because the user may not be online in any social network). Obviously, the infection can't work for the offline nodes. Without loss of generality, we assume D_0 is a special domain where offline nodes reside.

We assume the time interval from sending the trap message (t_s) to the receiver's response (t_r) is tw . And $tw = t_r - t_s$. From a statistical point of view, it can be found that tw follows a heavy-tailed distribution according to [6]

$$Pr(tw) = \frac{\tau}{(tw)^{\tau+1}} \quad (3)$$

Where $tw > 1$ and $\tau > 0$. In addition, τ determines the mean of tw , which have different values for different social networks.

We employ the discrete-time stochastic process to model the propagation process of SS-botnets. The justification for employing the stochastic process is as follows: (i) The cycle time of sending a trap message, which has been fixed in the design of bot programs, can be discretized. (ii) The parameter tw can be discretized, which won't alter the characteristics of the tw distribution.

n_i is not infected at time t if and only if it was not infected at time $(t-1)$, and effectively blocked or ignored all trap messages sent by its friends in the same social network domain at time $(t-tw_i)$. These events are independent, therefore $p_{i,t}$ can be expressed as

$$1 - p_{i,t} = (1 - p_{i,t-1}) \cdot \prod_{j \neq i} (1 - \beta_i \cdot \alpha_{ij,t} \cdot p_{j,t-tw_i}) \quad (4)$$

We use the approximation equation $(1-x)(1-y) \approx 1-x-y$ (when $x \ll 1$, $y \ll 1$) to simplify Equation 4. Since $p_{i,t-1} \ll 1$ and $p_{j,t-tw_i} \ll 1$, we can get

$$p_{i,t} = p_{i,t-1} + \beta_i \cdot \sum_{j \neq i} (\alpha_{ij,t} \cdot p_{j,t-tw_i}) \quad (5)$$

From Equation 5, it is seen that the current state of n_i depends on both the state of n_i at time $(t-1)$ and the states of its friends at time $(t-tw_i)$. In addition to this, the parameter $\alpha_{ij,t}$ indicates the connectivity between n_i and n_j at time t , dictating the probabilistic behavior of the stochastic process; while the parameter β_i indicates the immunity to bots for n_i , affecting the result of the infection process.

First, we analyze $\alpha_{ij,t}$, which can be stated as

$$\alpha_{ij,t} = \sum_{k=1}^K Act(i, k, t) \cdot Act(j, k, t-tw_i) \cdot e_{ij,k} \quad (6)$$

Then the function $Act()$ is given by

$$Act(i, k, t) = \begin{cases} 1 & \text{if } n_i \text{ is active in } D_k \text{ at } t, \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

Additionally, the value of $e_{ij,k}$ in Equation 6 can be determined according to the given topology of relationships in social network domains. From Equation 6, we can see that if n_i and n_j are friends in one or more domains, and n_i is active at t while n_j is active at $(t-tw_i)$, $\alpha_{ij,t} \geq 1$; 0 otherwise. And $\alpha_{ij,t} \in [0, K]$.

Next, we describe β_i , which expresses the ability of a user to assess and judge the trap message, along with the anti-bot capability of n_i . For simplifying the calculation, we set $\beta_i \in (0, 1]$. The smaller the value of β_i is, the larger the probability that the user remains uninfected will be. When β_i approaches to zero, n_i is hardly infected by bot programs, and if $\beta_i = 1$, n_i will be captured as long as it is attacked by other infected nodes with the trap message.

For the propagation process of an SS-botnet, we are concerned about the total number of bots at a given time (*i.e.*, the future size of an SS-botnet). Let I_t denote the set of infected nodes at time t , and the expectation of $|I_t|$ is given by

$$E(|I_t|) = \sum_{i=1}^{|V|} p_{i,t} \quad (8)$$

The propagation process is initiated by infecting a given number of nodes, which belong to the set S_0 . And the initial condition can be stated as

$$p_{i,0} = \frac{|S_0|}{|V|} \tag{9}$$

Where $|S_0| \geq 1$.

5 Evaluation

In this section, we firstly introduce the statistical investigation for determining $\alpha_{ij,t}$. Subsequently, we examine the ability of our proposed model to predict the propagation process of an SS-botnet, with a goal of achieving early warning. Finally, we analyze the effects of the cross-domain spreading and user dynamics.

5.1 Statistical Investigation

In order to apply the proposed model to the prediction of the real-world SS-botnet dynamics, we should make the value of the parameter $\alpha_{ij,t}$ in accordance with realistic situations. For the topological information of social networks, there are two factors affecting the value of $\alpha_{ij,t}$: (i)the visiting time and (ii)the duration of active users. The former can be determined by the number of active users in target social networks at given time point, while the latter can be modeled by a heavy-tailed distribution [6] [7]. Assume that the total number of users in a social network is more or less constant over a short period of time, and let Nt_k represent the total number of users in D_k . Furthermore, we use $Na_{k,t}$ to represent the number of active users in D_k at time t .

In our investigation, we design some crawlers for fetching the number of active users in four popular social networks. The whole operation process lasted about three months with a sampling cycle of one hour. The studied social networks are: the BBS of Graduate University of Chinese Academy of Sciences (KYXK BBS) [8], the BBS of Tsinghua University (SMTH BBS) [9], QQ Game and QQ Instant Message (QQ IM) [10]. Fig. 2 shows the statistical results. From Fig. 2 it can be found that the number of active users is typically at a low level from midnight to 8:00, and at a higher level from 13:00 to 22:00. The minimum value (at 5:00) is about 1/3 of the maximum. Furthermore, we find that the diurnal variation trends of active users in these four social networks are very similar. Therefore, our equation assessing the visitation time and the duration of users is as follows:

$$\frac{Na_{1,t}}{Nt_1} \approx \frac{Na_{2,t}}{Nt_2} \approx \dots \approx \frac{Na_{K,t}}{Nt_K} = R_t \tag{10}$$

Where R_t denotes the ratio of $Na_{k,t}$ and Nt_k . Thus $\alpha_{ij,t}$ can be calculated as follows:

$$\alpha_{ij,t} = (R_t) \cdot (R_{t-tw_i}) \cdot \left(\sum_{k \in D_{c_{ij}}} e_{ij,k} \right) \tag{11}$$

Where $D_{c_{ij}}$ indicates the common social network domains of n_i and n_j , and the value can be determined with the given topological information of the target social networks.

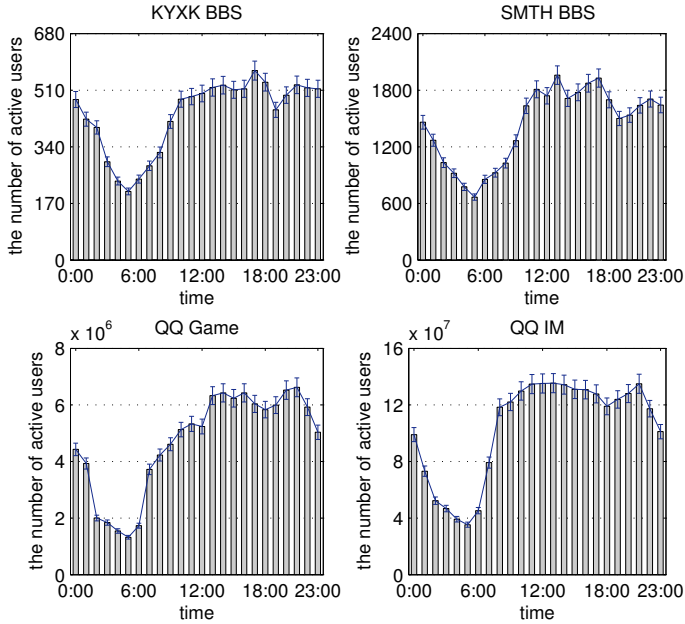


Fig. 2. The diurnal variation of active users in social networks

5.2 Simulation Experiment

OMNeT++ is an extensible, modular, component-based C++ simulation framework used for building network simulators. We’ve built a simulation platform based on OMNeT++ for the purpose of evaluating our propagation model. The simulation is made comparable to real-world situations by adjusting the values of the property parameters (*e.g.*, the out-degree and in-degree of nodes and the small average shortest path length). Thus, we can claim the three following characteristics of our simulation platform:

- 1) **High Extensibility:** this platform can simulate several virtual social networks simultaneously, and every virtual social network is generated according to the algorithm in [4].
- 2) **Fine-Grained:** the platform can simulate multiple behaviors of nodes, such as when to deal with the trap message.
- 3) **High Flexibility:** in our platform, the interval between discrete time points can be adjusted to simulate the infection processes of different SS-bontets.

In our platform, we designed a virtual SS-botnet that can spread across two social network domains, and then simulate the propagation process of the SS-botnet. Additionally, we assume β_i follows the uniform (0, 1) distribution. The values of related parameters are shown in Table 2.

Fig. 3 shows the comparison between our model and the results of two simulations for the propagation process of the SS-botnet in one and two domains.

Table 2. The values of the relative parameters in the simulation

Parameter	Value
Nt_k	10000
$ V $	10000 and 20000
K	1 and 2
$ S_0 $	1 and 100
τ in the distribution of tw	1 and 1.5
the max out-degree in D_k	100
the mean of β_i	0.7
the starting time	5:00

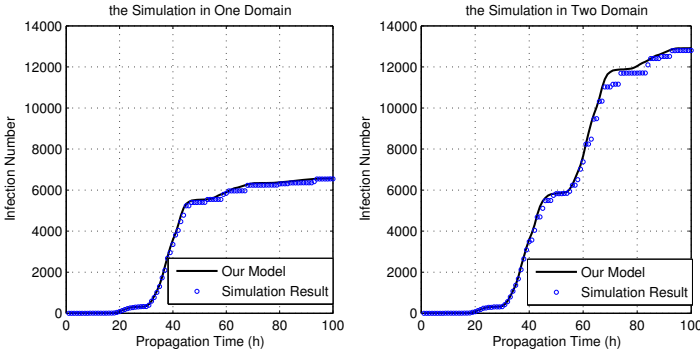


Fig. 3. The comparison between our model and the simulation results

The comparison indicates that our model can accurately predict the size of an SS-botnet with less than 5% deviation.

Furthermore, we experiment with MATLAB for evaluating the influence of different factors on the infection of the SS-botnet according to our model, which yielded some intriguing results (as shown in Fig. 4).

Fig. 4(a) illustrates the influence of the diurnal variation of active users on the propagation process of SS-botnets. Since the starting time of our simulation is 5:00, it can be found that the infection speed in the daytime is much faster than in the night, presumably because more users are offline at night. Fig. 4(b) illustrates the influence of the factor tw . If tw is not taken into consideration, the infection rate will be faster but not realistic. Fig. 4(c) presents the propagation processes of different numbers of cross-domain nodes (5, 10, 100). It can be found that the proportion of cross-domain nodes can also affect the infection rate of the SS-botnet. In other words, the more cross-domain nodes available for infection are, the faster the SS-botnet will spread. Fig. 4(d) illustrates the influence of both the degree and the number of initial infected nodes. In this figure, a *popular node* is the node with a high in-degree, which is different from a normal node. It's shown that choosing popular nodes as initial nodes or increasing initial nodes can accelerate the infection rate.

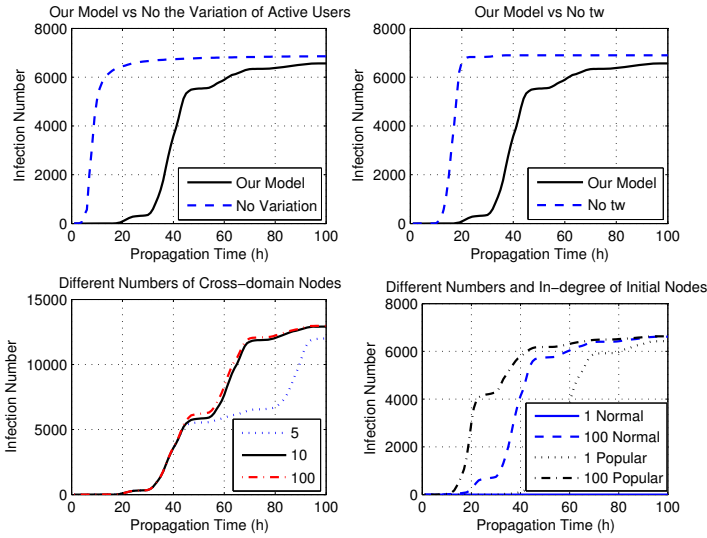


Fig. 4. The influence of different factors

6 Limitations and Discussion

In this section, we discuss some limitations of our propagation model. Firstly, in the proposed model we assume that social networks are static when considering their topological characteristics and the initial set of infected nodes. In reality, however, the number of registered users and the relationships in real-world social networks are always changing, which makes the propagation process of SS-botnets more complex. We believe these real-life variations are very important factors which should be considered in future work. Secondly, the parameter β_i follows a uniform $(0, 1)$ distribution in our model, while in reality it may follow other distributions (such as a beta distribution). In future work, we plan to investigate the distribution of β_i to optimize our model. Finally, since SS-botnets are emerging and our work is a prospective study, there is no real attack data available for the validation of our model. In addition, fetching the information of an SS-botnet’s bots is more related to the detection technologies, which is beyond the scope of this paper.

7 Conclusion

In this paper, we present a new propagation model for SS-botnets which represent an new type of emerging botnets and have a growing risk to social networking services. By exploiting the general infection process of SS-botnets, we reveal two key factors that affect the infection of SS-botnets: (i)the cross-domain spreading of bots and (ii)the dynamics of social network users. Next, we propose

the theoretical description of the proposed model based on stochastic process, and determine the values of some important parameters in our model with statistical data obtained from real-world social networks. Furthermore, we built a platform based on OMNeT++ for simulating the propagation process of SS-botnets to evaluate our model. The experimental results indicate that our model has the ability to accurately predict the size of SS-botnets with a small deviation. In future, we would like to make our simulation platform more comprehensive to support the further study of the SS-botnet dynamics. And then, we will focus more research on the ways to fight against SS-botnets. Our ultimate goal is to utilize the proposed model and our simulation platform to help defenders design effective containment mechanisms of SS-botnets.

References

1. Kwak, H., Lee, C., Park, H., Moon, S.: What is Twitter, a social network or a news media? In: 19th International Conference on World Wide Web, pp. 591–600 (2010)
2. Abraham, S., Chengalur-Smith, I.S.: An overview of social engineering malware: Trends, tactics, and implications. *Technology in Society* (2010)
3. Thomas, K., Nicol, D.M.: The Koobface botnet and the rise of social malware. In: 5th International Conference on Malicious and Unwanted Software (MALWARE), pp. 63–70 (2010)
4. Toivonen, R., Onnela, J.P., Saramáki, J., Hyvónen, J., Kaski, K.: A model for social networks. *Physica A: Statistical and Theoretical Physic.* 371(2), 851–860 (2006)
5. Ruitenbeek, E.V., Sanders, W.H.: Modeling peer-to-peer botnets. In: 5th International Conference on Quantitative Evaluation of Systems, pp. 307–316. IEEE Computer Society (2008)
6. Barabási, A.L.: The origin of bursts and heavy tails in human dynamics. *Nature* 435(7039), 207–211 (2005)
7. Sarat, S., Terzis, A.: On using mobility to propagate malware. In: 5th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks and Workshops (WiOpt 2007), pp. 1–8. IEEE (2007)
8. KYXK BBS, <http://kyxk.net>
9. SMTH BBS, <http://www.smth.edu.cn>
10. Tencent QQ, <http://www.qq.com>
11. Zou, C.C., Towsley, D., Gong, W.: Email worm modeling and defense. In: 13th International Conference on Computer Communications and Networks (ICCCN 2004), pp. 409–414 (2004)
12. Yan, G., Chen, G., Eidenbenz, S., Li, N.: Malware propagation in online social networks: nature, dynamics, and defense implications. In: 6th ACM Symposium on Information, Computer and Communications Security, pp. 196–206 (2011)
13. Cheng, S.M., Ao, W.C., Chen, P.Y., Chen, K.C.: On Modeling Malware Propagation in Generalized Social Networks. *IEEE Communications Letters* 15(1), 25–27 (2011)
14. Dagon, D., Zou, C., Lee, W.: Modeling botnet propagation using time zones. In: 13th Annual Network and Distributed System Security Symposium (NDSS 2006) (2006)
15. Varga, A., Hornig, R.: An overview of the OMNeT++ simulation environment. In: 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops, pp. 1–10 (2008)

Layered Analysis of Security Ceremonies

Giampaolo Bella^{1,2} and Lizzie Coles-Kemp^{3,4}

¹ Dipartimento di Matematica e Informatica, Università di Catania, Italy

² Software Technology Research Laboratory, De Montfort University, UK
giamp@dmi.unict.it

³ Information Security Group, Royal Holloway University of London, UK

⁴ School of Computer and Security Science, Edith Cowan University, Australia
lizzie.coles-kemp@rhul.ac.uk

Abstract. A security ceremony expands a security protocol with everything that is considered out of band for it. Notably, it incorporates the user, who, according to their belief systems and cultural values, may be variously targeted by social engineering attacks. This makes ceremonies complex and varied, hence the need for their formal analysis aimed at their rigorous understanding.

Formal analysis in turn requires clarifying the ceremony structure to build a ceremony model. The model defined here spans over a number of socio-technical layers, ranging from a computer network to society. It inspires a layered analysis of security ceremonies, that is layer by layer. This paper focuses on the human-computer interaction layer, which features a socio-technical protocol between a user persona and a computer interface. Future work will be to traverse all layers by formal analysis.

1 Introduction

The awareness that computer security is much more than a technical issue consolidated through the last decade. Today, it can be claimed that accomplishing the security goal in practice requires heterogeneous and combined efforts at least from computer scientists, social scientists, experimental psychologists, cognitive scientists, and web and HCI designers. Most recent flaws are not due to technical deficiencies but to social engineering techniques to fool the human user into making mistakes [1]. This growing awareness inspired the concept of *ceremony* a few years back, introduced by Walker and elaborated further by Ellison [2], which expands a security protocol to include whatever was left out-of band. In particular, a ceremony recognises the role played by the human, who is “likely to do incomplete comparisons of values, for example” [2]. Capturing this incomplete and partial behaviour is essential for ceremony analysis that reflects real-world interactions and behaviours. Socio-technical modelling in security [3] typically models actions on information objects. Whilst this is valuable, it does not account for the different patterns of practice that are influenced by social and personal factors. In recognising patterns of practice, we capture a range of incomplete comparisons of value that better reflect real-world behaviours.

Motivation. Many insights derive from putting a protocol in a context [45], which may involve a number of technical and social elements. A simple but representative note is that protocols involving public-key cryptography cannot be used safely in contexts where certification authorities cannot be accessed reliably. Also, the protocol users cannot be assumed to always act as anticipated by the protocol designers. These notes demonstrate the context sensitive nature of technology. They also indicate that it is more than context at work: the circumstances of technology use, the cultural values, the belief systems and the demographic factors, simplified below as the user's *persona*, play a substantial role in shaping user responses.

Cooper acknowledged human personas in the design process [6,7] but, to our knowledge, a framework that incorporates personas in the *formal analysis* of security ceremonies is not yet available — though strongly desirable. We shall see that our personas are more fine-grained than Cooper's as each user can express a variety of them in front of the same technology at different times. We recall that formal analysis means specification and verification whose syntax and semantics are mathematically founded. Traditional formal analysis of security protocols evaluates them against given security goals. This paper argues that, due to the complex nature of persona interaction with information protocols, it is valuable to graphically map out the interactions in order to visualise them prior to analysing them formally.

The complexity of ceremonies is acknowledged by many. Radke et al. observe that “a different context, even for the same set of protocols, is a different ceremony” [8]. We shall see that such a context may in turn entail various personas. For example, while approaching online banking, a user may express a very cautious persona at first, and a more relaxed one after reassurance from their friends. Karlof et al. appear to deduce that security can be established by constraining the user interaction with a number of *forcing functions* [9] such as the impossibility to continue unless a box is ticked. However, this may not always be viable, for example because there are many perspectives on privacy, and individuals construct notions of privacy in individual ways [10]. Therefore, a population of Internet service users will engage with a service provider's privacy policy in many ways, as privacy fieldstudies show [7,11]. In consequence, the challenge to develop methodologies for the analysis of general ceremonies where humans interact in non-deterministic ways is significant.

Contribution. In facing such a complexity, the first step of our research was to map the full structure of a security ceremony. The map can be viewed as a model that turns out to feature multiple layers, spanning from a computer network through the human and over to society. Depending on the analyst's focus, certain layers of the model can be conveniently collapsed and assumed to function. Layers can then be analysed in isolation, and future work is to tackle them in combination. Incidentally, “security” is used loosely here to refer to any security property, such as certification, confidentiality and also privacy. A ceremony will be termed accordingly to its main security goal, for example as a certification ceremony, a confidentiality ceremony or a privacy ceremony.

Our model enables a layered view of security ceremonies, and allows for formal analysis of ceremonies from different perspectives. The present paper concentrates on the human-computer interaction layer, III in the model. This layer features a protocol between a user persona and a computer interface, hence a *socio-technical protocol*. The analysis proceeds from the development of a graphical representation of a socio-technical protocol, which in turn demands an implicit encoding of human personas. As a result, our representation can show the various paths of execution of a protocol, each path implicitly encoding a specific persona. This is demonstrated over the socio-technical protocol of an example security ceremony whereby an Internet user registers with a service provider. This ceremony, which is widespread at present as many services require registration, sees the user enter some personal information, hence it is a privacy ceremony as the main security goal is the user's privacy.

It must be remarked that our ceremony model does not intend to prescribe a specific formal method. By contrast, it aims at providing a common canvas on which to use the methods that seem most appropriate. The method we choose for the human-computer interaction layer rests on mathematical induction for the sake of specification, and is inspired by security protocol analysis [12]. Properties of interest can then be assessed through the corresponding induction principle. Known strengths are the support offered by a tool running on a computer, a theorem prover, and the capacity to reason about systems of unbounded size.

Paper summary. A general model of security ceremony is defined (§2), and an example ceremony is outlined (§3). A graphical representation of a ceremony layer is proposed and demonstrated on an example ceremony (§4). Our method of formal analysis of ceremonies is described and demonstrated on that layer of the example ceremony (§5). Conclusions outline on-going and future work (§6).

2 A Security Ceremony Model

As mentioned above, a security ceremony expands a security protocol with the out-of-band, notably with the user [2]. However, a number of layers must be traversed for a security property that the protocol enforces to reach the human. This observation inspires the multiple-layer model of security ceremony pictured in Figure 1, which provides the basis of our formal analysis.

It can be seen that several layers are identified, going beyond the ceremonies between users and systems as described by Ellison. Our model is capable of capturing additional interaction layers between users and technology. This aim is also supported by Whitworth, who advances the importance of the interfaces between humans and machines, and acknowledges the outermost layer whereby society influences by any means, such as word of mouth and publicity, the humans' engagement with technology [13]. Therefore, we are oriented to see the workbench for a socio-technical security analysis as a finer-grained picture.

The layers in Figure 1 feature various abstractions of two example users Alice and Bob, additionally expanded with *Society* — such abstractions will be termed *players*. The (yellow) small boxes indicate the players. From right to left, they

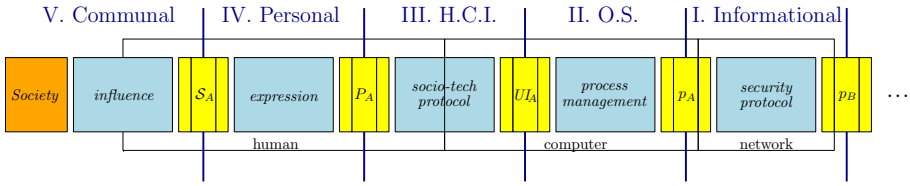


Fig. 1. The security ceremony model underlying our formal analysis

respectively are the computer process p_B running a security protocol with Alice on Bob’s behalf and the computer process p_A running the security protocol with Bob on Alice’s behalf. Then come the graphical user interface UI_A for her, a generic persona P_A of hers, and finally Alice as a human, that is, her self S_A . The layers can be understood as follows.

Layer I. Informational concerns the security protocol running between computer processes in order to secure Alice and Bob’s communications over a potentially insecure network.

Layer II. Operating System manages the inter-process communication between the process that executes the security protocol on behalf of a user and the process that runs the graphical interface presented to that user.

Layer III. Human-Computer Interaction indicates the socio-technical protocol whereby a user interacts with a graphical interface, typically by filling forms in. This is clearly a technical protocol because of the interaction with technology, but is deeply intertwined with the social protocols regulating the individuals’ expressions of social capabilities such as trust, recommendations and advice [14]. Precisely, the user is not involved directly but through one of their personas, expressed through the outermore layer.

Layer IV. Personal pertains to the user expression of a persona to engage with specific technology. According to Cooper [6], a persona is a realistic description of a user of the design, with their specific attitudes and goals. We take a yet finer-grained definition, upon the basis that a user may express various personas, as we shall see (§4). For example, arguably when accessing on-line bank services to pay for bills, users express different personas than when accessing their Facebook accounts to catch up with friends. Or they can be more unwilling to download new applications when in the middle of bank transactions than when attempting to share contents with friends.

Layer V. Communal reflects the reciprocal influence of society over individuals. For example, a national campaign could influence users towards being more careful in opening attachments from unknown sources.

Adjacent layers share a player, who plays in both layers. Each layer features what is termed an *interaction* between a pair of players, such as the socio-technical protocol or the security protocol. The (blue) rectangles indicate the interactions. Players only interact within a layer. Interacting players may not belong to the same user, as is the case of layer I. An early version of this model,

which is also inspired by Whitworth's socio-technical research [13], was already published [15]. The version presented in this paper fixes players and interaction of layer III thanks to the work presented below. Most importantly, it also points out a new interpretation of the model, where layers can be conveniently collapsed as a concertina.

3 Example Ceremony Outline

Privacy ceremonies are particular security ceremonies whose main goal is privacy. Typically, they are studied through fieldwork that relies on surveys as a traditional means of measuring privacy attitudes.

An example of privacy ceremony sees an Internet user register with a service provider. Two surveys indicated a range of privacy practices and attitudes when humans engage with on-line services [16,17]. The surveys confirmed the paradox traditionally found in privacy literature [18,19,20] that users want autonomy over on-line privacy but are prepared to trade their privacy in return for some reward. The VOME project [21] took a multi-method research approach to observe and denote the personas of the engaging users [22,23]. In noting that service users have varying senses about how their private information will be treated by the service provider, the fieldwork identified two main personas. One is goal-driven and accepts the provider's conditions in order to obtain its service. Another common one queries the provider and takes various actions to either obtain more information about the provider's privacy policy or to disengage from the registration process. Each of these personas need to be modelled in such a way that their particular security implications can be analysed.

For example, the current registration page with Amazon implements a very basic privacy ceremony: after the user information is entered, an account can be created with just one click. It is clear that simplicity is boosted with the aim of enhancing usability and therefore engagement. However, arguably those users expressing the second persona outlined above, who seek explanations and reassurances about their privacy, feel that their privacy is not being adequately treated by the present ceremony, and hence are unlikely to engage. The following sections describe our formal approach to capturing these complex interactions.

4 Representing Layer III of Security Ceremonies

Layer III, termed human-computer interaction, sees a socio-technical protocol between a persona and an interface. We develop a graphical representation of this layer taking advantage of an encoding of personas. While this is useful for the formal analysis that will follow (§5), it is valuable on its own as it may be yet more widely understandable than a formal specification. The graphical representation provides a means of extending the informal descriptions of privacy ceremonies. We give an example below based on multi-persona registration using various specified notions, highlighted in italics. Working with security senses enables the

map of persona interaction to be more expressive, enabling the formal analysis to be more nuanced in articulating different persona interactions.

With security ceremonies in general, personas may have various security *senses*. A security sense is a *feeling that contributes to an emotional or attitudinal position at a given time on the security goals that the ceremony aims at achieving*. Therefore, we shall be allowed to talk about confidentiality senses, or authentication senses or privacy senses, depending on the type of ceremony being analysed. Four basic meaningful senses, which could of course be refined and specialised, can be described:

caution is the main sense that personas have at the beginning of their interaction with an interface. This sense is often engendered by the public discussion of security concerns.

puzzlement exemplifies the perception that something may be going wrong or out of the user control. The fieldwork indicates that it may also characterise personas expressed (through layer IV) by experienced users.

confidence is the positive sense that the interaction with the (interface of the) service is going satisfactorily. A common requirement is that confidence should always accompany the completion of a session.

unconfidence is the negative sense that the interaction with the service is not going satisfactorily. A common requirement is that unconfidence should never arise, especially out of session completion.

From examples of persona interaction with the ceremony, we can abstract that security senses may arise in combination, namely in security *stances*. A stance is a set of senses, that is an *emotional or attitudinal position at a given time on the security goals that the ceremony aims at achieving*. The number of potential stances on four senses is sixteen, which is the cardinality of the powerset of the set of senses. For our treatment, it is useful to number the potential stances, for example as in Figure 2. The fieldwork confirms that some stances are practically rare, such as the empty stance, St0, or contradictory, such as those stances featuring both confidence and unconfidence, St10, St13 and St15. Other stances are rather implausible, such as those featuring both puzzlement and confidence, St8 and St12 (and obviously St15).

After security senses and stances, the possible *action names*, also termed *cues*, used in the ceremony to analyse must be defined. It must be remarked that whenever user and provider are mentioned in the following, they act respectively through their persona and interface. Our example ceremony rests on these cues:

Begins indicates a persona's wish to initiate a session with a service provider, such as typing in its URL to obtain the main interface.

Registers expresses a persona's successful completion of a session.

Aborts expresses a persona's unsuccessful completion of a transaction.

Compels indicates the interface use of forcing functions whereby a user is forced to do some action in order to proceed with the interaction. For example, the provider's privacy policy must be accepted by a tick to continue.

St0 = {}
 St1 = {caution}
 St2 = {puzzlement}
 St3 = {confidence}
 St4 = {unconfidence}
 St5 = {caution, puzzlement}
 St6 = {caution, confidence}
 St7 = {caution, unconfidence}
 St8 = {puzzlement, confidence}
 St9 = {puzzlement, unconfidence}
 St10 = {confidence, unconfidence}
 St11 = {caution, puzzlement, confidence}
 St12 = {puzzlement, confidence, unconfidence}
 St13 = {confidence, unconfidence, caution}
 St14 = {unconfidence, caution, puzzlement}
 St15 = {caution, puzzlement, confidence, unconfidence}

Fig. 2. The potential security stances

Queries represents a request of additional clarification or information in general, which both persona and interface may make through specific clicks.

Explains indicates the release of additional information that both persona and interface may do, such as the provider’s (uncommon though desirable) practice of providing additional information about their privacy policy, which could ultimately be negotiated with the user.

An *action* is an occurrence of a cue between two players, such as “P Begins I” to indicate a persona P who begins interacting with an interface I. Actions and their implications on security senses were explored in the qualitative research [23], highlighting that actions influence the security senses of a persona. In particular, each stance depends on the previous stance and on the action just taken. Therefore, it is convenient to define an *episode* as the pair consisting of an action and the stance it leads to. Upon this basis, a *socio-technical protocol* can be defined as a *list of episodes*, hence its players are identified and also the sequence of stances that the user takes throughout. Therefore, *a persona is the sequence of stances taken during a socio-technical protocol*, which form an expression of the user’s while the user interacts. The advantage of this approach is that we are able to move beyond simple action-object socio-technical expression and introduce the notion of patterns of practice influenced by different personal and social factors (represented here as sequences of stances).

The dynamics of layer III of the ceremony to analyse can be now represented as a graph, accounting for a number of personas. Our example ceremony is built on the actual fieldwork [23,22]. It maps typical responses to the request to disclose personal data. Layer III of the resulting ceremony, featuring all stances except the contradictory or implausible ones, can be represented as a graph, precisely a tree, which is in Figure 3. Reading the stances on each path describes a persona, while reading the episodes describes a socio-technical protocol.

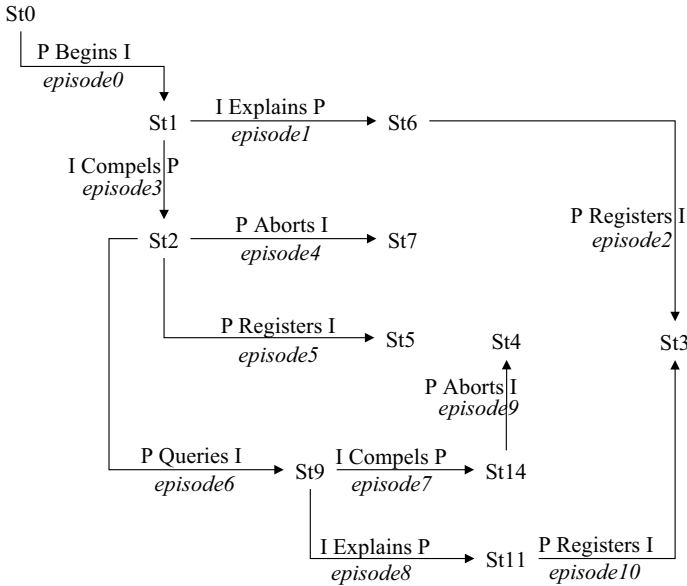


Fig. 3. Graphical representation of layer III of the example ceremony: each path encodes a specific persona interacting with a user interface, i.e. a socio-technical protocol

In particular, path St0, St1, St2, St7 is the persona that aborts with a stance of caution with unconfidence because compelled by a forcing function; path St0, St1, St2, St5 is the persona that gets a stance of caution with puzzlement following registration. The path departing from stance St1, a sense of caution, and leading to registration with stance St3, a sense of confidence, indicates the persona facing an interface where the provider explains their privacy policy adequately and then exits. Another path departing from stance St2, a sense of puzzlement, represents a persona who wants to understand how their privacy is going to be managed, and therefore poses queries. The path branches towards abortion when the provider refuses to provide adequate explanation, or towards registration when the provider accepts to dialogue with the persona on privacy.

Figure 3 communicates several aspects of the ceremony layer. With the mentality of traditional security protocol analysis, it can be understood as a protocol with branches. From the standpoint of socio-technical analysis, it is the complete layer III: each of its paths encodes specific persona and interface, and also the socio-technical protocol between them. Our representation also appears to be flexible. For example, the sequence of actions in a path could be repeated in a parallel path featuring difference stances to account for different personas taking the same actions.

The outcome of formal analysis can then be used to motivate refinements of the underlying social theory as well as modifications to the technical protocols to better support the required stances. Also, it must be remarked that our ceremony

representation does not aim at a descriptive model of human behaviour, as is often found in statistical analysis. Rather, human behaviour is represented as personas, in turn encoded as combinations of actions and stances.

5 Analysing Layer III of Security Ceremonies Formally

This Section explores the type of analysis that may be performed once the Layer III interactions are mapped. The formal analysis of layer III of security ceremonies is challenging because of the variety of socio-technical protocols and personas to account for. In order to include Layer III within formal security analysis, the ceremonies first need to be mapped so that systematic analysis can take place. Our formal analysis at this layer aims at providing a mathematically-solid ground to the analysis of ceremonies. This will help ceremony designers predict the potential effects of the various combinations of stances and actions, and to suggest possible refinements of such combinations in step with a refinement of the underlying social theory. In other words, the formal analysis can be used in many ways, such as to foresee what protocol may lead to a desired stance or, vice versa, to evaluate the stances of a protocol.

A variety of formal approaches can be taken, such as *model checking* [24] and *theorem proving* [25]. Here, we opt for the latter because we aim at deriving formal statements, supported by mathematical proof, about the security stances of the players interacting at layer III. Another reason is to answer queries of the form: if a user registers with a provider without getting any explanation, will he feel confident? The theorem prover we use is Isabelle [25], taking inspiration from vast work conducted to analyse security protocols [12]. The main specification strategy that is adopted is mathematical induction, which allows for the compact specification of systems of unbounded size, such as the natural numbers. The main reasoning strategy is the corresponding induction principle. A brief outline of Isabelle is in Appendix A.

5.1 Formal Specification of the Example Ceremony

This Section takes the graphical representation of a ceremony seen above (§4) and reformulates it using a formal language (whose syntax and semantics are mathematically defined).

A type for players and a datatype of cues are defined:

```
types player = nat
datatype cue = Begins | Queries | Registers | Aborts
             | Compels | Explains
```

Actions are defined as the cartesian product between players, cues and players. An action is thus formalised as a triple, making it possible to denote, for example, actions involving a specific cue among generic players or vice versa:

```
types action = "entity * cue * entity"
```

Another datatype formalises security senses, while a stance can be introduced by an intuitive notation:

```
datatype sense = caution | puzzlement | confidence | unconfidence
types stance = "sense set"
```

We also decide to formalise *episodes* as the cartesian product of actions and stances. An episode then is a pair consisting of an action and a stance:

```
types episode = "action * stance"
```

In order to explain our method, it is useful to introduce two example players, that is to define two constants:

```
consts PP::entity
consts II::entity
```

Here is a specific example episode with a specific stance including two senses:

```
definition ep :: episode where
  "ep == ((PP, Begins, II), {caution, confidence})"
```

It can now be demonstrated how to prove obvious lemmas about the constants:

```
lemma "fst ep = (PP, Begins, II)" by (simp add: ep_def)
lemma "snd ep = {caution, confidence}" by (simp add: ep_def)
lemma "snd ep = {confidence, caution}" by (auto simp add: ep_def)
```

The first lemma states what the first element of example episode *ep* is; its Isabelle proofscript consists of a single method (on the same line as the lemma statement) which invokes the simplifier (via *simp*) with an appeal to the definition of the episode, automatically stored as *ep_def*. The second lemma is homologous for the second component. The third lemma is equivalent to the second one, but also confirms that because stances are defined as sets, the security senses they feature can be indicated in any order. It can be observed that this simple form of reasoning based on set theory cannot be handled by the simplifier alone, but necessitates some classical reasoning (via *auto*).

A socio-technical protocol is defined coherently with what was seen above (§4) — each path in the ceremony in Figure 3 is of this type:

```
types protocol = "episode list"
```

Finally, the constant *hci* can be declared:

```
inductive_set hci :: "protocol set"
```

This constant, which formalises the full layer III, is defined by structural induction on the length of a protocol. Its definition is omitted due to space limits.

5.2 Formal Verification of the Example Ceremony

This Section outlines the main guarantees we have proved about our model of layer III of the example ceremony. It can be seen how existential or universal quantifiers are used to strengthen the conclusions. Full proof scripts are omitted.

A first relevant statement is a theorem stating that a persona always aborts with a sense of unconfidence:

theorem *P_aborts_unconfident*:
 "[((P,Aborts,I), sigma) ∈ set stp; stp ∈ hci]
 ⇒ unconfidence ∈ sigma"

More formally, theorem *P_aborts_unconfident* states that, given a generic protocol *stp* of the model, featuring an episode whose action sees a generic agent abort with a generic provider, and whose stance is generic, then that stance can be proved to feature *unconfidence*.

Another theorem states when the sense of unconfidence is in a stance:

theorem *unconfident_P_when*:
 "[(alpha, sigma) ∈ set stp; unconfidence ∈ sigma; stp ∈ hci]
 ⇒ ∃ P I. alpha = (P,Aborts,I) ∨
 alpha = (P,Queries,I) ∨
 alpha = (I,Compels,P)"

Precisely, theorem *unconfident_P_when* insists on a generic protocol featuring an episode that is generic except for the fact that its stance includes *unconfidence*. The theorem concludes that, for some specific persona *P* and provider *I*, the action in the precondition could only be of three forms.

A subsidiary lemma is useful to prove subsequent theorems:

lemma *I_Ca_Co_explains*:
 "[((I, gamma, P), sigma) ∈ set stp; caution ∈ sigma; confidence ∈ sigma;
 stp ∈ hci]
 ⇒ gamma = Explains"

Lemmas often have their own significance. This says that whenever a generic episode appears but its stance features *caution* with *confidence*, then the cue that is involved must be *Explains*.

An important theorem states the circumstances for registration to take place:

theorem *P_registers_confident_when*:
 "[((P,Registers,I), {confidence}) ∈ set stp; stp ∈ hci]
 ⇒ ∃ sigma. ((I,Explains,P), sigma) ∈ set stp ∧
 caution ∈ sigma ∧ confidence ∈ sigma"

More formally, theorem *P_registers_confident_when* insists on a protocol featuring a specific registration episode. It then concludes that a corresponding explanation episode must occur on the same protocol. Further, it specifies the stance of the latter episode, which must include *caution* with *confidence*.

A final theorem specifies the stance of a registration episode appearing in a protocol where no corresponding explanation episode occurs:

theorem *P_registers_without_confidence*:
 "[((P,Registers,I), sigma) ∈ set stp;
 ∀ sigma'. ((I,Explains,P), sigma') ∉ set stp; stp ∈ hci] ⇒
 confidence ∉ sigma"

Precisely, theorem *P_registers_without_confidence* assumes a protocol with a registration episode featuring a stance *sigma*. Also, it assumes that no explanation episode, for no possible stance *sigma'*, ever occurs on that protocol. Upon these preconditions, it concludes that *sigma* cannot feature the *confidence* sense.

6 Conclusions

The contribution of this paper is threefold: the definition of a detailed model for security ceremonies that relates the technical and the social layers; a graphical representation of the complicated layer that features user personas who interact with human-computer interfaces; the formal analysis of this layer.

There are several benefits of using formal methods at the persona layer. By creating a formal model linking human practices with mental and emotional models that users have about security, it becomes easier to conceptualise of on-line services that respond to particular personas and how this might take place. For example, the modelling of when explanations are required or not required is a valuable tool for the design of on-line services because it provides indicators as to when service providers need and need not intervene with privacy information in the on-line process. In consequence, certain aborts that were often invisible to analysts from the technological, rather than human, perspective may become visible. Aborts are an important aspect of any ceremony because they provide valuable input as to engagement with a particular security or privacy technology.

Formal methods also provide insights that may help detect potential weaknesses in the social and practice theories relating to security and privacy practices. For example, potential ceremony paths that have not surfaced in the field research but that are possible in the formal model can be clearly identified. This can draw field research into considering why certain ceremony paths have not emerged in the grounded research and theorise about why this is so, thus maturing the social and practice theories by identifying the all-important outliers. The formal reasoning also provides a valuable critique to the technology design, pinpointing where the technology design is insufficient for particular personas or where superfluous functionality is provided. Finally, formal analysis is powerful because it is enmeshed in practice-led research. This means that as the fieldwork findings refine, so too does the formal model. Therefore, both model and fieldwork interoperate to refine both the ceremony descriptions and their analysis. The next challenge ahead is to tackle the other layers of our full ceremony model.

Acknowledgements. This work was supported by the Technology Strategy Board; the Engineering and Physical Sciences Research Council and the Economic and Social Research Council [grant number EP/G00255/X]. We are grateful to Gabriele Lenzini and Paul Curzon for useful discussions, and to Elahe Kani-Zabihi and Yee-Lin Lai for working in the team that conducted the initial field research used as the basis for this work.

References

1. URL: Microsoft warning over browser security flaw (2011), <http://www.bbc.co.uk/news/technology-12325139>
2. Ellison, C.: Ceremony design and analysis. Technical report, Cryptology ePrint Archive, Report 2007/739 (2007)
3. Pieters, W.: Representing Humans in System Security Models: An Actor-Network Approach. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications* 2, 75–92 (2011)
4. Martina, J.E., de Souza, T.C.S., Custódio, R.F.: Ceremonies design for PKI's hardware security modules. In: Proc. of the 9th Brazilian Symposium on Information and Computer System Security (NDSS 2009), pp. 115–128. SBC Press (2009)
5. Karlof, C., Tygar, J.D., Wagner, D.: Conditioned-safe ceremonies and a user study of an application to web authentication. In: Proc. of the 16th Network and Distributed System Security Symposium (NDSS 2009) (2009)
6. Cooper, A.: *The Inmates Are Running the Asylum*. SAMS Publishing (2004)
7. URL: Primelife project (2008), <https://www.primelife.eu>
8. Radke, K., Boyd, C., Gonzalez Nieto, J., Brereton, M.: Ceremony Analysis: Strengths and Weaknesses. In: Camenisch, J., Fischer-Hübner, S., Murayama, Y., Portmann, A., Rieder, C. (eds.) SEC 2011. IFIP AICT, vol. 354, pp. 104–115. Springer, Heidelberg (2011)
9. URL: Forcing functions (2011), http://www.interaction-design.org/encyclopedia/forcing_functions.html
10. Solove, D.J.: *Understanding Privacy*. Harvard University Press (2008)
11. Kumaraguru, P., Cranor, L.F.: Privacy indexes: A survey of westin's studies. Technical report, SCS Technical Report Collection (2005)
12. Bella, G.: Formal Correctness of Security Protocols. In: *Information Security and Cryptography*. Springer (2007)
13. Whitworth, B.: The Social Requirements of Technical Systems. In: *Socio-technical Design and Social Networking Systems*, pp. 3–22. IGI Global (2009)
14. Jr., J.M.R.: *Social Protocols* (1998), <http://www.w3.org/Talks/980922-MIT6805/SocialProtocols.html>
15. Bella, G., Coles-Kemp, L.: Seeing the Full Picture: the Case for Extending Security Ceremony Analysis. In: *Proceedings of the 9th Australian Information Security Management Conference*. Secau Security Research Centre Press (2011)
16. URL: Privacy on the internet: Attitudes and behaviours (2010), <http://www.vome.org.uk/wp-content/uploads/2010/03/VOME-exploratorium-survey-summary-results.pdf>
17. Hubbard, T., Ampofo, L.: What's out there? an evaluation of online identity management. Technical report (2010)
18. Buchanan, T., Reips, U.D., Paine, C., Joinson, A.N.: Development of measures of on-line privacy concern and protection for use on the internet. *Journal of the American Society for Information Science and Technology* 58, 157–165 (2007)
19. Norberg, P.A., Horne, D.R., Horne, D.A.: The privacy paradox: Personal information disclosure intentions versus behaviors. *Journal of Consumer Affairs* 41, 100–126 (2007)
20. Paine, C., Reips, U.D., Stieger, S., Joinson, A., Buchanan, T.: Internet users' perceptions of 'privacy concerns' and 'privacy actions'. *International Journal of Human-Computer Studies* 65, 526–536 (2007)

21. Coles-Kemp, L., Kani-Zabihi, E.: On-line privacy and consent: A dialogue, not a monologue. In: Proc. of the New Security Paradigms Workshop (NSPW 2010). ACM Press (2010)
22. Coles-Kemp, L., Kani-Zabihi, E.: Practice Makes Perfect: Motivating Confident On-line Privacy Protection Practices. In: Proceedings of the 3rd IEEE International Conference on Social Computing (SocialCom 2011). IEEE (2011)
23. Coles-Kemp, L., Kani-Zabihi, E.: Service users' requirements for tools to support effective on-line privacy and consent practices. In: Proc. of the 15th Nordic Conference in Secure IT Systems (NordSec 2010). LNCS, pp. 106–120. ACM Press (2010)
24. McMillan, K.: Symbolic Model Checking. Kluwer Academic Publisher (1993)
25. Nipkow, T., Paulson, L.C., Wenzel, M.: Isabelle/HOL: A Proof Assistant for Higher-Order Logic. LNCS, vol. 2283. Springer, Heidelberg (2002)
26. URL: Isabelle download page (2011),
<http://www.cl.cam.ac.uk/Research/HVG/Isabelle/download.html>

A A Primer on the Theorem Prover Isabelle

Isabelle is a generic, interactive theorem prover [26]. *Generic* means that it can reason in a variety of formal systems. This paper refers to Isabelle/HOL [25], which supports the formal language *higher-order logic*, a typed formalism that allows quantification over functions, predicates and sets, but has no temporal operators. *Interactive* means that it is not entirely automatic and, rather, requires a good amount of human intervention. But Isabelle also provides much automation. Its *simplifier*, which can be invoked by the proof method *simp*, combines rewriting with arithmetic decision procedures. Its *automatic provers* can solve most simple proof scenarios. For example, the proof method *blast* implements a fast classical reasoner, and *auto* combines that with the simplifier.

Most proofs are conducted interactively. In a typical proof, the user directs Isabelle to perform a certain induction and then to simplify the resulting proof state, which may contain a number of subgoals. Each subgoal can be given to an automatic prover or be reduced to other subgoals by the use of some lemma. Failure to find a proof for a conjecture may simply mean that the user is not skilled enough; otherwise, it may exhibit what in the modelled system contradicts the conjecture and hence help in locating a system bug or an erroneous human intuition. The series of commands used to prove a theorem can be seen as a proof sketch. Confidence that the proof is sound comes from inspecting the line of reasoning adopted and the lemmas it requires.

The typical Isabelle proof methods mentioned above, *simp* for the conditional term rewriter and *blast* for the classical reasoner, are combined by the proof method *force*, which only applies to the first subgoal in the proof state, and by *auto*, which applies to the entire state. All these commands allow the installation of additional lemmas to appeal to. A very useful method is *clarify*, performing obvious steps, while *clarsimp* interleaves it with the simplifier. A variant of *clarify* is *safe*, which additionally performs the obvious steps that split up the subgoal.

A Small Depth-16 Circuit for the AES S-Box

Joan Boyar^{1,*} and René Peralta²

¹ Department of Mathematics and Computer Science
University of Southern Denmark

joan@imada.sdu.dk

² Information Technology Laboratory, NIST
rene.peralta@nist.gov

Abstract. New techniques for reducing the depth of circuits for cryptographic applications are described. These techniques also keep the number of gates quite small. The result, when applied to the AES S-Box, is a circuit with depth 16 and only 128 gates. For the inverse, it is also depth 16 and has only 127 gates. There is a shared middle part, common to both the S-Box and its inverse, consisting of 63 gates. The best previous comparable design for the AES S-Box has depth 22 and size 148 [12].

1 Introduction

Constructing optimal combinational circuits is an intractable problem under almost any meaningful metric (gate count, depth, energy consumption, etc.). In practice, no known techniques can reliably find optimal circuits for functions with as few as eight Boolean inputs and one Boolean output (there are 2^{256} such functions). Thus, heuristic or specialized techniques are necessary in practice. The depth of a circuit is the number of gates on a longest path. Reducing depth generally leads to faster circuits. Reducing the number of gates is important for reducing area and power consumption. The aim is to reduce both simultaneously, but there is often a trade-off between these two goals.

Many different logically complete bases are possible for circuits. Since the operations in the basis (XOR, AND) are equivalent to addition and multiplication modulo 2 (i.e., in $GF(2)$), much work on circuits for cryptographic functions uses this basis. For logical completeness, the negation operation (or the constant 1) is needed as well. In $GF(2)$, negation corresponds to $x + 1$. For technical reasons, and for an accurate gate-count, we use XNOR gates ($XNOR(x, y) = x + y + 1$ in $GF(2)$) instead of negation. This platform-independent basis leads to easy comparison with previous implementations. Note that changing AND gates to NANDs is not difficult.

The Advanced Encryption Standard (AES) is a widely used symmetric key encryption system that was adopted as a standard by the U.S. government in 2002. The nonlinear part of AES is the SubBytes step, which is often referred

* Partially supported by the Danish Council for Independent Research, Natural Sciences.

to as the S-Box. The original specification of AES [11] defined the S-Box as the multiplicative inverse in the field $GF(2^8)$, followed by an affine transformation. However, the S-Box can also be implemented using table look-up. This requires a ROM with 2048 bits. However, ROMs are not efficient in terms of size/power consumption. Satoh et al. [15], using a *tower-of-fields* approach to compute the S-Box, designed a combinational circuit that can be implemented with an area under 1/4 of previous ROM-based designs. Other work, for example [3,10,1], has improved on this size.

Recently, Nogami et al. [12] presented a technique for reducing circuit depth in the forward direction of the AES S-Box. Their technique was primarily to choose mixed bases for the tower-of-fields architecture in such a way that the 8×8 matrices doing the linear transformations at the top and bottom had at most 4 ones in every row. In this way they were able to compute these transformations in depth 2 each, for a total of depth 4. Their technique cannot be simultaneously applied to the AES circuit in the reverse direction, as the inverse matrices do not have at most 4 ones in every row.

We present more general techniques that are less dependent on the actual representation of the fields. These techniques allow us to produce circuits which are both smaller and shorter than those in [12]. Although the size of our construction is larger than our size-optimized circuit [1] (which had only 115 gates but had depth 28) it is comparable to previous efforts to make a small circuit (the constructions in [3] and [10] have depths 25 and 27, respectively). The work of Nogami et al. improved on the forward direction of the S-Box to depth 22 at the cost of increasing the number of gates to 148. Our new circuits have depth 16 in both directions, size 128 in the forward direction, and size 127 in the reverse direction. The part that is shared between the forward and reverse directions is of size 63.

A general overview of the technique is described in Section 2, and the technique used for the nonlinear component, inversion in $GF(2^4)$, is described in Section 3. A greedy heuristic for linear components is described in Section 4. Further techniques for depth and size reduction are described in Section 5. The AES S-Box circuits are presented in Section 6 and conclusions in Section 7.

2 Combinational Circuit Optimization

Under the basis (XOR,XNOR,AND), classic results by Shannon [16] and Lupanov [9] show that almost all predicates on n bits have circuit complexity about $\frac{2^n}{n}$. The *multiplicative complexity* of a function is the number of AND gates necessary and sufficient to compute the function. Analogous to the Shannon-Lupanov bound, it was shown in [2] that almost all Boolean predicates on n bits have multiplicative complexity about $2^{\frac{n}{2}}$. Strictly speaking, these theorems say nothing about the class of functions with polynomial circuit complexity. However, it is reasonable to expect that, in practice, the multiplicative complexity of functions is significantly smaller than their Boolean complexity. This is one of the principles that guide our design strategy.

Circuits with few AND gates will naturally have large sections which are purely linear. The authors [1] and Courtois et al. [5] have used this insight to construct circuits much smaller than previously known for a variety of applications. The heuristic is a two-step process which first reduces multiplicative complexity and then optimizes linear components.

The work presented in this paper is based on the idea that a short circuit may be obtained by starting from a small (i.e. optimized for size) circuit and performing three types of optimizations with automatic heuristics:

- use techniques from automatic theorem proving to re-synthesize non-linear components into lower-depth constructions;
- apply a greedy heuristic to re-synthesize linear components into lower-depth constructions;
- perform depth-shortening and size-reducing local replacement.

These techniques are explained below. They will often increase the size of the circuit, but we start with our small circuit from [1], and the techniques are designed to minimize the size increase.

3 The Tower Field Construction: A Nonlinear Component

There are many representations of $GF(2^8)$. We construct

- $GF(2^2)$ by adjoining a root W of $x^2 + x + 1$ over $GF(2)$;
- $GF(2^4)$ by adjoining a root Z of $x^2 + x + W^2$ over $GF(2^2)$.
- $GF(2^8)$ by adjoining a root Y of $x^2 + x + WZ$ over $GF(2^4)$.

Thus, there is a tower of fields, each one contained in the previous, and one can do multiplication and inverse operations in the larger fields efficiently by implementing the relevant operations in the smaller fields efficiently, as in [8].

As does Canright in [3], we represent $GF(2^2)$ using the basis (W, W^2) , $GF(2^4)$ using the basis (Z^2, Z^8) , and $GF(2^8)$ using the basis (Y, Y^{16}) .

Let $A = a_0Y + a_1Y^{16}$ be an arbitrary element in $GF(2^8)$. Following [8], the inverse of A can be computed as follows:

$$\begin{aligned}
 A^{-1} &= (AA^{16})^{-1}A^{16} = ((a_0Y + a_1Y^{16})(a_1Y + a_0Y^{16}))^{-1}(a_1Y + a_0Y^{16}) \\
 &= ((a_0^2 + a_1^2)Y^{17} + a_0a_1(Y^2 + Y^{32}))^{-1}(a_1Y + a_0Y^{16}) \\
 &= ((a_0 + a_1)^2Y^{17} + a_0a_1(Y + Y^{16})^2)^{-1}(a_1Y + a_0Y^{16}) \\
 &= ((a_0 + a_1)^2WZ + a_0a_1)^{-1}(a_1Y + a_0Y^{16}).
 \end{aligned}$$

Thus, computation of the inverse in $GF(2^8)$ can be done using operations in $GF(2^4)$ as follows:

$$\begin{aligned}
 T_1 &= (a_0 + a_1); & T_2 &= (WZ)T_1^2; & T_3 &= a_0a_1; & T_4 &= T_2 + T_3; \\
 T_5 &= T_4^{-1}; & T_6 &= T_5a_1; & T_7 &= T_5a_0;
 \end{aligned}$$

The result is $A^{-1} = T_6Y + T_7Y^{16}$.

The $GF(2^4)$ operations involved are addition, multiplication, square and scale by WZ, and inverse. Of these, only multiplication and inverse turn out to be non-linear. We derive a standard $GF(2^4)$ multiplication circuit by reduction to $GF(2^2)$ operations. The standard inversion circuit, however, has more gates and depth than necessary. Hence we derive a better circuit here.

Let $\Delta = (x_0W + x_1W^2)Z^2 + (x_2W + x_3W^2)Z^8$ be an arbitrary element in $GF(2^4)$. It is not hard to verify that its inverse is $\Delta^{-1} = (y_0W + y_1W^2)Z^2 + (y_2W + y_3W^2)Z^8$ where the y_i 's satisfy the following polynomials over $GF(2)$ (see [1]):

- $y_0 = x_1x_2x_3 + x_0x_2 + x_1x_2 + x_2 + x_3$
- $y_1 = x_0x_2x_3 + x_0x_2 + x_1x_2 + x_1x_3 + x_3$
- $y_2 = x_0x_1x_3 + x_0x_2 + x_0x_3 + x_0 + x_1$
- $y_4 = x_0x_1x_2 + x_0x_2 + x_0x_3 + x_1x_3 + x_1$

The heuristic we used in [1] to compute the y_i 's was inspired by methods from automatic theorem proving. Consider an arbitrary predicate f on n inputs. We refer to the last column of the truth table for f as the *signal* of f . The columns in the truth table corresponding to each of the inputs to f are *known* signals. A search for a circuit for f starts with this set S of known signals. If u, v are known signals for functions g, h respectively, then the bit-wise XOR (AND) of u and v is the signal for the predicate $g \oplus h$ ($g \wedge h$). We can *grow* the set S by adding the XOR of randomly chosen signals. We call this step an *XOR round*. The analogous step where the AND of signals is added to S is called an *AND round*. Each round is parametrized by the number of new signals added and the maximum number of AND gates allowed. In either an XOR round or an AND round, two signals are not combined if doing so creates a circuit with more AND gates than is allowed. The heuristic alternates between XOR and AND rounds until the target signal is found or the set S becomes too large. In the latter case, since this is a randomized procedure, we start again.

In [1] we used this heuristic to find a circuit with only 5 AND gates and 11 XOR gates, but depth 9. In terms of size, this was a significant improvement over previous constructions (e.g. [13,4]). All these constructions, however, were mostly concerned with size. To minimize depth without incurring a large increase in size, we use the search techniques described above, but add code to discard search paths that either violate given depth constraints or size constraints. With this modification, we found a circuit with depth 4 and size 17. The straight-line program for the circuit is in Figure 1 (arithmetic is over $GF(2)$).

4 A Greedy Heuristic for Linear Components

After this small circuit for inversion in $GF(2^4)$ was found, it was set into the original circuit for AES (we started with the circuit in [1]), and maximal linear components, connected components of the circuit not containing any AND gates, were found. The largest linear components in our circuit are the top linear and bottom linear components. These components contain more than the

$t_1 = x_2 + x_3$	$t_2 = x_2 \times x_0$	$t_3 = x_1 + t_2$
$t_4 = x_0 + x_1$	$t_5 = x_3 + t_2$	$t_6 = t_5 \times t_4$
$t_7 = t_3 \times t_1$	$t_8 = x_0 \times x_3$	$t_9 = t_4 \times t_8$
$t_{10} = t_4 + t_9$	$t_{11} = x_1 \times x_2$	$t_{12} = t_1 \times t_{11}$
$t_{13} = t_1 + t_{12}$	$y_0 = t_2 + t_{13}$	$y_1 = x_3 + t_7$
$y_2 = t_2 + t_{10}$	$y_3 = x_1 + t_6$	

Fig. 1. Inversion in $GF(2^4)$. Input is (x_0, x_1, x_2, x_3) and output is (y_0, y_1, y_2, y_3) .

$$U^T = \begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Fig. 2. The transpose U^T of the top linear transformation U

linear operations defined explicitly in the definition of the AES S-Box and the matrices to do the basis changes. This is because they include some of the finite field inversion operations. The top linear component is defined by the matrix U , a 22×8 matrix (Figure 2), meaning that 22 linear combinations of the 8 inputs are calculated. One can compute all 22 of the required outputs with only 23 XOR gates, and 23 are necessary [16,7]. But these results do not attempt to minimize depth (the depth is 7). Since there are only 8 columns in this matrix, each of the 22 outputs could clearly be calculated independently using depth at most 3, simply by using a balanced binary tree with the inputs as leaves. The challenge is to achieve the low depth without increasing the number of XOR gates drastically. The algorithm below does this. (Note that although the linear transformation at the top of Nogami et al.’s circuit only has depth 2, they have XOR gates at depth 3, so their top linear component also has depth at least 3.)

The bottom linear component is defined by the matrix B , an 8×18 matrix (Figure 3). The row with the largest Hamming weight (number of ones = number of variables added together) has 12 ones, so depth at most 4 is possible for this component.

The smallest circuits for these two matrices, U and B , use the concept of cancellation of variables. Note that in [1], the variable y_{11} is computed as $y_{20} \oplus y_9$. Since $y_{20} = x_0 \oplus x_1 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6$ and $y_9 = x_0 \oplus x_3$, the result is $y_{11} = x_1 \oplus x_4 \oplus x_5 \oplus x_6$; the x_0 and x_3 are cancelled.

When attempting to find small, low-depth circuits for a linear component, one expects that cancellation of variables will be of limited help, since it would often require that something with a large Hamming weight has already been computed, before adding one to the depth at the gate where the cancellation

$$B = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}$$

Fig. 3. The bottom linear transformation B

occurs. Thus, it seems reasonable to start with a technique which does not allow cancellation, and then try to add cancellation afterwards where it helps.

We modify Paar’s technique [14], a greedy approach which produces cancellation-free programs. Paar’s technique keeps a list of variables computed, which is initially only the inputs. Then it repeatedly determines which two variables, XORed together, occur in most outputs. One such pair is selected and XORed together. This result is added as a new variable which appears in all outputs where both variables previously appeared. This is repeated until everything has been computed. Paar’s technique is implemented by starting with the initial matrix and adding columns corresponding to the new variables which are added. When a new column is added, this corresponds to adding two existing variables, u and v . In all rows in the matrix which currently have a one in both of the columns corresponding to u and v , those two ones are changed to zeros, and a one is placed in the corresponding row of the new column. All other values in the new column are set to zero.

The `Low_Depth_Greedy` algorithm maintains the greedy approach of Paar’s technique, but only allows this greediness as long as it does not increase the circuit’s depth unnecessarily. Assume that k is the depth we are aiming for, i.e. $k = \lceil \log_2(w) \rceil$, where w is the largest Hamming weight of any row. This new algorithm has k phases, starting with 0. At the beginning of a new phase, we check if any row has Hamming weight two. Since there must be an additional gate to produce that output, we produce it at the beginning of the phase so that it affects all counting in the current phase. During phase $i \geq 0$, no row in the current matrix has Hamming weight more than 2^{k-i} and only inputs or gates already produced at depth i or less are considered as possible inputs to gates in phase i . Thus, the depth of gates in phase i is at most $i + 1$. When choosing two possible inputs for gates, one chooses a pair which occurs most frequently in the current rows, with the restriction, of course, that both inputs are at level i or less. Pseudo-code for this algorithm is given in Figure 4. This algorithm produces a minimum depth (optimal depth) circuit.

Theorem 1. *When given an $m \times n$ 0-1 matrix, M , with maximum Hamming weight at most 2^k in any row, Algorithm `Low_Depth_Greedy`, produces a correct, depth- k circuit for computing the linear component defined by the matrix. The*

```

Low_Depth_Greedy( $M, m, n, k$ ):
{  $M$  is an  $m \times n$  0-1 matrix with Hamming weight at most  $2^k$  in any row }
   $s := n + 1$  { index of the next column }
  for  $i := 0$  to  $k - 1$  do
     $ip := s - 1$  { keep track of which gates had depth at most  $i$  }
    while there is some row in  $M$  with Hamming weight  $> 2^{k-i-1}$  do
      { Phase  $i$  }
      if some row  $\ell$  in  $M$  with weight 2
        had weight 2 at the beginning of the phase
        then let  $j_1$  and  $j_2$  be the columns in row  $\ell$  with ones
        else find two columns  $1 \leq j_1, j_2 \leq ip$ 
          which maximize  $|\{\ell \mid M[\ell, j_1] = M[\ell, j_2] = 1\}|$ 
        add an XOR gate with inputs from the variables for columns  $j_1, j_2$ 
        the output variable produced will correspond to column  $s$ 
      for  $\ell = 1$  to  $m$  do
        if  $M[\ell, j_1] = M[\ell, j_2] = 1$ 
          then  $M[\ell, j_1] := 0; M[\ell, j_2] := 0; M[\ell, s] := 1$ 
          else  $M[\ell, s] := 0$ 
       $s := s + 1$ 

```

Fig. 4. Algorithm for creating a minimum depth circuit for linear components

running time is $O(mt^3)$, where t is the final number of columns and is at most $mn + n - m$.

Proof. If one considers the inputs as being produced at depth zero, in phase i of the algorithm, only variables which have been produced at depth at most i are considered as possible inputs to XOR gates, so the XOR gates produced have depth at most $i + 1$. This is maintained inductively by only considering columns between 1 and ip , and ip is reset at the end of each phase to the last column produced in that phase. Since the algorithm maintains that at the beginning of phase i , no more than 2^{k-i} of the current variables have to be XORed to produce any output, the algorithm terminates in phase $k - 1$, giving maximum depth k . Note that it will always be possible to proceed from phase i to phase $i + 1$, since combining the at most 2^{k-i} ones any row by pairs will reduce the number of ones to at most half as many, at most 2^{k-i-1} .

For each XOR gate added, the algorithm checks every pair of columns between 1 and $ip < s$, where s is the new column being added. For each of these pairs of columns, one checks for each row if both entries corresponding to these columns are one and then does some updating. The number of rows is m , so the total running time is $O(mt^3)$. Since there are at most n ones in every row, each row will be computed using at most $n - 1$ XORs, and all m rows will be computed with at most $m(n - 1)$ XORs. There are n columns initially, so in all $t \leq mn + n - m$. \square

Another possibility for an algorithm to produce optimal depth circuits for linear components would have been to finish with all pairs of inputs before continuing

to pairs involving gates at depth one, and then to finish with all pairs at depth one (or involving the possibly one remaining input which has not been paired), etc. However, the method chosen here allows more flexibility in choosing gates, thus allowing more possibilities to create gates which can be used more than once.

5 Local Optimizations

After applying the techniques discussed in the previous sections, highly localized optimizations may be possible.

We are able to further decrease the number of gates in the top linear component since not all the XOR gates at level three (an output of the top linear component) would necessarily increase the total depth if they were at level four or more (for the AES S-Box, k and $k+1$ more generally). Or, on the other hand, one might be able to reduce the depth even more by calculating some outputs of the top linear component at lower depth than the depth indicated by the matrix row with largest Hamming weight, if these “outputs” are on a longest path.

It is easy to determine which outputs of the top linear component could be allowed to be at a larger depth or should be at a lower depth if possible, using a program which calculates the depth and height of every gate. If all of the outputs of the top linear component which have depth and height values adding up to exactly the total depth of the circuit are such that they could have been calculated at lower depth than their current depth, then one can probably reduce the depth of the circuit. On the other hand, when these values add up to less than the total depth of the circuit, there is some *slack* at that gate. For XOR gates at depth 3 (in an AES S-Box circuit) which have slack, one can check if they are the sum of any two of the other outputs of the top-linear part. If they are, these other outputs were computed at depth 3, so adding them together only gives depth 4, which is acceptable when the output was originally created at a gate with slack. Note that cancellation of variables should be allowed here.

The `Low_Depth_Greedy` algorithm can be modified to take advantage of slackness. In this case, an extra array *Factor* is initialized for each input to the linear transformation. Rows with no slack are given the value 1, and rows that could be at j levels further down than the minimum are given the value 2^j in *Factor*. Then, when checking if one should proceed to the next phase, rather than check if all rows have Hamming weight at most 2^{k-i} for phase i , one checks if its Hamming weight divided by its value in *Factor* is at most 2^{k-i} . This allows the possibility of choosing inputs required for these outputs at a larger depth. These modifications of the `Low_Depth_Greedy` algorithm were not actually necessary to produce the circuits found.

Finally, there are straight-forward techniques for reducing depth in linear components via local replacement. Consider any gate in such a component. The output produced there is the XOR of several values (either inputs or outputs from other gates). These values can be XORed in any order to get this result. For example, suppose $g = g_1 \oplus g_2$, $g_1 = g_3 \oplus g_4$, and g_1 is at depth d . If the depths

of g_2 and g_3 are at most $d - 2$, then g is at depth $d + 1$ and g_4 is at depth at $d - 1$. Now calculating $h_1 = g_2 \oplus g_3$ and $h_2 = h_1 \oplus g_4$ results in h_2 computing the same result as g , but at depth one lower, d . If the result computed at g_1 was not used anywhere else in the circuit, then this does not increase the total number of gates. However, if g_1 is used elsewhere, it would still need to be computed, and the number of gates would increase by one.

6 The Circuits

The depth-16 circuits are shown in Figures 5, 6, 7, 8, and 9. Note that the addition and multiplication operations are modulo 2, so they are XOR and AND operations. The $\#$ operation is an XNOR (adding modulo 2 and then complementing the result) and is only used for the outputs. We used Algorithm Low_Depth_Greedy for the four linear transformations (here, we do not include the binary matrices corresponding to the transformations in the reverse direction of the AES S-Box). The circuits are divided into three components: top linear transformations (Figures 5 and 6), shared non-linear component (Figure 7), and bottom linear transformations (Figures 8 and 9).

T1 = U0 + U3	T8 = U7 + T6	T15 = T5 + T11	T22 = T7 + T21
T2 = U0 + U5	T9 = U7 + T7	T16 = T5 + T12	T23 = T2 + T22
T3 = U0 + U6	T10 = T6 + T7	T17 = T9 + T16	T24 = T2 + T10
T4 = U3 + U5	T11 = U1 + U5	T18 = U3 + U7	T25 = T20 + T17
T5 = U4 + U6	T12 = U2 + U5	T19 = T7 + T18	T26 = T3 + T16
T6 = T1 + T5	T13 = T3 + T4	T20 = T1 + T19	T27 = T1 + T12
T7 = U1 + U2	T14 = T6 + T11	T21 = U6 + U7	

Fig. 5. Top linear transform in forward direction. Inputs are U0...U7.

The circuits were generated automatically using randomization for tie-resolution. Different runs of our code yield depth 16 consistently. However, size can vary by a few gates. As long as the topology derived from the tower-of-fields method is maintained, we conjecture that it is unlikely that the size of the circuits can be significantly reduced without increasing the depth. We also conjecture that it is

T23 = U0 + U3	T19 = T22 + R5	T17 = U2 # T19	T6 = T22 + R17
T22 = U1 # U3	T9 = U7 # T1	T20 = T24 + R13	T16 = R13 + R19
T2 = U0 # U1	T10 = T2 + T24	T4 = U4 + T8	T27 = T1 + R18
T1 = U3 + U4	T13 = T2 + R5	R17 = U2 # U5	T15 = T10 + T27
T24 = U4 # U7	T3 = T1 + R5	R18 = U5 # U6	T14 = T10 + R18
R5 = U6 + U7	T25 = U2 # T1	R19 = U2 # U4	T26 = T3 + T16
T8 = U1 # T23	R13 = U1 + U6	Y5 = U0 + R17	

Fig. 6. Top linear transform in reverse direction

M1 = T13 x T6	M17 = M5 + T24	M33 = M27 + M25	M49 = M43 x T16
M2 = T23 x T8	M18 = M8 + M7	M34 = M21 x M22	M50 = M38 x T9
M3 = T14 + M1	M19 = M10 + M15	M35 = M24 x M34	M51 = M37 x T17
M4 = T19 x D	M20 = M16 + M13	M36 = M24 + M25	M52 = M42 x T15
M5 = M4 + M1	M21 = M17 + M15	M37 = M21 + M29	M53 = M45 x T27
M6 = T3 x T16	M22 = M18 + M13	M38 = M32 + M33	M54 = M41 x T10
M7 = T22 x T9	M23 = M19 + T25	M39 = M23 + M30	M55 = M44 x T13
M8 = T26 + M6	M24 = M22 + M23	M40 = M35 + M36	M56 = M40 x T23
M9 = T20 x T17	M25 = M22 x M20	M41 = M38 + M40	M57 = M39 x T19
M10 = M9 + M6	M26 = M21 + M25	M42 = M37 + M39	M58 = M43 x T3
M11 = T1 x T15	M27 = M20 + M21	M43 = M37 + M38	M59 = M38 x T22
M12 = T4 x T27	M28 = M23 + M25	M44 = M39 + M40	M60 = M37 x T20
M13 = M12 + M11	M29 = M28 x M27	M45 = M42 + M41	M61 = M42 x T1
M14 = T2 x T10	M30 = M26 x M24	M46 = M44 x T6	M62 = M45 x T4
M15 = M14 + M11	M31 = M20 x M23	M47 = M40 x T8	M63 = M41 x T2
M16 = M3 + M2	M32 = M27 x M31	M48 = M39 x D	

Fig. 7. Shared part of AES S-box circuit. $D = U7$ in the forward direction and $D = Y5$ in the reverse direction.

L0 = M61 + M62	L10 = M53 + L4	L20 = L0 + L1	S0 = L6 + L24
L1 = M50 + M56	L11 = M60 + L2	L21 = L1 + L7	S1 = L16 # L26
L2 = M46 + M48	L12 = M48 + M51	L22 = L3 + L12	S2 = L19 # L28
L3 = M47 + M55	L13 = M50 + L0	L23 = L18 + L2	S3 = L6 + L21
L4 = M54 + M58	L14 = M52 + M61	L24 = L15 + L9	S4 = L20 + L22
L5 = M49 + M61	L15 = M55 + L1	L25 = L6 + L10	S5 = L25 + L29
L6 = M62 + L5	L16 = M56 + L0	L26 = L7 + L9	S6 = L13 # L27
L7 = M46 + L3	L17 = M57 + L1	L27 = L8 + L10	S7 = L6 # L23
L8 = M51 + M59	L18 = M58 + L8	L28 = L11 + L14	
L9 = M52 + M53	L19 = M63 + L4	L29 = L11 + L17	

Fig. 8. Bottom linear transform in forward direction. Outputs are $S0 \dots S7$.

P0 = M52 + M61	P10 = M57 + P4	P20 = P4 + P6	W1 = P26 + P29
P1 = M58 + M59	P11 = P0 + P3	P22 = P2 + P7	W2 = P17 + P28
P2 = M54 + M62	P12 = M46 + M48	P23 = P7 + P8	W3 = P12 + P22
P3 = M47 + M50	P13 = M49 + M51	P24 = P5 + P7	W4 = P23 + P27
P4 = M48 + M56	P14 = M49 + M62	P25 = P6 + P10	W5 = P19 + P24
P5 = M46 + M51	P15 = M54 + M59	P26 = P9 + P11	W6 = P14 + P23
P6 = M49 + M60	P16 = M57 + M61	P27 = P10 + P18	W7 = P9 + P16
P7 = P0 + P1	P17 = M58 + P2	P28 = P11 + P25	
P8 = M50 + M53	P18 = M63 + P5	P29 = P15 + P20	
P9 = M55 + M63	P19 = P2 + P3	W0 = P13 + P22	

Fig. 9. Bottom linear transform in reverse direction. Outputs are $W0 \dots W7$.

unlikely that the depth can be reduced without significantly increasing size. Of course, if the logical base is expanded, we may be able to do better. For example, if NAND gates are used in the circuit for inversion in $GF(2^4)$, it is not hard to reduce the number of gates by two without increasing the depth (see appendix). Since there are only 256 possible inputs, we verified the circuits fully against the specifications in [11].

7 Conclusion

The techniques used in [11,5] to obtain smaller AND/XOR circuits were modified and extended here to consider the depth of the circuits produced. The resulting techniques were successfully applied to the AES S-Box, significantly improving, both in size and depth, over another recent attempt at a low depth S-Box [12].

The techniques presented in this paper appear, not surprisingly, to lead to a trade-off between size and depth. The modification of our search technique [1] to find circuits with few AND gates, which rejected candidates with too large depth, decreased the depth of the $GF(2^4)$ inversion from 9 to 4 while only increasing the number of gates from 16 to 17, changing 5 AND gates and 11 XOR gates to 7 AND gates and 10 XOR gates. Thus, most of the trade-off was due to the techniques for handling large linear components, especially the Low-Depth-Greedy algorithm. To illustrate this trade-off, we list the depths and sizes of some of the circuits for the AES S-Box which were obtained in the course of this research:

- depth 28; size 115: original circuit
- depth 23; size 116
- depth 22; size 117
- depth 21; size 118
- depth 20; size 122
- depth 16; size 128: final result

References

1. Boyar, J., Peralta, R.: A New Combinational Logic Minimization Technique with Applications to Cryptology. In: Festa, P. (ed.) SEA 2010. LNCS, vol. 6049, pp. 178–189. Springer, Heidelberg (2010)
2. Boyar, J., Peralta, R., Pochuev, D.: On the multiplicative complexity of Boolean functions over the basis $(\wedge, \oplus, 1)$. Theoretical Computer Science 235, 43–57 (2000)
3. Canright, D.: A Very Compact S-Box for AES. In: Rao, J.R., Sunar, B. (eds.) CHES 2005. LNCS, vol. 3659, pp. 441–455. Springer, Heidelberg (2005)
4. Canright, D.: A very compact Rijndael S-box. Tech. Rep. NPS-MA-05-001, Naval Postgraduate School (2005)
5. Courtois, N., Hulme, D., Mourouzis, T.: Solving circuit optimisation problems in cryptography and cryptanalysis. IACR Cryptology ePrint Archive 2011, 475 (2011)
6. Fuhs, C., Schneider-Kamp, P.: Synthesizing Shortest Linear Straight-Line Programs over $GF(2)$ Using SAT. In: Strichman, O., Szeider, S. (eds.) SAT 2010. LNCS, vol. 6175, pp. 71–84. Springer, Heidelberg (2010)

7. Fuhs, C., Schneider-Kamp, P.: Optimizing the AES S-Box using SAT. In: Proceedings of the 8th International Workshop on the Implementation of Logics (2010)
8. Itoh, T., Tsujii, S.: A fast algorithm for computing multiplicative inverses in $GF(2^m)$ using normal bases. *Inf. Comput.* 78(3), 171–177 (1988)
9. Lupanov, O.B.: A method of circuit synthesis. *Izvestia V.U.Z. Radiofizika* 1, 120–140 (1958)
10. Morioka, S., Satoh, A.: An Optimized S-Box Circuit Architecture for Low Power AES Design. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 172–186. Springer, Heidelberg (2003)
11. NIST: Advanced Encryption Standard (AES) (FIPS PUB 197). National Institute of Standards and Technology (November 2001)
12. Nogami, Y., Nekado, K., Toyota, T., Hongo, N., Morikawa, Y.: Mixed Bases for Efficient Inversion in $\mathbb{F}(((2^2)^2)^2)$ and Conversion Matrices of Subbytes of AES. In: Mangard, S., Standaert, F.-X. (eds.) CHES 2010. LNCS, vol. 6225, pp. 234–247. Springer, Heidelberg (2010)
13. Paar, C.: Some remarks on efficient inversion in finite fields. In: 1995 IEEE International Symposium on Information Theory, p. 58 (1995)
14. Paar, C.: Optimized arithmetic for Reed-Solomon encoders. In: 1997 IEEE International Symposium on Information Theory, p. 250 (1997)
15. Satoh, A., Morioka, S., Takano, K., Munetoh, S.: A Compact Rijndael Hardware Architecture with S-Box Optimization. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 239–254. Springer, Heidelberg (2001)
16. Shannon, C.E.: The synthesis of two-terminal switching circuits. *Bell System Tech. J.* 28, 59–98 (1949)

A Appendix

$t_1 = x_2 + x_3$	$t_2 = x_2 \times x_0$	$t_3 = x_1 + t_2$
$t_4 = x_0 + x_1$	$t_5 = x_3 + t_2$	$t_6 = t_5 \times t_4$
$t_7 = t_3 \times t_1$	$t_8 = x_0 \text{ NAND } x_3$	$t_9 = t_4 \times t_8$
$t_{10} = x_2 \text{ NAND } x_1$	$t_{11} = t_1 \times t_{10}$	$y_0 = t_2 + t_{11}$
$y_1 = x_3 + t_7$	$y_2 = t_2 + t_9$	$y_3 = x_1 + t_6$

Fig. 10. Inversion in $GF(2^4)$ using NAND gates. Input is (x_0, x_1, x_2, x_3) and output is (y_0, y_1, y_2, y_3) .

Formal Verification of the mERA-Based eServices with Trusted Third Party Protocol

Maria Christofi^{1,2} and Aline Gouget¹

¹ Gemalto - 6, rue de la Verrerie - 92447 Meudon sur Seine - France

² Versailles Saint-Quentin-en-Yvelines University - France
{maria.christofi,aline.gouget}@gemalto.com

Abstract. Internet services such as online banking, social networking and other web services require identification and authentication means. The European Citizen card can be used to provide a privacy-preserving authentication for Internet services enabling e.g. an anonymous age verification or other forms of anonymous attribute verification. The Modular Enhanced Symmetric Role Authentication (mERA) - based eServices with trusted third party protocol is a privacy-preserving protocol based on eID card recently standardized at CEN TC224 WG16. In this paper, we provide a formal analysis of its security by verifying formally several properties, such as secrecy, message authentication, unlinkability, as well as its liveness property. In the course of this verification, we obtain positive results about this protocol. We implement this verification with the ProVerif tool.

Keywords: privacy, authentication, mERA, formal verification, cryptographic protocol, ProVerif.

1 Introduction

Cryptographic protocols can be seen as small programs which are designed to guarantee the security of exchanges between participants. Even if some protocols look very simple, the design of a secure cryptographic protocol is often difficult and error-prone. This is the case for many protocols, for which attacks were found many years after their design. It is thus very important to formally analyze the properties fulfilled by these protocols.

A protocol verification procedure can be seen as a procedure with two inputs and one output. The procedure takes as inputs a protocol definition and a statement describing a property of protocol execution and then outputs a “yes” or “not sure” answer. A “yes” answer means that the input property is indeed a property of the input protocol. A “not sure” answer means that some effort to establish this proposition has failed. This means that either the proposition is false and so cannot be established, or it may be true but more efforts are needed in order to establish it.

The verification of security properties of protocols has been the subject of many recent research works. The general idea is to give a description of the

protocol in the formal style of Dolev-Yao model [8], a description of security properties such as secrecy and message authentication. Then verifiers, such as ProVerif [7] or AVISPA [5], can be used to nearly automatically check various security properties of protocols.

This paper describes the use of one of these tools (mainly ProVerif) for the verification of the mERA-based eServices with trusted third party protocol. This is a recently standardized protocol (EN 14890) presented in [12]. This protocol is used when a user wants to have access to some services where he has to prove that he fulfills some criteria. The aim is that the user preserves his privacy concerning his identity or personal data while being able to prove the fulfillment of these criteria.

The verification yields assurance about the secrecy, the message authentication and the unlinkability during the protocol. The reasoning could also be used in other related protocols.

Structure of the Paper. In this paper, we review in Section 2 the description of the mERA-based eServices with trusted third party protocol. In Section 3, we briefly present the verification tool. We explain our formal specification in Section 4 and then prove the security properties in Section 5.

Related Work. It is quite common to reason informally about security protocols. For instance, in [9] Krawczyk define the protocol SKEME and he gave some informal arguments about the properties SKEME. Generally, such arguments are informative, but far from being complete and fully reliable. Formal proofs could be necessary according to the critical impacts of security flows. Nevertheless, formal proofs for concrete, practical protocols remain relatively rare. We state some of these results. The work of Abadi and Blanchet in [11] reports on the formal specification of a non trivial protocol for certified email, as well as on the verification of its main security properties. In [2], Abadi, Blanchet and Fournet combine manual and automated proofs for analyzing the Just Fast Keying (JFK) protocol which was one of the candidates to replace IKE as the key exchange protocol in IPSec. They verify classical properties such as secrecy and authenticity, but also properties like forward secrecy and identity protection. Both [10] and [6] formalize and analyze privacy properties for electronic voting. Indeed, in [10] Kremer and Ryan provide the first definitions in the symbolic model of vote-privacy, receipt-freeness and coercion-resistance. However, this work does not consider forced-abstention attacks and do not apply to remote voting protocols. These two last properties have been studied by Backes et al. in [6].

In this paper, we use ProVerif to formally verify some properties of the privacy-preserving authentication protocol called mERA-based eServices with trusted third party protocol.

2 mERA-Based eServices with Trusted Third Party Protocol

This section recalls the description of the mERA-based eServices with trusted third party protocol (mERA stands for “modular Enhanced Symmetric Role

Authentication”). The reader can also see the original description in [4] for additional details.

The participants of this protocol are: a smart card (denoted by ICC for Integrated Circuit Card) with its owner and a local reader, an identity provider (denoted by IdP), a service provider (denoted by SP) and a certification authority (denoted by CA).

The aim of the mERA-based eServices with trusted third party protocol is that a user could remotely use a privacy-preserving credential without disclosing his identity or personal data to the service provider, while proving he fulfills the profile access criteria and without disclosing any knowledge about the service to the identity provider. The privacy-preserving credential is issued by an identity provider to the card. Indeed, a user who has a card and wants to access a service from a service provider has to prove first that his profile fulfills different criteria required by the service provider. A card can get a privacy-preserving credential from an identity provider by proving the compliance of its profile stored in the card with the profile required by the service provider while preserving the user privacy. The identity provider learns nothing about the service delivered by the service provider, except the criteria required by the service provider to access the service. The information delivered to the service provider by the card, which is included in the privacy-preserving credential, is under the control of the user.

2.1 Cryptographic Primitives

The protocol relies on a number of cryptographic primitives. The corresponding notations are as follows:

- an encryption function (the encryption of m using the key k) : $\{m\}_k$
- a decryption function: dec
- a signature: $sign$
- a Message Authentication Code function: MAC
- a key derivation function : KDF
- two secure hash functions: H and H_{dh}

For this protocol, we use also the concatenation. The concatenation of the m_i is denoted by: $m_1 | m_2 | \dots | m_n$.

2.2 Description of the Protocol

The mERA protocol is a modular authentication protocol which means that it can be used in two different ways called mERA1-3 and mERA1-7. The mERA-based eServices with trusted third party protocol involves three parties: a smart card (ICC), a service provider (SP) and an identity provider (IdP). The goal of this protocol is to grant user access to some eServices while preserving user-privacy.

A smart card owns a master secret key which is shared by a large number of cards for user-privacy purpose, i.e. for preventing the identification by a service

provider of a specific card within the group of cards sharing the same master key. Every service provider owns a specific secret key which can also be computed on the card side from both the master key and the service provider's identifier. In addition, every service provider is equipped with a long term Diffie-Hellman key pair and a pseudo-certificate in order to prevent cards to impersonate a service provider.

In this protocol, we can distinguish three main phases:

- **Phase 1 (mERA1-3):** Exchanges between a smart card and a service provider. The card first requests to the service provider a set of criteria conditioning access to a service. It then authenticates the service provider while remaining anonymous. The authentication in this phase is performed using the mERA1-3 protocol.
- **Phase 2 (privacy-preserving mutual authentication):** a mutual authentication happens between the identity provider and the card. The protocol used for the mutual authentication during this phase is not part of the specification of the mERA-based eServices protocol. For the purpose of our verification we used the “device authentication with privacy protection” protocol described in [12]. At the end of this phase, the identity provider delivers a privacy-preserving credential.
- **Phase 3 (mERA1-7):** Exchanges between the card and a service provider. During this phase, the card requests access to the service provider. There is a mutual authentication using the mERA1-7 protocol and the card securely transmits the privacy-preserving credential to the service provider.

These exchanges are detailed in Figure 1.

Key Setup. The following setup of the system concerns only phases 1 and 3. For the second phase, we consider that it is performed using the “device authentication with privacy protection” protocol [12].

- A master key mk is generated by a probabilistic algorithm and transmitted to any legitimate card (ICC).
- A new identifier id_i is generated for the service provider (SP) and the corresponding sk_i is computed : $sk_i = KDF(mk, id_i)$.
- A private Diffie-Hellman element x_i , a public Diffie-Hellman element g^{x_i} and a certificate σ_i are generated and securely transmitted to SP. Then, SP securely receives its personal secret values (sk_i, x_i) and its personal public values $(id_i, g^{x_i}, \sigma_i)$.

Phase 1 Get the Profile

1. The user first requests the service provider to get the profile (access conditions associated to the service) and he checks if it is acceptable. Afterwards, the user authenticates himself via PIN presentation to notify his content; this step is not included in our formal verification.
2. The service provider is authenticated by the card in order to prove to the card that it is authorized to store a profile in the card.

3. The card generates an ephemeral key pair (u, g^u) serving to link the profile to the service provider and to securely connect later to this service provider.

During this phase, the profile is transmitted encrypted and the ciphertext is transmitted with a Message Authentication Code (MAC)

Phase 2. Getting the Privacy-Preserving Credentials with Mutual Authentication

4. A mutual authentication takes place between the identity provider and the card, and a secure channel is established protecting all following commands.
5. The identity provider may verify the revocation status of the card.
6. The identity provider recovers the public part of the ephemeral key g^u and the profile. It asks for the user's consent and makes the necessary verifications about the questions and answers.
7. The identity provider delivers to the card a privacy-preserving credential including the public part of the ephemeral key g^u and the profile, and signed with its private key.

During this phase, the profile, the public part of the ephemeral key g^u and the privacy-preserving credential are transmitted in a secure channel. The user also checks the profile.

Phase 3. Use of the Privacy-Preserving Credentials

8. A mutual authentication takes place between the card and the service provider.
9. At the end of the authentication, the card and the service provider share a common DH-based ephemeral secret, based on their public parameters exchanged during the mutual authentication. A secure channel is established between the card and the service provider. The card sends the privacy-preserving credential to the service provider, within the secure channel.
10. The service provider verifies the elements of the privacy-preserving credential and that this credential has been signed by a recognized identity provider. If the verification is successful, the service provider knows that the card is compatible with its use condition, and so the service provider grants access to the service.

3 The Formal Verification Tool: ProVerif

In this section, we review the verification tool that we use for our analysis.

The verification tool ProVerif (Protocol Verifier) [7] enables to perform a static analysis for cryptographic protocols under the assumption that the cryptographic primitives are idealized. The tool requires expressing protocols and the properties that we want to prove in a formal language. The tool is sound with respect to the semantics of this language. The formal verification using this tool guarantees the absence of the attacks as captured by the language semantics, but not necessarily of other attacks.

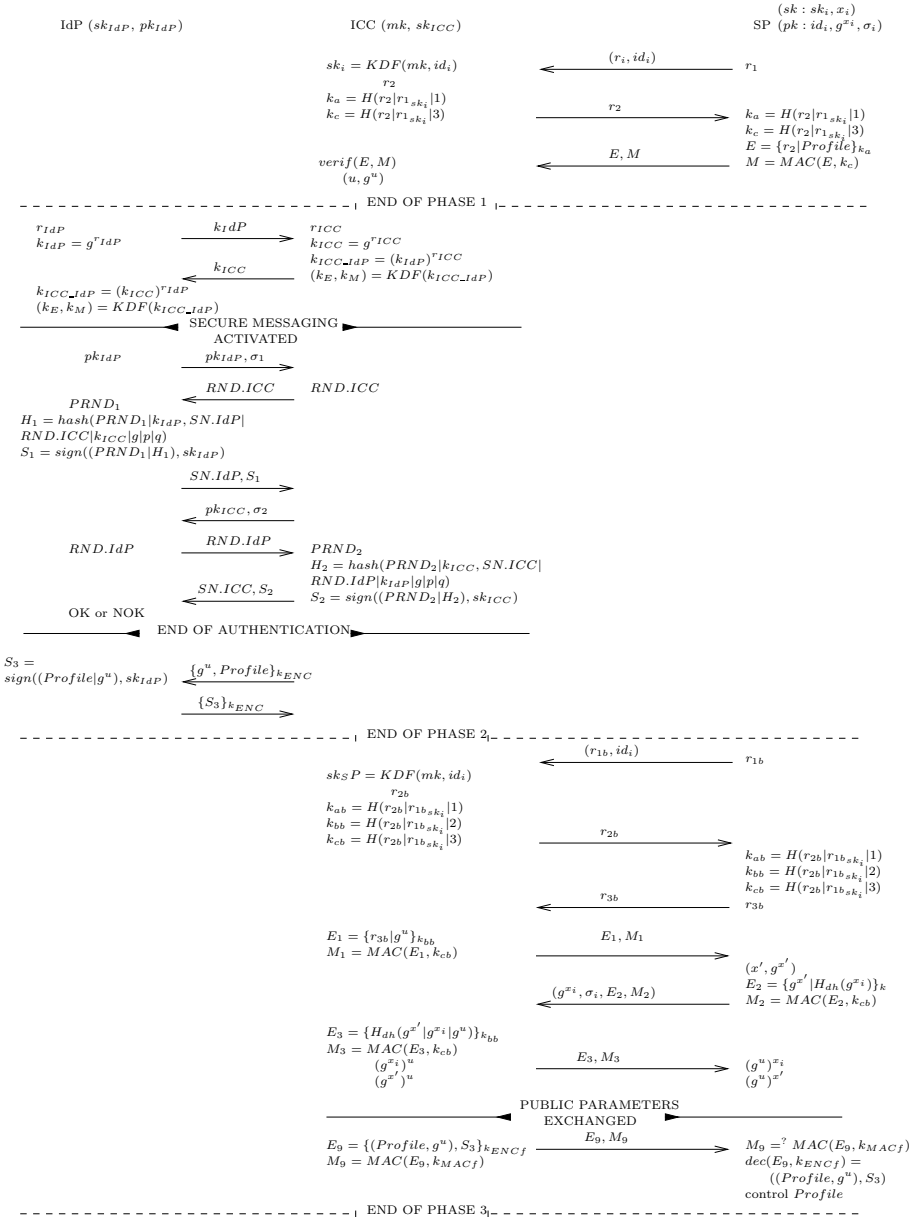


Fig. 1. mERA-based eServices with trusted third party protocol

3.1 Proof Engine

The proof-engine uses a resolution-based solving algorithm in order to determine properties of the protocol. Actually the verifier has to describe the protocol as well as the properties that (s)he wants to prove in a formal language. The tool proposes two formal languages for the protocol description; both languages are equivalents and are presented later on. ProVerif translates both languages into a set of Horn clauses before using a resolution-based algorithm (see [7] for more details).

In case the property we want to verify is not true, the output of ProVerif is a *trace* explaining the failure. For example, if we request the verification of the secrecy of a variable whereas this variable does not remain secret, the trace explaining the failure would be an attack path leading to reach this variable. In most cases, the output of ProVerif is either a confirmation that the property checked is verified, or an attack as a counterexample to robust safety. In rare cases that ProVerif does not terminate we cannot say anything.

3.2 The Language

As seen in Section 3.1 there are two options for the input language. The first one is an abstract representation of the protocol by a set of Horn clauses:

- predicate logic formulas of the form :
 $(\forall \mathbf{x})(P_1(\mathbf{p}_1) \wedge \dots \wedge P_n(\mathbf{p}_n) \Rightarrow Q(\mathbf{q}))$, where $n \geq 0$ and P_1, \dots, P_n, Q are predicate symbols from a fixed set and with fixed arity
- the clauses represent deduction rules for protocol participants and adversary
- one special clause, the goal, represents a property
- ProVerif checks, using resolution, whether $\{\text{goal, protocol, clauses}\}$ is a satisfiable set

The second option is the description of the protocol in a programming language which corresponds to an extension of the *pi calculus* (named applied pi-calculus) introduced by M. Abadi and C. Fournet in [3]. This calculus represents messages by terms M, N, \dots and programs by processes P, Q, \dots . Identifiers are partitioned into names, variables, constructors and destructors.

A function symbol with arity 0 is a constant symbol. Functions are distinguished in two categories: constructors and destructors. Constructors serve for building terms. Thus, the terms are variables, names, and constructors applications of the form $f(M_1, \dots, M_n)$. A constructor of arity n is introduced with the declaration *fun* f/n . On the other hand, destructors do not appear in terms, but only manipulate terms in processes. They are partial functions on terms that processes can apply. Typical examples of constructors are encryption and pairings. As for the destructors, we can consider decryption and projections. More generally, we can represent data structures and cryptographic operations using constructors and destructors (as we will see below in our coding of this protocol).

The grammar for processes of the ProVerif process language is the following:

$P, Q, R ::=$	plain processes
0	null process
$P \mid Q$	parallel composition
$!P$	replication
$\text{new } n; P$	name restriction
$\text{if } M = N \text{ then } P \text{ else } Q$	conditional
$\text{let } x = g(M_1, \dots, M_n) \text{ in } P \text{ else } Q$	destructor application
$\text{event } M \text{ } [;P]$	events
$\text{in } (M, x); P$	message input
$\text{out}(M, N); P$	message output
$\text{phase}(n) \text{ } [;P]$	phase indication

The null process does nothing. The replication $!P$ represents an unbounded number of copies of P in parallel. The restriction $\text{new } a; P$ creates a new name a , then executes P . The conditional is actually defined in terms of *let*. A destructor *equals* is defined, with the reduction $\text{equals}(x, x) \rightarrow (x)$. The *if* construct stands for *let* $x = \text{equals}(M, N)$ *in* P *else* Q . We also use some syntactic sugar to allow the *let*-construction to introduce abbreviations such as *let* $x = M$ *in* P .

The process calculus includes auxiliary events that are useful in specifying security properties. A query of the form *query* $ev: M \Rightarrow ev: N$, allows to express and to verify a property of the form “if the event M has been executed, then the event N must have been executed before”. For example, authentication between A and B can be seen as a query of the form “if B receives a message m , then A must have sent it before” and so it can be proved using the events.

Secrecy assumptions correspond to an optimization of our verifier. The declaration *not* M indicates to the verifier that M is secret. The verifier can then use this information to speed up the solving process. At the end of the process, the verifier checks whether the secrecy assumption is true, so that a wrong secrecy assumption is leading to an error message but not to an incorrect result.

The input process *in* $(M, x); P$ inputs a message on channel M , then runs P with the variable x bound to the input message. The output process *out* $(M, N); P$ outputs the message N on the channel M , then runs P .

The phase separation command *phase* $n; P$ indicates the beginning of phase n . When a process starts running it is in *phase* 0 . When a process enters the next phase, all remaining instructions from the previous phase are discarded. In that sense, one can understand the phase separators as global synchronization commands. The adversary obviously keeps its knowledge when changing phases.

We generally assume that processes execute in the presence of an adversary, which is itself a process in the same calculus and so it is allowed to execute the same actions as any other process. The adversary needs not to be programmed explicitly; we usually establish results with respect to all adversaries.

4 Verifying mERA Protocol

In order to analyze the mERA protocol (see Figure 1 for an informal description), we program it using the input language described in section 3.2. We give in Appendix A, an extract of the code which describes the exchanges that SP participates under the name of SP process.

4.1 Modelisation of Cryptographic Primitives

In the code, we firstly declare some cryptographic primitives. For example,

- * the constructor *encrypt* represents the encryption function, which takes two parameters, a public key and a message, and returns a ciphertext
- * the destructor *decrypt* represents the corresponding decryption function. From a ciphertext $encrypt(x, y)$ and the corresponding secret key y , it returns the ciphertext x . Hence we give the rule: $decrypt(encrypt(x, y), y) = x$. We assume perfect cryptography, so the cleartext can be obtained from the ciphertext only when one has the decryption key
- * the constructor *sign* representing a signature scheme, which takes two parameters, a secret key and a message, and returns a signature
- * the destructor *checksign*, checks the signature, while *getmess* returns the message without checking the signature. (In particular, the adversary may use *getmess* in order to obtain message contents from signed messages). *checksign* takes two parameters, the signature and the corresponding public key, and returns the message only if its signature is valid, whereas *getmess* has only one parameter, the signature, and returns the signed message without checking its signature.
- * the constructor *MAC* for the MAC function
- * the constructor *KDF* for the key derivation function
- * two constructors *hash* and *hash_{dh}* for the two hash functions that we need
- * the constructor *pk* in order to compose the public keys for the authentication from the secret keys, as well as to compose *id_i* from the *sk_i*
- * the constructor *g* used for the exponentiation of the generator of the group to a power

We analogously define the Diffie-Hellman function. Concatenation is represented by tuples, which are pre-defined by default in the tool with all the necessary operations.

4.2 Declarations and Channels

In ProVerif we can declare two types of variables. The *free* type variable which declares public free names and the *private free* variables which declares private free names (not known by the adversary).

Communication channels are often public free variables. So we declare them using the “free” declaration. For this protocol, we need three channels:

- `c` which is a public channel used for exchanges between the card and the service provider during the first and the third phase
- `auth` which is a second public channel used only between the card and the identity provider during the second phase of the protocol and
- `cert` which is a private channel used in order to communicate to a certification authority whenever we need to verify a certificate using the public keys of the corresponding entities. The certificates used are σ_1 -which is the identity provider’s certificate- , σ_2 -which is the card’s certificate- and σ_i -which is the service provider’s certificate- .

4.3 Processes

Until now, we have described the declaration of the cryptographic primitives and the different variables that we need for this protocol. So we are ready to describe the protocol itself.

Every entity of the protocol (in this case: ICC, IdP and SP) represents a process that describes every action of this entity using the language described before. For example (see the code in Appendix [A](#)), the process SP first generates a random number and then sends it as well as the SP identifier (noted pkSP) on the public channel `c`. Then it executes the steps of the protocol description.

The process *let $x = g(M_1, \dots, M_n)$ in P else Q* tries to evaluate $g(M_1, \dots, M_n)$; if this succeeds then x is bound to the result and P is run, else Q is run. A pattern $=M$ matches only the term M . So, on the example of Appendix [A](#), we can find the pattern *let $(=r3, pkDHICCr) = decrypt(enc1, kBb)$ in $[...]$* . This tries to evaluate $decrypt(enc1, kBb)$; if this succeeds, then $r3$ is bound to the result $decrypt(enc1, kBb)$ and the rest of the code is run. So the destructor $decrypt$ above succeeds if and only if $decrypt(enc1, kBb) = r3$.

In addition of the three processes corresponding to the three entities, we define two processes `lookup_cert` and `lookup_certSP` to verify the certificates.

All these processes are composed in the last part of the protocol specification (see Appendix [B](#)). This last part computes SP’s, ICC’s and IdP’s encryption keys from their secret keys by the constructor `pk`. For example: `pkSP = pk(skSP)` corresponds to the computation of public identifiers called here id_i using the constructor `pk` and the sk_i (`pkSP := id_i` and `skSP := sk_i`). Then the corresponding public keys are output in the public channel `c`, so that the adversary can have them. We proceed similarly for both the authentication keys (of ICC and IdP), as well as for SP’s Diffie-Hellman pair of keys (`skDHSP`, `pkDHSP`) but using the constructor `g` this time.

5 Properties Formally Verified

In this section, we present the properties that we have formally verified.

5.1 Liveness Property

The liveness property of a protocol consists in proving that the protocol is not in a permanent deadlock situation. That means that at least an exchange will occur during its life time.

In this case, we have successfully verified the protocol completion. It is verified that both an exchange between the card and the service provider and an exchange between the card and the identity provider will occur at some time in the life of the card.

This is done with the help of a *query* which verifies the existence of the events “ICC accepts SIG_3 ” (sent by IdP) and “SP accepts the signature SIG_3 ” (sent by the card). The first event verifies that at least one exchange between IdP and ICC will occur, and the second one that at least one exchange between ICC and SP will occur. So the verifier has to prove the existence of both events in order to ensure us that at least one exchange will occur between every entity who participates in the protocol.

5.2 Security Properties

Secrecy The secrecy is then a correctness property which must be an invariant of the protocol. The property must be true at the beginning of the protocol execution and must be true at all instances of the protocol execution. In ProVerif, this is done with the appropriate query *attacker “variable”*.

In particular,

- * the secret keys of all the parties of the protocol, that is: sk_i , x_i , sk_{ICC_AUTH} , sk_{IdP_AUTH} , remain secrets all over the protocol
- * the master key mk remains secret all over the protocol (if not the attacker could recover the sk_i and so run all the protocol instead of a legitimate card)
- * the secret part of the ephemeral DH keys (generated by both ICC -named u - during the first phase and SP -named x' - during the third phase of the protocol) remain secret throughout all the exchanges occurred in the protocol
- * the keys chosen randomly by both IdP and ICC during the DH of the second phase of the protocol (that is r_{IdP} and r_{ICC}) remain secrets all over the protocol
- * the common key between ICC and IdP used in order to build the keys of encryption and MAC during the second phase of the protocol ($g^{r_{ICC} \cdot r_{IdP}}$) remains secret all over the protocol
- * the profile of ICC remains secret to the attacker throughout the second phase of the protocol
- * the keys shared between SP and ICC used at the end of the protocol in order to build the final keys of encryption and MAC (i.e. $g^{x_i \cdot u}$ and $g^{x' \cdot u}$) remain secrets throughout the protocol
- * even the public part of the DH key generated by ICC remains secret throughout the protocol and finally
- * the unique identifiers (e.g. unique public key, unique serial number) of both ICC and IdP remain secrets throughout the protocol

Message Authentication. Authentication is used to verify that a party is indeed who (s)he claims to be.

The message authentication is required in different points of the protocol (i.e. verify that if an information is received by an entity, then only the corresponding entity could have sent it). More precisely:

- * the authentication of the Profile sent by ICC to SP during the first phase
- * the authentication of the public keys of both ICC and IdP and their certificates as well as the authentication of the signatures exchanged during the mutual authentication of the second phase of the protocol
- * the authentication of the public part of the ephemeral key of ICC (g^u) and the Profile sent by ICC to IdP, as well as the one of the credential sent by IdP to ICC (during the second phase of the protocol)
- * the DH-message authentication during the third phase of the protocol, i.e. if SP accepts a valid public part of an ephemeral key, then only the ICC that has the secret part of the key can have sent it and respectively for the public part of the SP's ephemeral key
- * during the third phase, we also verify the validity of the SP's DH key and its certificate
- * the DH-message authentication of the three public keys : g^u , $g^{x'}$ and g^{x_i} , at the end of the third phase of the protocol; this is verified through the establishment of a secure channel by using k_{ENCf} and k_{MACf}
- * the authentication of the final signature sent by ICC to SP during the third phase of the protocol and which includes the public part of the ephemeral key of ICC (g^u) and the Profile

All these properties are formalized and verified using *queries* about *events*, as described in Section 3. For example, the query:

$$\text{query evinj: ICCaccepts}(t) \implies \text{evinj: SPsends}(t).$$

is verified during the first phase of the protocol and check the authentication of the profile, that is if the event “ICC receives a profile” occurs, then the event “this profile is the one sent by SP” has occurred before.

Unlinkability. The notion of the unlinkability [11] of two or more items of interest means that within the system (including these and possibly other items), from the attackers perspective, these items of interest are no more and no less related after his observation than before according to his initial knowledge.

In ProVerif, this property can be proved either using the *choice* construction (which helps us prove an observational equivalence between two processes) or the *noninterf* query. Using the choice-construction, we can express two processes in one. For example, if we want to verify the unlinkability about the message, we can introduce two messages m_1 and m_2 and using the *choice*[m_1, m_2] ask ProVerif if m_1 and m_2 are equivalents from an observer point of view. If it is the case, this means that either we use m_1 or m_2 , an observer cannot see any difference, which is what we want to prove in the case of the unlinkability.

On the other hand, “*noninterf* x_1, x_2, \dots ” is supposed to check if a process preserves secrecy of variables x_1, x_2, \dots in the strong sense. Strong secrecy means that an adversary cannot see any difference when the value of the secret changes, which is exactly what we are looking to prove for the unlinkability. The choice between the *choice*-construction and the *noninterf*-command is up to the user according to the elements for which (s)he wants to prove the unlinkability and the complexity of the protocol.

In the case of mERA, we verified the unlinkability for two different executions of the phase 1, phase 2 and phase 3, the unlinkability between the phase 1 and the phase 2, between the phases 1 and 3 and between the phases 2 and 3. For example, for the case of the unlinkability between the phases 1 and 3, we had to prove it for the profile as well as the Diffie-Hellman key produced at the end of the phase 1 - u, g^u -.

6 Conclusion

In this paper, we present the formal verification of the mERA-based eServices with trusted third party protocol, as well as the verification of its main security properties, that is secrecy, authentication and unlinkability. This is done using an automatic protocol verifier. The aim of using an automatic verifier is to reduce the human interaction in a so complicated proof and so reduce the risk of error.

In our case, we used ProVerif. An automatic verifier which, unlike other verifiers, does not limit the number of concurrent protocol runs that a modeled attacker may execute. So, it is able to find attacks that require any number of concurrent protocol runs. ProVerif is completely automatic, in the sense that after starting the verifier, it does not need any user input to complete its task. However, a user has to completely understand and model the whole protocol as well as describe the properties that (s)he wants to prove. This step is often the most difficult part of the verification, mainly for complicated protocols like mERA.

Acknowledgments. Maria Christofi’s research is a PhD research work partly funded by ANRT (CIFRE).

References

1. Abadi, M., Blanchet, B.: Computer-assisted verification of a protocol for certified email. *Sci. Comput. Program.* 58(1-2), 3–27 (2005)
2. Abadi, M., Blanchet, B., Fournet, C.: Just fast keying in the pi calculus. *ACM Trans. Inf. Syst. Secur.* 10(3) (2007)
3. Abadi, M., Fournet, C.: Mobile values, new names, and secure communication. In: *POPL*, pp. 104–115 (2001)
4. ANTS, Gemalto, Oberthur Technologies, and Safran Morpho. Access to e-services with privacy-preserving credentials. Technical Report (August 10, 2011), http://www.ants.interieur.gouv.fr/IMG/pdf/IAS/TR-Privacy_Preserving_Credentials_10082011_v2.0.pdf

5. Armando, A., Basin, D., Boichut, Y., Chevalier, Y., Compagna, L., Cuellar, J., Drielsma, P.H., Heám, P.C., Kouchnarenko, O., Mantovani, J., Mödersheim, S., von Oheimb, D., Rusinowitch, M., Santiago, J., Turuani, M., Viganò, L., Vigneron, L.: The AVISPA Tool for the Automated Validation of Internet Security Protocols and Applications. In: Etesami, K., Rajamani, S.K. (eds.) CAV 2005. LNCS, vol. 3576, pp. 281–285. Springer, Heidelberg (2005)
6. Backes, M., Hritcu, C., Maffei, M.: Automated verification of remote electronic voting protocols in the applied pi-calculus. In: Proceedings of the 21st IEEE Computer Security Foundations Symposium, CSF 2008, Pittsburgh, Pennsylvania, June 23–25, pp. 195–209. IEEE Computer Society (2008)
7. Blanchet, B.: From Secrecy to Authenticity in Security Protocols. In: Hermenegildo, M.V., Puebla, G. (eds.) SAS 2002. LNCS, vol. 2477, pp. 342–359. Springer, Heidelberg (2002)
8. Dolev, D., Yao, A.C.-C.: On the security of public key protocols. *IEEE Transactions on Information Theory* 29(2), 198–207 (1983)
9. Krawczyk, H.: SKEME: a versatile secure key exchange mechanism for Internet. In: Symposium on Network and Distributed System Security, p. 114 (1996)
10. Kremer, S., Ryan, M.: Analysis of an Electronic Voting Protocol in the Applied Pi Calculus. In: Sagiv, M. (ed.) ESOP 2005. LNCS, vol. 3444, pp. 186–200. Springer, Heidelberg (2005)
11. Pfitzmann, A., Köhntopp, M.: Anonymity, Unobservability, and Pseudonymity - A Proposal for Terminology. In: Federrath, H. (ed.) Anonymity 2000. LNCS, vol. 2009, pp. 1–9. Springer, Heidelberg (2001)
12. CEN/TC224 prEN 14890-1:2008. Application interface for smart cards used as secure signature creation devices (2008)

A Part of the Proof: The SP Process

```

let SP =   new r1;
           out (c, (r1, pkSP));
           in (c, NX);
             let zz = encrypt ((NX,r1), skSP) in
let kA = hash((zz,1)) in
let kC = hash((zz,3)) in
let sh = (NX, Profile) in
  event SPsends(Profile);
    let E = encrypt(sh, kA) in
  out (c, (E, MAC (E, kC)));
phase 1; (* empty because SP doesn't participate in phase 1 *)
phase 2;
           new r1b;
           out(c, (r1b, pkSP));
           in(c, NXb);
let zzb = encrypt((NXb,r1b), skSP) in
let kAb = hash ((zzb,1)) in
let kBb = hash ((zzb,2)) in
let kCb = hash ((zzb,3)) in
  new r3;
  out(c, r3);
  in (c, (enc1, mac1) );
  if mac1 = MAC( enc1, kCb) then
let (=r3, pkDHICCr) = decrypt (enc1, kBb) in
event SPaccepts(pkDHICCr);
new x1; (* x' *)
let gx1 = g(x1) in
  event SPsendsDH(gx1);
    let ENC2 = encrypt ( (gx1, hash_dh(pkDHSP)), kAb) in
  out( c, (pkDHSP, sigmaSP, ENC2, MAC ( ENC2, kCb)));
  in ( c, (enc3, mac3));
  if mac3 = MAC( enc3, kCb) then
  if hash4 ( gx1, pkDHSP, pkDHICCr) = decrypt (enc1, kBb) then
    out (c, ok);
    let k1 = f (skDHSP, pkDHICCr ) in
let k2 = f (x1, pkDHICCr) in
  let ( kENCf, kMACf) = KDF3 (k1 k2) in
in (c, (enc9, mac9));
  if mac9 = MAC( enc9, kMACf) then
let (Profile, =pkDHICCr, SIG3) = decrypt ( enc9, kENCf) in
  if (Profile, pkDHICCr ) = checksign ( SIG3, pkIdP) then
    event SignAccepts(SIG3).

```

B Part of the Proof: The Composition of All Processes

```

process
new skSP;
  let pkSP = pk(skSP) in out (c, pkSP);
new skICC;
new skDHSP;
  let pkDHSP = g(skDHSP) in out (c, pkDHSP );
new skDHICC;
new SN_ICC;
new skICC_AUTH;
  let pkICC_AUTH = pk (skICC_AUTH) in out (c, pkICC_AUTH);
new SN_IdP;
new skIdP_AUTH;
  let pkIdP_AUTH = pk (skIdP_AUTH) in out (c, pkIdP_AUTH);
new skIdP;
  let pkIdP = pk (skIdP)in out (auth, pkIdP);
( !lookup_cert
| !lookup_certSP
| !SP
| !ICC
| !IdP )

```

My Authentication Album: Adaptive Images-Based Login Mechanism

Amir Herzberg and Ronen Margulies

Dept. of Computer Science, Bar Ilan University
{herzbea,margolr}@cs.biu.ac.il

Abstract. We present the design and user study of an adaptive authentication mechanism based on recognition of user-custom images. The mechanism relies on memorizing the custom images on each primary login, and adaptively increasing the authentication difficulty upon failures (suspected impersonation attempts). The constant memorization of the images allows fallback authentication by recognizing all/most of the user's custom images. Our mechanism can be used for multiple authentication scenarios; in particular, it can provide effective phishing protection for primary and/or fallback web login. The mechanism features quick authentication times and low guessing probabilities.

Keywords: web authentication, phishing, fallback authentication, password reset, memorability, human factors, user study, security by design.

1 Introduction

In today's world, users have accounts to many websites, devices and systems which require them to remember a password in order to identify. As seen on [1] and other studies, most users can only remember a limited amount of passwords, especially if those passwords are strong and hard-to-guess. Therefore, many users often forget their passwords. To deal with passwords forgetfulness, two families of countermeasures were suggested:

- Automatic fallback authentication techniques, aiming to allow authentication even if the password is forgotten.
- Memorable alternatives to textual passwords, aiming to *prevent users from forgetting their passwords* in the first place.

Each authentication technique, primary or fallback, should deal with the following security and usability parameters:

Susceptibility to guessing attacks which includes both the password space (which should be large enough to effectively prevent brute-force attacks), and the susceptibility for educated guesses which could significantly reduce the password space.

Susceptibility to phishing primary/fallback authentication techniques should prevent users from submitting their identifying information to a spoofed site.

Memorability of the identifying information provided by the user.
Usability the authentication times and ease of use.

Fallback authentication techniques ought to be (almost) as secure as the primary authentication in their resistance to guessing and phishing attacks, or else attackers will focus on attacking the fallback authentication mechanism, instead of the primary. In addition, since the fallback authentication process is not used frequently, the memorability of the identifying information has an increased importance. Since *easily-remembered* information might be *easily-guessed*, it is a true challenge to preserve its memorability and prevent its predictability. On the other hand, the fallback authentication could be somewhat slower and less convenient in comparison to the primary login.

Like fallback authentication, alternatives for the primary login also need to be *as secure as the login ceremony they intend to replace*, and should not have worse authentication times.

Textual passwords encounter some additional problems: users often choose passwords that are easily guessed, keyloggers can transfer them to an attacker, and sometimes typing is inconvenient/infeasible.

1.1 The Fallback Authentication Challenge

Today's most common mechanisms for automatic fallback authentication/password recovery are email-based mechanisms and security questions (a.k.a. challenge questions). Email-based mechanisms require access to a pre-configured email address, in order for the site to send back the password or a link to a password reset page. Security questions are used as identifying information, provided by the user during registration and supposed to be known only to the user.

Email-based password recovery reduces the security to that of the email account, which is often unacceptable. Security questions suffer from a few problems. First, they are susceptible to phishing attacks if no effective security indicators are used to help the user identify she reached her target site. This is due to the fact that security questions, like login forms, stimulate an automatic response among users causing them to answer the questions mindlessly. Karlof et al. found 92.7% spoof rate in their user study of security questions [5].

In addition, early and recent studies of leading websites have shown that users tend to forget ~20% of their answers and they are easily-guessed due to inherently *low entropy* [12, 7, 8]. If a few and/or trivial questions are used, it is easy to find the answers, especially in today's social-networked world, where attackers can harvest all kinds of 'secret' information about users. Using many and/or non-trivial questions makes it harder for users to remember their questions for a long period of time.

Finally, the security questions mechanism does not prevent attackers from accessing the fallback authentication page, where they can try guessing the user's answers, possibly by presenting the same questions to the user and using the answers.

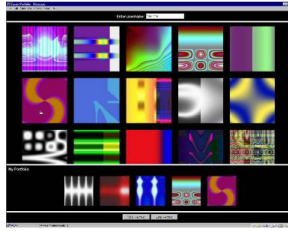


Fig. 1. Dèjà Vu’s abstract images selection

1.2 Graphical Passwords

Graphical passwords, a general name for authentication methods where images are used for authentication, were originally proposed as an *alternative to textual passwords* to deal with the problems textual passwords encounter:

- Ease of use on devices where typing is inconvenient (e.g. mobile devices with a touch screen, ATMs).
- Increased memorability – Images are better remembered than textual phrases [9].
- It is hard to write down a graphical password, or tell it to someone orally.
- Resistance to keyloggers.

An example for a graphical password scheme is Dèjà Vu, which was suggested by Dhamija and Perrig[2]. Dèjà Vu uses randomly created abstract images, which are unlikely to be guessed. On the authentication screen Dèjà Vu displays a grid with n images, k of whom are the user’s custom images, which the user should identify (by ‘clicking’) to login; see figure 1.

Graphical passwords seem like a very nice and appealing alternative to textual passwords, and indeed several additional graphical password schemes were proposed [10, 13]. Yet, due to their rather long authentication times (~ 30 seconds in [2]), graphical passwords were not widely deployed as a substitute for textual passwords. In fact, for that reason they are *better suited for fallback authentication*.

In particular, a fallback authentication technique which require users to recognize items (as text phrases or images) they like or dislike was suggested by Jakobsson et al. [4]. Jakobsson et al. argue that this method is preferably, since it has higher entropy, and since they believe that preferences don’t change much over the years, hence are better remembered.

Yet, in today’s social-networked world users often publish their preferences in many areas, so (part of) the fallback authentication information can be found. In addition, as discussed in section 3, this method is susceptible to educated guesses. Finally, in their evaluation Jakobsson et al. found that 16 items (8 likes and 8 dislikes) are needed for sufficient security. We believe that recognizing such a large set of images/items after a long period of time from the initial setup, may prove hard, especially without periodically refreshing the user’s memory.

The poor authentication times for primary authentication and poor memorability for fallback authentication raises a dilemma for the usefulness of graphical passwords. We propose a “learn-by-use” login mechanism which provides memorization of custom-images on the primary login while preserving its quick authentication times, and requires the user to recognize all/most of her images when fallback authentication is requested.

1.3 Contributions

We have designed and implemented a learn-by-use authentication mechanism, which is comprised of the following elements:

Single stage authentication requires the user to recognize and click one/few of her custom images from a small set of images on each primary login.

Multiple Stages Authentication requires the user to recognize all/most of her custom images in multiple stages. The multiple-stages authentication is used for fallback authentication or as a difficult authentication scheme when an impersonation attack was detected. The use of multiple stages allows gradually increasing the authentication difficulty by increasing the number of stages.

Detecting impersonation attacks and handling them with different levels of certainty.

Adaptivity using different authentication difficulties on different scenarios – weak authentication when additional identifiers are provided (passwords, cookies) and increasing the difficulty when no identifiers are provided and upon impersonation attacks.

One contribution of our mechanism is the constant memorization of the identifying information (custom images) it provides to the users, which can be used when fallback authentication is needed. Another contribution is the adaptive nature of our mechanism, which detects impersonation attacks and increases the authentication difficulty gradually as the certainty that an attack is witnessed increases. We also present the new concept (to the best of our knowledge) of detecting that part of the authentication secret is exposed to an attacker and asking the user to replace it upon high attack-certainty.

Another contribution is the flexibility of our mechanism, which allows different levels of security and usability. We present two main authentication usages; one that is more stringent and provides two-factor authentication and phishing protection, hence more suitable to sensitive sites; and one that is more alleviate, hence more suitable for less-sensitive sites. In both usages, our mechanism increases the security level of current login ceremonies without increasing the authentication times, provides a secure fallback authentication ceremony and handles impersonation attacks.

Finally, we conducted a long-term user study, whose results confirmed our mechanism’s long-term memorability, quick authentication times and effective phishing-resistance.

1.4 Paper Layout

The next section describes our mechanism. Section 3 provides a security analysis of our mechanism in comparison to related work. Section 4 discusses its different applications and section 5 describes our user study and its results.

2 Mechanism Description

Registration. During registration the user chooses (or assigned) a small number of custom images (k) from a large collection of images.

Single Stage Weak Authentication. On the primary login, the system displays a small set of L images, one of whom is one of the user's k custom images which she has to click in order to login. After each *successful* login, a new set of L images is chosen randomly for the next login.

Since the user's custom images are displayed and clicked on each login, the primary login ceremony *constantly refreshes the user's memory* of her custom images, so they are better remembered when fallback authentication is needed. The single stage authentication provides only weak authentication ($1/L$ guessing probability) and in most applications will be used in conjunction with other identifiers (passwords, cookies, etc.). In addition, our mechanism increases its authentication difficulty adaptively upon suspected impersonation attacks, thus the weak authentication can rapidly become strong.

Multiple Stages Authentication. On the multiple stages authentication, which is used for fallback authentication or upon suspecting attacks, the user is displayed with a series of k stages. Each stage displays n images, only one of whom is a user-custom image, and the user has to correctly click all/most of her custom images. If the user has chosen a wrong image in any of the stages, the ceremony continues until her k -th click and only then announces an unsuccessful authentication attempt. See figure 2.

It is important to note that in all stages of *different* authentication attempts, the *same images* are displayed on each stage, or else an attacker will be able to isolate the user's custom images from the non-custom images.

Detecting and Handling Impersonation Attacks. Since one of the user's custom images is displayed on the primary login page, an attacker can find that image by accessing the primary login page and providing an initial identifier. After each successful login, the custom image will differ the next time the user (or the attacker) reaches the login page. This way, the attacker could constantly monitor the login page and check if any new images are displayed, until he finds most/all of the user's custom images.

An impersonation attack includes both a monitoring attack, and guessing attempts of the user's password or custom images. The first stage is to *detect* such attacks, and there are several ways for doing so. The system can notice when a user reaches the login page and does not try to authenticate at all, or

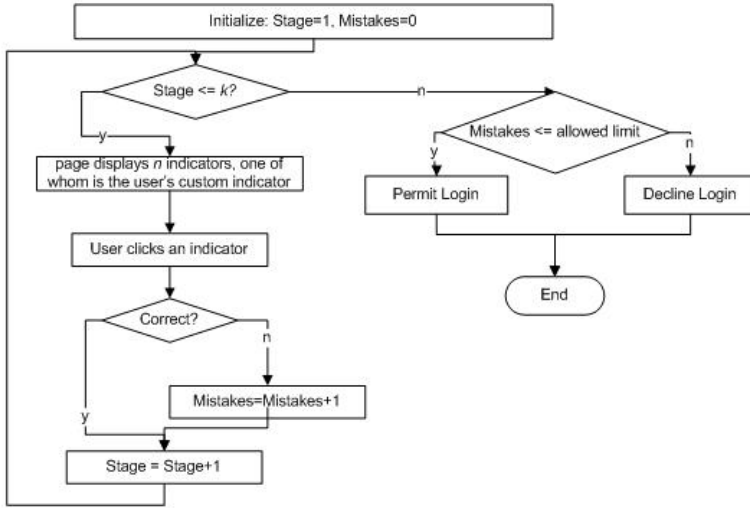


Fig. 2. The multiple (k) stages authentication scheme. The user must click all/most of her custom images.

mistakenly trying to guess the user's correct image (out of L), especially if this happened several times within a pre-determined period of time. Guessing attacks are detected by failed authentication attempts, i.e. submitting a wrong password or clicking incorrect images on the fallback authentication page.

After the system detected a suspected attack it increases the authentication difficulty adaptively to prevent the attacker from gaining more of the user's custom images or apply guessing attacks. As the difficulty increases, the attacker is likely to make more mistaken clicks or disconnect without trying to authenticate, which increases the attack's certainty even more.

First, the system can increase the number of images displayed on the login page (for example, double them) after reaching some certainty threshold. Second, the system can apply the multiple stages scheme for increased difficulty. Third, the amount of stages can also be increased to $m > k$ stages, each containing a "none of my images are here" choice, and only k of the stages display a user-custom image. The authentication difficulty can be further increased by using hindering measures such as CAPTCHAs between the stages.

After a pre-configured level of certainty was reached, the system can ask the user to replace her initial identifier (discussed on next section), and by logging the images that the suspected attacker correctly clicked, can also ask the user to replace those images. The site can also deal with guessing attempts of the textual password; if the guessing attempts were close enough to the user's correct textual password (for example, a small edit distance value), the site can ask the user to replace her password.

On [6] we provide an algorithm for detecting and handling impersonation attacks, with varying levels of certainty, by using all the above-mentioned measures.

3 Security Analysis

In this section we analyze the security strength of our method in comparison to Dèjà Vu [2] and preferences-based security questions [4] we discussed previously. The theoretical password space of our mechanism is n^k and Dèjà Vu's is $\binom{n}{k}$ (390,625 and 12,650 respectively for reasonable values of $n=25$ and $k=4$). The theoretical password space of preferences-based security questions is $\binom{16}{8} = 12,870$, similar to Dèjà Vu's.

As seen on Davis et al.'s user study [1], user chosen images/items can be highly predictable, especially among certain populations (based on gender, geographic location, age, etc.). It was evidenced in [1] and in [2] that more appealing images and popular items (objects, places, food, etc.) have higher probabilities to be selected by users. In addition, correlations between images/items also exist. Therefore, basic prior knowledge of the users' background and preferences (taken for example from social networks) as well as knowledge of the selection probabilities and correlations (among the general population or different populations) can significantly increase the guessing probabilities in all three methods.

With preferences-based security questions, a unique correlation also exist: category-based correlation. Since users are requested to select their items from different categories, a person who likes sports, for example, might choose most of his 'likes' from the sports category. The mere fact that a person likes one of the categories in general could be quite easily drawn from social networks. If users are allowed to choose only few likes/dislikes from each category, then they won't choose their most liked and most disliked items, resulting in weaker memorability.

To avoid different probabilities and correlation as much as possible, the images used by a site implementing images-based authentication should:

- Be as general as possible (e.g., abstract images) and/or taken from the same category/topic. The site's topic can be a good idea for the images portfolio.
- Be unfamiliar. For example, a traveling agency should use general images of landscapes and urban views, instead of popular sites.
- Have the same level of appeal.

Obviously, preferences-based security questions cannot follow those guidelines. Therefore, it is likely that our method and Dèjà Vu have lower guessing probabilities than preferences-based security questions, and the actual password space should be close to the theoretical password space (though correlations and different probabilities are still possible with abstract images or images from the same topic). Another advantage over preferences-based security questions is the ability of the site to continuously learn guessing probabilities and correlations, and to remove images with high/low probabilities and avoid displaying correlating images together in the setup phase of a specific user. Therefore, since

Dèjà Vu has a similar theoretical password space as preferences-based security questions, but reduced guessing probabilities (with a good choice of the images portfolio), its security is stronger. In addition, since the guessing probabilities of our method and Dèjà Vu’s are similar, our larger theoretical password space gets the lead.

Another advantage of our method in comparison to Dèjà Vu and preferences-based security questions is the ability to increase the authentication difficulty to the m -stages scheme. Suppose the attacker knows c of the user’s custom images; with the k -stages scheme his guessing probability is $(\frac{1}{n})^{k-c}$. With the m -stages scheme, he has to guess the correct $k-c$ stages where the other custom images are displayed, and then guess the correct images on those stages. This reduces the guessing probability to $\frac{1}{\binom{m-c}{k-c}} \cdot (\frac{1}{n})^{k-c}$.

4 Applications

4.1 Authentication to Highly-Sensitive Sites

On [3] we describe an extensive long-term user study we conducted, which examined login mechanisms that force and train users to login safely and resist phishing attacks. We tested two *forcing functions* mechanisms:

A login bookmark which forces users to reach the site’s login page via the bookmark by not allowing them to login when reaching it otherwise. The login page looks for a secret token contained in the bookmark link and presents an error message if a familiar token is not provided. This mechanism also provides *two-factor authentication* as both the secret token and the password are necessary for authentication.

An interactive user-custom image which forces the user to look for and click her custom image on the login page, by hiding the password text field until the custom image is clicked.

The study’s results showed that a combined method, which uses a login bookmark with an interactive custom image, displays several images in the login page, and training the user not to follow links by intentionally providing ‘non-working’ links, received outstanding prevention and detection rates, and resisted 93% of the attacks.

We exploited the idea of the login bookmark and interactive custom images, and used them in conjunction with our learn-by-use authentication mechanism. The only difference of the primary login in comparison to the combined method in [3] is that the user now has k custom images instead of just one, and one of them is displayed on the login page along with $L-1$ other images (e.g. $L=4$). In addition to the site identification that the image click provides to the user, it also provides her with the necessary memorization of the custom images.

Users can choose to go to the fallback authentication page when they have forgotten their passwords or when using a mobile application/browser on a mobile device, where it might be easier to click images than typing the password.

Access to a user's fallback authentication page requires the secret token, just like the primary login. This prevents anyone who doesn't have the secret token from accessing the fallback authentication page, and from performing guessing attacks or spoofing the fallback authentication page.

Entrance to the fallback authentication page can be done via the primary login page; After the bookmark click, the site initially identifies the user by her secret token and sends back the login page with the user's custom image. After the user clicks her custom image, the site reveals, in addition to the password text field, a button which leads the user to the fallback authentication page. This promises that the user effectively identifies the site before trying to authenticate. It is important to note that the login bookmark itself isn't necessary to achieve two-factor and two-sided authentication, only the secret token is. An alternative can be a cookie containing the secret token, which should be previously installed via a secondary channel on each computer the user uses (for example with a registration email). Yet, we recommend using the login bookmark for several security and usability advantages discussed on [3].

We designed and implemented WAPP (Web Application Phishing Protection), a server side solution which implements the above mentioned mechanisms for primary login, as well as our fallback authentication mechanism (see <http://submit2.cs.biu.ac.il/WAPP/> for a live demo).

4.2 Authentication to Less-Sensitive Sites

The mechanism we described can be altered for less-sensitive sites which do not aim to provide effective phishing protection or two-factor authentication. Many of today's sites that require authentication (e.g. gmail, facebook) provide the user with an option to create an authentication cookie containing a secret token when the user provides her password for the first time on a certain browser (usually with a "keep me signed in" check box). On the next logins from the same browser, those sites don't require a password at all if the cookie is installed. Our mechanism can also be used on such sites:

- When an authentication cookie is already installed, the site can display one of the user's custom images, together with several non-custom images (e.g., $L=9$) which provides the necessary memorization and further security (since a password is not required). The mechanism can be set to allow no mistakes and double the amount of images immediately after a single mistaken click (and increasing the difficulty rapidly on further mistakes). If a password is also required, the mechanism also provides phishing protection.
- When an authentication cookie is not installed, no other initial identifier (such as the username) should be used instead. If the username is used to initially recognize the user and display her image, then the site is susceptible to MITM attacks: the attacker can spoof the initial page where the user provides her username, forward the username to the original site, get the image and display it to the user, and finally get the password from the user. This even provides a false-sense-of-security to the users.

Therefore, the site should first ask for the password and only then display the images. This does not provide phishing protection, but it provides the necessary memorization and further weak (and increasing) security.

- If the user has forgotten her password or uses a mobile device, the site can allow her to authenticate using the k -stages authentication by providing her username only.

5 User Study

5.1 Study’s Framework System

For our long-term phishing study in [3], we used an online exercise submission system called ‘submit’ (submit.cs.biu.ac.il), which is used by most courses at the computer science department of Bar-Ilan University. With the submit system, students submit their exercises and receive emails announcing new grades. Due to its wide usage, most users logged-in to the system dozens to hundreds of times throughout the study. We have used the system for three semesters, among a population of ~ 400 students. In addition to its primary usage as an exercise submission system, we simulated several phishing attacks and collected the results.

We extended the phishing study to test our fallback authentication mechanism in the fourth semester. Users were asked to choose 3 or 4 new custom images (in addition to their existing custom image). One of their custom images was randomly selected and displayed on the login page on each login (along with three other non-custom images). Users were tested with the k -stages authentication scheme twice – immediately after they have chosen their additional custom images and once again 2-3 months later, while using the system regularly (and clicking one of their images on each login). In both tests, users had only one authentication attempt, and they were not forced to succeed in order to proceed with their exercise submission.

We logged the success rates and authentication times, and also examined if there was a decrease in the detection rates of spoofed login pages in comparison to the previous semesters (where users had only one custom image), especially for spoofed login pages displaying only fake images.

5.2 Results

The memorability of the custom images and the authentication times are of course affected by the amount of entrances to the system between the setup and the authentication attempt. The more times a user enters the system, the better she remembers her images and authenticates faster. Table 1 summarizes our results and compares them to Dhamija and Perrig’s study [2], where all users authenticated once after one week from the setup phase. The table displays the results of users from the top 25% of entrances (21+), top 50% of entrances (8+) and a base line of minimum entrances we chose (5+). The results show reasonable authentication times, similar to Dèjà Vu.

Table 1. Memorability and authentication times of our method in comparison to Dèjà Vu. We expect that allowing several login attempts will improve our method’s results.

method	entrances	images used	weeks from setup	no mistakes	up to one mistake	auth. attempts	auth. time (seconds)
Ours	5+	photos+abstract	8-12	65.3%±11.2	87.7%±7.7	1	38
Ours	8+	photos+abstract	8-12	67.5%±12.2	90%±7.8	1	38
Ours	21+	photos+abstract	8-12	84.2%±13.8	94.7%±8.4	1	31
Dèjà Vu	0	photos	1	95%±8	100%	no limit	27
Dèjà Vu	0	abstract	1	90%±11	100%	no limit	32

The success rates of users who entered the system eight times or more are quite good, especially if the system allows a single mistake. For users who entered the system more than 20 times, the results are very good even when no mistake is allowed. The results are expected to improve even more if several attempts are allowed and if users will *have to* succeed to use the system. Yet, it can be assumed that users remembered their custom image that preceded the experiment, so the results can be somewhat reduced. For a better picture, a longer-term experiment, which compares our method with other techniques, is still needed.

Surprisingly, we found better success rates for users with five custom images than users with four, suggesting that using five images and allowing a single mistake is the best option. Even with four images (out of five) needed for authentication, the password space of our method is larger than the password space of Dèjà Vu with five images¹.

Finally, there was *no degradation* in the detection rates of spoofed pages for users with eight or more entrances, and no increased false negative rates (falsely reporting a non-spoofed page).

6 Conclusions and Future Work

We have described an images-based authentication mechanism which reminds the user of her custom images on each primary login, so they will be highly memorable when a fallback authentication is needed. Our fallback authentication mechanism was shown to be quick and memorable after sufficient training on our user study. Our mechanism also increases its difficulty adaptively upon detecting impersonation attacks. The mechanism is well suited for different web authentication scenarios, providing different levels of security and quick authentication times. Our fallback authentication scheme provides a large password space and high resistance to guessing attacks, and for sensitive sites it also provides effective phishing protection.

Another important authentication scenario that our mechanism suits for, is the authentication to mobile devices. Since authentication to mobile devices should be quick (as it is done very frequently) and the mere possession of

¹ $25^4 / \binom{5}{4} = 78,125$ and $\binom{25}{5} = 53,130$ respectively.

the device can be assumed to provide some identification, than the authentication can start with our single-stage authentication and rapidly increase its difficulty upon each mistaken click. It can also be combined with other identifying measures.

Acknowledgments. This research was supported by grant 1354/11 from the Israeli Science Foundation (ISF).

References

- [1] Davis, D., Monrose, F.: On user choice in graphical password schemes. In: Proceedings of the 13th USENIX Security Symposium. USENIX (August 2004)
- [2] Dhamija, R., Perrig, A.: Dèjà vu: a user study using images for authentication. In: Proceedings of the 9th conference on USENIX Security Symposium, vol. 9. USENIX Association, Berkeley (2000)
- [3] Herzberg, A., Margulies, R.: Forcing Johnny to Login Safely. In: Atluri, V., Diaz, C. (eds.) ESORICS 2011. LNCS, vol. 6879, pp. 452–471. Springer, Heidelberg (2011)
- [4] Jakobsson, M., Yang, L., Wetzel, S.: Quantifying the security of preference-based authentication. In: DIM 2008: Proceedings of the 4th ACM Workshop on Digital Identity Management, pp. 61–70. ACM, New York (2008)
- [5] Karlof, C., Tygar, J.D., Wagner, D.: Conditioned-safe ceremonies and a user study of an application to web authentication. In: SOUPS 2009: Proceedings of the 5th Symposium on Usable Privacy and Security (2009)
- [6] Margulies, R.: Usable and phishing-resistant authentication mechanisms. Master's thesis, Bar-Ilan University (July 2011), <http://submit2.cs.biu.ac.il/WAPP/thesis.pdf>
- [7] Rabkin, A.: Personal knowledge questions for fallback authentication: security questions in the era of facebook. In: SOUPS 2008: Proceedings of the 4th Symposium on Usable Privacy and Security, pp. 13–23. ACM, New York (2008)
- [8] Schechter, S., Brush, A.J.B., Egelman, S.: It's no secret. measuring the security and reliability of authentication via 'secret' questions. In: SP 2009: Proceedings of the 2009 30th IEEE Symposium on Security and Privacy, pp. 375–390. IEEE Computer Society, Washington, DC (2009)
- [9] Shepard, R.N.: Recognition memory for words, sentences, and pictures. *Journal of Verbal Learning and Verbal Behavior* 6(1), 156–163 (1967)
- [10] Suo, X., Zhu, Y., Owen, G.S.: Graphical passwords: A survey. In: Proceedings of the 21st Annual Computer Security Applications Conference, pp. 463–472. IEEE Computer Society, Washington, DC (2005)
- [11] Vu, K.-P.L., Proctor, R.W., Bhargav-Spantzel, A., Tai, B.-L.B., Cook, J., Eugene Schultz, E.: Improving password security and memorability to protect personal and organizational information. *Int. J. Hum.-Comput. Stud.* 65(8), 744–757 (2007)
- [12] Zviran, M., Haga, W.J.: User authentication by cognitive passwords: an empirical assessment. In: JCIT: Proceedings of the Fifth Jerusalem Conference on Information Technology, pp. 137–144. IEEE Computer Society Press, Los Alamitos (1990)
- [13] Passfaces™: Graphical password technology, <http://realuser.com/> (last visited December 2011)

Balancing Security and Usability of Local Security Mechanisms for Mobile Devices

Shuzhe Yang and Gökhan Bal

Chair of Mobile Business and Multilateral Security
Goethe University Frankfurt am Main, Germany
{shuzhe.yang,goekhan.bal}@m-chair.net
<http://www.m-chair.net>

Abstract. The loss of control over a new-generation mobile device (e.g. loss of device or short time of inattention) can have negative impacts on the owner's privacy due to the increasing number of privacy-sensitive data stored on such devices. Current mobile platforms either lack the required protection mechanisms or the implementations lack a balance between the level of security and usability. In order to fill this gap, we propose a design for a local security mechanism for mobile devices by using an reasonable combination of existing technologies.

Keywords: Privacy, Biometrics, Usable Security, Mobile Platform Security, Trustworthy User Devices.

1 Introduction

Recent developments in the domain of mobile technology caused a significant change in the prominence of mobile devices in peoples' daily lives. Users not only trust their devices to provide them with classical communication capabilities such as phone calls or short messaging. Instead, they trust new-generation devices to store and managed a significant amount of privacy-sensitive data such as photos, location information, browser credentials, or the media library. Furthermore, one property of these platforms is their openness for third-party applications. In order to enable innovative personalized services, these applications are provided with access to those data. The success of application stores for mobile platforms substantiate the perceived usefulness of these applications by users. Thus, mobile devices have turned into personal assistants in daily life and are capable of revealing a lot about the habits and activities of their users.

The downside of this development is the increased risk for misuse. A loss of a mobile device would not only cause financial losses (e.g. through unauthorized phone calls), it could furthermore lead to a critical invasion into users' privacy since an attacker would have access to a significant amount of privacy-sensitive information. Thus, two relevant attack scenarios to protect users from are (1) the loss of the device (e.g. through theft) and (2) short period of inattention that could be exploited by an attacker. As countermeasures for these attack scenarios, device and platform manufacturers introduced a series of mechanisms such as

PIN-protected access to the device user interface. But with the increasing functionality of mobile devices grows the complexity of their usage. This necessitates the consideration of user's capabilities to use such a device. Current mobile platforms either lack the required protection mechanisms or the implementations lack a balance between the level of security and usability. Moreover, there are still deficits in existing approaches concerning the integration of technologies to achieve a balance between usability and security. In this paper, we bridge this gap by presenting a conceptual design of a local security mechanism. As a first step, we establish a set of evaluation criteria based on the global goals of security and usability. These criteria lead to a set of design decisions that will serve as the basis for the design.

The structure of the paper is as follows. Chapter 2 will provide an overview on related work in practice and research. Chapter 3 first defines the two relevant attack scenarios in more detail and classifies them according to existing classification schemes for mobile security attacks. Furthermore, frameworks for security and usability evaluation will be established. Chapter 4 first presents the set of design decisions and then introduces our proposed design for a local security mechanism. Chapter 5 summarizes this work and concludes with limitations of this research and gives hints for potential future work directions.

2 Related Work

Research in the field of mobile security experienced a significant growth in the last years. This is due to new manifold challenges that new-generation mobile technology poses to research and development. In order to expedite advancements in this field, several efforts to substantiate this research domain have been performed. In a survey paper, Becher et al. (2011) provide a classification framework for the different aspects of mobile security vulnerabilities [16]. This framework comprises of the categories of *hardware-centric attacks*, *device-independent attacks*, *software-centric attacks*, and *user layer attacks*. One prominent attack scenario related to hardware-centric attacks they present is *Forensics Analysis*. In this attack scenario the attacker's target is to break the confidentiality of the stored data. As countermeasures the authors propose encryption of the non-volatile memory of the device and to use a secure store for secure keys (e.g. using a Trusted Platform Module (TPM)). Another taxonomy for security threats to mobile computing is presented by Friedman and Hoffman (2008) [17]. Security threats are grouped into the categories 1. Malware, 2. Phishing and Social Engineering, 3. Direct Attack by hackers, 4. Data communications interception and spoofing, 5. Loss and theft of devices, 6. malicious insider actions, and 7. user policy violations. As countermeasure, the authors propose data encryption, backup and recovery mechanisms, and device control.

Several countermeasures have been developed in research and practice to cover different aspects of mobile security and especially for the considered attack scenarios. In Nicholson et al. (2006), the authors propose a solution for the device loss scenario by introducing the concept of *Transient Authentication*. Transient

authentication lifts the burden of authentication from the user by using a wearable token that constantly attests to the user's presence [18]. When the user departs, the token and device lose contact and the device secures itself. The main drawback of this multi-factor authentication approach is the dependence on a second device. If the user loses or forgets that token, access to his data cannot be granted or can only be granted with additional efforts. Another feature that is included in some of the new-generation mobile platforms is *Remote Wipe* which allows users to delete data remotely. This solution depends on the ability to locate the device via the Global Positioning System (GPS) or the mobile cellular network. Thus, an attacker could simply disconnect positioning features. Further approaches use *implicit authentication* mechanisms to enhance the usability of security. In this approach, different behavioural patterns of humans are exploited to identify the user [20]. Other approaches use biometric data or the combination of different biometric data to authorize the user [7].

3 Requirements and Evaluation Framework

This section first describes the attack scenarios that are in focus of this research. Furthermore, it derives relevant requirements to protection mechanisms that will serve as the basis for the development of our proposed design. Based on these requirements, we set-up an evaluation framework, considering both usability and security aspects.

3.1 Attack Scenarios and Impact

The two relevant attack scenarios that we tackle are: *loss of mobile device* and *short time of inattention*. The common ground of these two attack scenarios is the mobile device owner's loss of physical control over the device. Thus, these attack scenarios can be categorized under the class of *hardware-centric attacks* according to the classification scheme of Becher et al. (2011) [16]. Regarding the taxonomy of Friedman and Hoffmann (2008), the threat category covering our attack scenarios is *loss and theft of device* [17]. The relevance of these two scenarios is strongly related to the unique property of mobility. In consequence of the small size of mobile devices, the risk of device theft or loss is high. Without proper protection mechanisms, an attacker can easily communicate in the name of the device owner or manipulate data on the mobile device. The main distinction point between the two scenarios is the available time for an attack. If the mobile device is stolen, the attacker has an unlimited amount of time to break the security mechanism. In the case of inattention, the attacker has a limited time frame to break the system. The main goal of a security system that protects the user in these two scenarios is to keep the confidentiality of the user. Thus, unauthorized access to the data and manipulation of data must be prevented. Another important protection goal is related to accountability. The attacker must not be able to transmit data or to use the communication capabilities of the device.

3.2 Evaluation Criteria

Based on our primary goal to achieve a balance between security and usability, we developed a set of evaluation criteria. This evaluation framework comprises of the two parts *security* and *usability*. Each of those parts is further divided into several aspects. This evaluation framework can be used to evaluate existing and future security mechanisms to protect against the two attack scenarios. In the following, the evaluation criteria are presented in detail. Table 1 further maps each criterion to the IT security protection goals that are affected.

Security Evaluation

- *User Interface Access Control*. A simple security mechanism that authenticates the user to unlock the user interface.
- *Access over Application Programming Interface (API)*. The security mechanism must consider the possibility that an attacker can install third-party applications, which use bugs to access data stored on the mobile device.
- *Self-Security*. The security mechanism itself must be protected, which means that an unauthorized deactivation should not be possible.
- *Direct Storage Access*. This attack requires to open the mobile device. If another device mounts the storage of the mobile device, security mechanisms are not loaded. In consequence, the attacker can extract stored data (only relevant to *loss of device* scenario, because this attack requires much time).
- *Recovery*. The possibility to find the mobile device after a loss to cut the financial loss (only relevant to *loss of device* scenario).

Usability Evaluation

- *Simplicity of Setup*. If the setup is too complex or the purchase price is too high, it might deter users from using it or leads to misconfiguration.
- *Efforts for Maintenance*. The aspect *maintenance* contains for example additional infrastructure, which needs administration effort.

Table 1. Evaluation Framework for Security

Scenario	Aspects of Security Mechanism	IT Protection Goals
Loss of Mobile Device	User Interface Access Control	Confidentiality, Authenticity, Integrity
	Access over API	
	Direct Storage Access	
	Recovery	Accountability
	Self-Security	-
Inattention	User Interface Access Control	Confidentiality, Authenticity, Integrity
	Access over API	
	Self-Security	-

Table 2. Evaluation Framework for Usability

Usability		Aspects of Usability
Comfort	Simplicity of Setup	Efficiency
	Maintenance	Efficiency, Satisfaction
	Frequency & Effort of Credential Request	
	Frequency of Decision Request	
	User Comfort after an attack	Efficiency, Effectiveness
Cost	One-time costs	Efficiency
	Running costs	Efficiency, Satisfaction
	Transaction-based costs	Efficiency, Effectiveness

- *Frequency & Effort of Credential Request.* Evaluates how often the user has to enter credentials and how many resources – physical or mental load, time, monetary or material costs [2] – are required to perform this task (e.g. entering a complex but secure password vs. entering a PIN).
- *Frequency of Decision Request.* Some security applications let the user decide which data should be stored securely. This decreases the usability.
- *User comfort after an attack.* Evaluates the complexity and number of steps to activate the security mechanism after an attack.
- *One-time costs.* E.g. price for purchasing software.
- *Running costs.* E.g. monthly fee for software usage.
- *Transaction-based costs.* E.g. fees for activating the security mechanism.

According to the International Organization for Standardization (ISO) Norm 9241-11 [2] a security mechanism is usable, if it fulfils the aspects of *efficiency*, *effectiveness* and *satisfaction*. Table 2 maps each evaluation criteria to these aspects of usability.

4 Design of a Security Mechanism

This chapter presents the proposed design for a security and usability-balanced protection mechanism for mobile devices. Based on the evaluation criteria presented in Section 3 we first make a set of design decisions that will affect the shaping of our design.

4.1 Design Decisions

Design Decision 1: One level of protection for both types of privacy-sensitive data.

Two protection levels decreases the usability because either the user would have to categorize each data item or a policy-based mechanism would have to be introduced to automate the categorization of the data items on the device. A level of protection that is appropriate for direct identifiable data will also be suitable for indirect identifiable data.

Design Decision 2: Fingerprint recognition as authentication method

Due to the common limitations of mobile devices (small size, etc.), biometric authentication will provide the highest usability on mobile devices. Fingerprint, iris and face recognition are appropriate mobile authentication methods (cf. [6,7,8,9,10]). Fingerprint recognition has the best efficiency/price ratio and is therefore the best candidate for mobile devices. Token-based authentication as proposed in [8] is possible but unsuitable since the user must carry the token and that involves the risk of losing it.

Design Decision 3: Hardware-based full disk encryption

Encryption technologies are used to prevent the possibility of installing third-party applications and extracting data with *direct storage access* attacks. A solution should implement a hardware-based full disk encryption because of several reasons. A hardware-based encryption enables higher performance than software solutions, which increases the usability and user's workflow is not affect and stronger encryption algorithm can be used to enhance security.

Design Decision 4: Separate hardware storage for credentials

A separate and secure credential storage is needed because credentials stored in the local memory can be attacked by malware using bugs in the operating system or in applications. Using a hardware credential storage enhances the protection of credentials and increases the trustworthiness.

Design Decision 5: Secure boot process

Another attack possibility is compromising the Master Boot Record by installing malware [11]. Thus, the proposed design should implement countermeasures to prevent this attack.

4.2 Entities and Used Technologies

Considering the design guidelines and the design decisions presented in the previous sections, this section presents the entities of our proposed design. Furthermore, in order to support implementations, we present existing technologies and mechanisms that can be adopted to implement the respective entities.

MTM. The Mobile Trusted Module (MTM) is a aligned version of the TPM for mobile devices [19]. In our design the MTM serves as secure credential storage and is used for platform integrity verification (e.g. for a secure boot process). The MTM could also be used as encryption module, but for performance reasons we do not make use of this capability [13,14].

Encryption Module. For encryption purposes, an encryption module such as implemented in Apple's iPhone 3GS, is required [1]. Today's mobile devices' processors usually do not have encryption accelerating instruction sets. As mentioned earlier, using an MTM would not meet our performance requirements.

Fuzzy Cryptography. Fingerprint-based authentication requires the creation and storage of a fingerprint template on the device, more specifically on the MTM. We recommend to use fuzzy cryptography which introduces a biometric cryptographic system, which verifies the fingerprint with error-redundant functions and corrects potential disturbances [4]. Fuzzy cryptography makes brute-force attacks [3,4,5] and duplication of credentials difficult. Trivedi and Seshadri (2011) present a method to generate and derive the encryption key from biometric data (e.g. fingerprint) [15].

Capacitive Touch Screen and Optical Sensors for Fingerprint Recognition. Capacitive touch screens, which are mostly built in mobile devices, can be used for scanning fingerprints in combination with optical sensors [10]. It uses fourier enhancement algorithms and rank-order transformation to increase the quality of the scanned fingerprint.

RIM Certificates . As countermeasure for attacks that compromise the Master Boot Record, Dietrich and Winter (2008) present a solution that uses Reference-Integrity-Metric (RIM) certificates to measure the integrity of the boot software [12]. RIM certificates store reference integrity values of boot-relevant services in an uncompromised state (e.g. delivery state). During the boot process, software integrity is measured and compared to the RIM certificates. The boot process will be stopped, if the integrity is compromised. By using RIM certificates and integrity verification of all unencrypted components within a mobile device, the trustworthiness can be ensured.

4.3 Process Description

In this section we describe how the used technologies and entities can be combined in order to provide the user with an effective local security mechanism. Figure 1 depicts a component diagram with the used components. The essential processes are described in the following.

Setup. The setup process is depicted in Figure 2. The process begins with the user putting his finger on the touch screen for the initial fingerprint scan. Two credentials will be derived from the scanned data. First, a hash value will be created using fuzzy cryptography, second an encryption key will be generated that will be encrypted with the Storage Root Key (SRK) and stored in the key storage of the MTM. The hash value will then be stored in the Platform Configuration Register (PCR) of the MTM.

Encryption. After the hash value and encryption key have been created, the encryption process starts (Figure 3). The first step is decrypting the encryption key with SRK. After that, two encryption processes are needed. First, data that is already on the device (e.g. mobile operating system) will be encrypted in-place

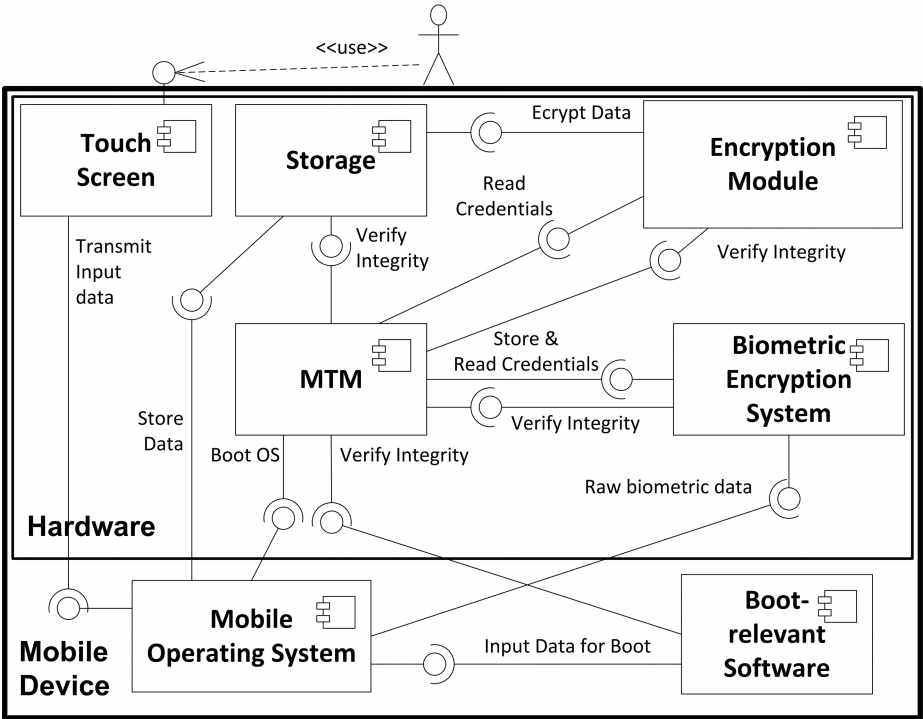


Fig. 1. Component Diagram of the Design

in the background. Second, the empty space on the storage must be encrypted in order to prepare it for future usage and to prevent unauthorized installation of third-party applications. In both processes the file system is also encrypted, which means that the encryption layer is between file system and the physical storage. In consequence, if the storage is connected to another device, the file system will not be visible to the platform.

Boot and Authentication. As stated before, the mobile device contains the required hardware hash values and RIM certificates for the secure boot process. When the user starts the mobile device, the biometric encryption system immediately verifies the integrity of the hardware and boot-relevant services with the

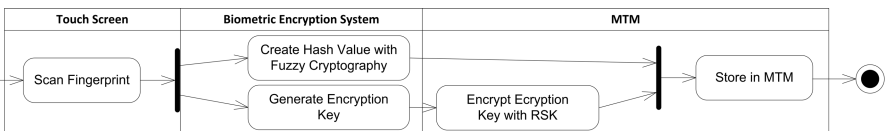


Fig. 2. Setup Process

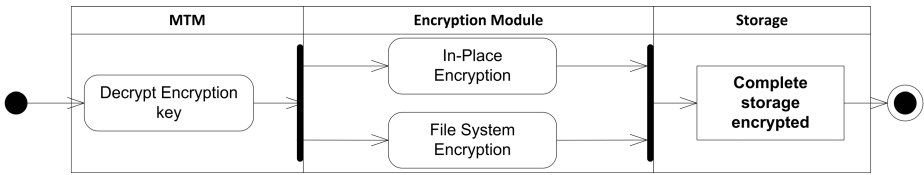


Fig. 3. Encryption Process

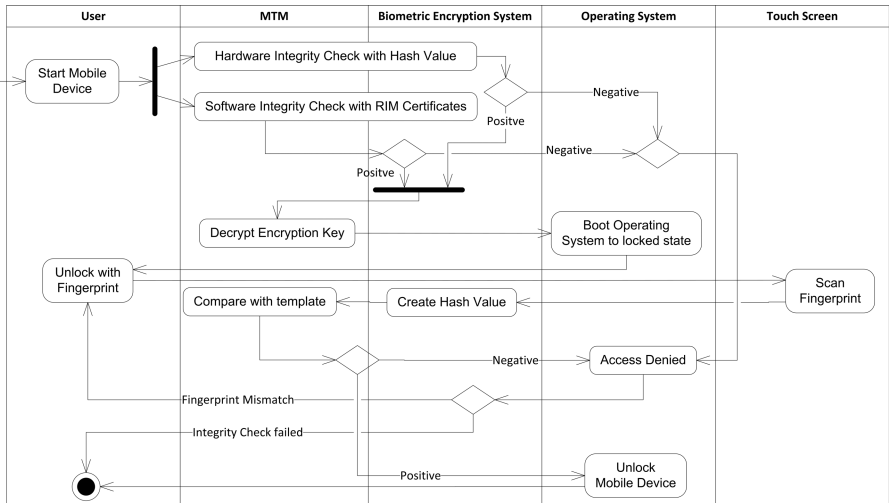


Fig. 4. Boot and Authentication Process

hash values and RIM certificates stored in MTM. Only in case of a mismatch between the measured values and the stored values, the operating system stops the boot process. Otherwise, the boot process will continue. After the boot process, the user interface to the mobile operating system is locked. To unlock the device, the user needs to put his finger on the touch screen for fingerprint-based identity verification. The touch screen scans the fingerprint and transfers the data to the biometric encryption system which then generates a hash value of the scan. In the next step, this hash value is compared with the stored hash value template in the PCR of the MTM. If the hash value does not match, the access is denied. If they match, the mobile device will be unlocked and the user has access to the interface and data. This process is depicted in Figure 4.

4.4 Evaluation

This section provides a security and usability evaluation of the proposed system based on the evaluation framework presented Chapter 3.

Security

- *User Interface Access Control.* Based on the low EER, the acceptance of an unauthorized person’s fingerprint is unlikely [10]. A fake of the fingerprint is difficult because capacitive touch screens recognize the electronic charge of the finger which avoids attacks based on fingerprint imitations.
- *Access over API.* These attacks are prevented by the RIM certificate-based integrity checks. Hardware modification and/or installing malware in boot-relevant software is difficult with this system.
- *Direct Storage Access.* This class of attacks is prevented by the full disk encryption.
- *Self-Security.* A deactivation of the security mechanism is difficult due to the components being implemented as hardware.
- *Recovery.* Our proposed system does not support recovery. Thus, other solutions must be used in parallel for supporting recovery.

These evaluation results hold for the device loss scenario as well as for the inattention scenario.

Usability

- *Simplicity of Setup.* Compared to current implementations, our setup process requires a little bit more effort. But since this task has to be performed only once, it still is in an acceptable level.
- *Maintenance.* The design does not require any significant maintenance efforts after setup.
- *Frequency & Effort of Credential Request.* Compared to current unlock mechanisms based on passwords or gestures, our approach is more comfortable in daily usage. On the one hand, the user does not need to remember passwords and on the other hand today’s fingerprint-based identity verification technologies are fast and reliable.
- *Frequency of Decision Request.* After setup, the user does not need to make any decisions. The services run in the background.
- *User Comfort after an Attack.* The comfort is on a high level because all data stored on the mobile device is safe and do not need the user to interfere.
- *Costs.* We propose to implement the security mechanism into the mobile operating system and the mobile device, which means it is cost neutral for the user.

5 Conclusion

The growing privacy-sensitivity of smartphones pose high requirements to protection mechanisms. To this day, several concepts and technologies have been developed to enhance users’ trust into these devices. Nevertheless, they failed to provide appropriate concepts that on the one hand provides protection and on the other hand considers users’ habits for using new-generation devices. In

this paper, by exploiting the capabilities of existing security technologies, we presented a new design for a holistic security mechanism to protect the user from two prominent security attacks on mobile devices: device loss and inattention. The goal was to provide a design that explicitly can be implemented with today's existing technology and to consider users' usability experiences with new-generation smartphone. The novelty lies in the combination of those technologies to a holistic design. Our design is based on a requirements analysis regarding the aspects of security and usability. We derived a set of design decisions that we considered essential to achieve the expected level of security and usability.

The main limitation of our work is that we do not provide an practical evaluation of the proposed design. An evaluation of an implementation of the system in terms of performance and usability should be the next steps to prove the effectivity of the system. Another limitation is that it partly builds on technology which is not widespread (e.g. only a little number of devices with an MTM exists). Thus, even if we build only on existing technology, due to the limited diffusion of MTMs an implementation of our system may not be possible yet. Still, the proposed solution shows the direction to which mobile security research and development should go. It is essential for mobile security technologies to keep up with the developments in the usability of new-generation smartphones. Thus, research has to figure out how to provide existing technologies to the users in a way that does not disturb the user experience.

References

1. Morrissey, S., Campbell, T.: IOS forensic analysis. Apress, Berkeley (2010)
2. ISO: ISO 9241-11:1998-03, Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability 35.180; 13.180(9241-11:1998-03). ISO, Geneva (1998)
3. Buhan, I., Kelkboom, E., Simoens, K.: A Survey of the Security and Privacy Measures for Anonymous Biometric Authentication Systems. In: International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP 2010), pp. 346–351. IEEE Press, New York (2010)
4. Merkle, J.: Biometrie-Daten-Schutz. Funktionsprinzip und Chancen biometrischer Kryptosysteme. In: KES 2008, vol. 6, p. 52. SecuMedia-Verlag, Ingelheim (2008)
5. Cavoukian, A., Stoianov, A.: Biometric Encryption. Positive-Sum Technology that Achieves Strong Authentication, Security and Privacy. Discussion paper of the Office of the Information and Privacy Commissioner of Ontario (2007)
6. Park, K.R., Park, H., Kang, B.J., Lee, E.C., Jeong, D.S.: A study on iris localization and recognition on mobile phones. EURASIP Journal on Advances in Signal Processing, vol. 2008 (2008)
7. Tao, Q., Veldhuis, R.N.J.: Biometric Authentication for a Mobile Personal Device. In: Third Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services, pp. 1–3. IEEE Press, New York (2006)
8. Furnell, S., Clarke, N., Karatzouni, S.: Beyond the PIN – Enhancing user authentication for mobile devices. In: Computer Fraud & Security 2008, vol. 8, pp. 12–17. Elsevier Science Inc., New York (2008)

9. Abileah A., Green P.: Optical sensors embedded within AMLCD panel: design and applications. In: Association for Computing Machinery (ACM) (eds.) *Images and Beyond: the Future of Displays and Interaction*, article no. 27. ACM, New York (2007)
10. Marcialis, G.L., Roli, F.: Fingerprint verification by fusion of optical and capacitive sensors. *Pattern Recognition Letters* 25(11), 1315–1322 (2004)
11. Schmidt, J.: Krypto für Jedermann. Richtig verschlüsseln mit Linux. Verschlüsselung unter Linux. In: *c't - Magazin für Computertechnik*, vol. 11, pp. 192–195. Heise Verlag, Hannover (2011)
12. Dietrich, K., Winter, J.: Secure Boot Revisited. In: Wang, G. (ed.) *The 9th International Conference for Young Computer Scientists*, pp. 2360–2365. IEEE Press, New York (2008)
13. Cesena, E., Löhr, H., Ramunno, G., Sadeghi, A., Vernizzi, D.: Anonymous Authentication with TLS and DAA. In: Acquisti, A., Smith, S.W., Sadeghi, A.-R. (eds.) *TRUST 2010*. LNCS, vol. 6101, pp. 47–62. Springer, Heidelberg (2010)
14. Kursawe, K., Schellekens, D., Preneel, B.: Analyzing trusted platform communication. In: *ECRYPT Workshop, CRASH – Cryptographic Advances in Secure Hardware* (2005)
15. Raghav Trivedi, T., Seshadri, R.: Efficient Cryptographic Key Generation using Biometrics. *International Journal of Computer Technology and Applications* 2(1), 183–187 (2011)
16. Becher, M., Freiling, F.C., Hoffmann, J., Holz, T., Uellenbeck, S., Wolf, C.: Mobile Security Catching Up? Revealing the Nuts and Bolts of the Security of Mobile Devices. In: *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 96–111. IEEE Press, New York (2011)
17. Friedman, J., Hoffman, D.V.: Protecting Data on Mobile Devices: A Taxonomy of security threats to mobile computing and review of applicable defences. *Information Knowledge Systems Management* 7(1,2), 159–180 (2008)
18. Nicholson, A.J., Corner, M.D., Noble, B.D.: Mobile Device Security Using Transient Authentication. *IEEE Transactions on Mobile Computing* 5(11), 1489–1502 (2006)
19. Trusted Computing Group: *TCG Mobile Trusted Module Specification. Specification Version 1.0 Revision 7.02* (2010)
20. Shi, E., Niu, Y., Jakobsson, M., Chow, R.: Implicit Authentication through Learning User Behavior. In: Burmester, M., Tsudik, G., Magliveras, S., Ilić, I. (eds.) *ISC 2010*. LNCS, vol. 6531, pp. 99–113. Springer, Heidelberg (2011)

Analyzing Value Conflicts for a Work-Friendly ISS Policy Implementation

Ella Kolkowska¹ and Bart De Decker²

¹Örebro University School of Business, Sweden

²KU Leuven, Dept. of Computer Science, IBBT-DistriNet, Belgium
ella.kolkowska@oru.se, bart.dedecker@cs.kuleuven.be

Abstract. Existing research shows that the Information Systems Security policies' (ISSPs) inability to reflect current practice is a perennial problem resulting in users' non-compliant behaviors. While the existing compliance approaches are beneficial in many ways, they do not consider the complexity of Information Systems Security (ISS) management and practice where different actors adhere to different and sometimes conflicting values. The unsolved value conflicts often lead to unworkable ISS processes and users' resistance. To address this shortcoming, this paper suggests a value conflicts analysis as a starting point for implementing work-friendly ISSPs. We show that the design and implementation of a work-friendly ISSP should involve the negotiation for different values held by the different actors within an organization.

Keywords: value conflicts, ISS compliance, policy implementation.

1 Introduction

Employees' poor compliance with ISS policies is a perennial problem for many organizations [1]. Previous research points out various reasons for this problem but leading amongst them are the inability of the policy to reflect current practices [2] and users' resistance to security rules [3]. Recent literature [e.g. 4] suggests that value correspondence between ISS values and employees' values might solve the problem of lack of compliance. Hence many of the suggested approaches focus on changing users' values to align them with ISS values included in the ISSP [e.g. 5]. The consequence of this narrow end-user focus is that values included in ISSP are seldom questioned and the larger context of organizational factors influencing employees' values and behaviors is not considered in the design and implementation of ISSPs [e.g. 6, 7]. We regard the lack of sufficient consideration of the current practice in the design and implementation of ISSPs as a considerable shortcoming, especially in light of the central role that ISSPs play in managing the users' ISS behaviors.

ISSPs are often expressions of management's values and beliefs about how to manage ISS in the organization [8] while employees' behaviors are often anchored in deeply held values and beliefs often related to their profession and work practices [9, 10]. Previous research shows that tensions and value conflicts are a natural part of

ISS management [9-11]. We claim that the current limited focus on changing users' values do not consider the complexity of ISS management and practice where different groups and collaborating actors adhere to different and sometimes even conflicting values. We argue that the design and implementation of a *work-friendly ISSP* should begin with the analysis of existing value conflicts and involve a negotiation of the different values held by different actors within an organization.

The purpose of this paper is to show how value conflict analysis might be used as a starting point for implementing a work-friendly ISSP. Our findings also suggest how managers can use value conflict analysis as a starting point for finding workable ISS procedures and processes.

2 ISSP Implementation and Value Conflicts

Security policies and codes of conducts are often the main and most important tool used by managers to guide and control employees' security behaviors. However little research has explored how good ISSP are designed and implemented [12]. On the contrary, previous research shows that ISSPs are often developed and implemented through a top-down approach, using international standards, without sufficient consideration for the daily work practice [13, 14], resulting in employees' resistance and non-compliance [3].

To deal with the problem, many behavioral compliance approaches emphasize the importance of value correspondence between users' values and the values included in the ISSP. According to the literature value correspondence might be achieved by cultivating a security culture [5, 15] and setting up awareness programs [16, 17]. The focus in these approaches is on making users internalize ISS values in their daily work practices [5] and in this way achieve compliance with the implemented ISSPs [14, 15].

However, if the implemented policies cannot reflect the current practice and result in unworkable security procedures, it will be difficult to achieve value correspondence and to align users' values with implemented ISS values. When forced to choose between ISS values and work values, employees will often choose to have the work done over security. Value conflicts and prioritizations are thus a natural part of security in practice.

In the context of ISS, many different actors and values are involved [10]. These different collaborating and communicating actors such as IT-technicians, management and users may adhere to different and sometimes conflicting values in the design, implementation and use of ISSPs. For example, Kolkowska [18] found that, based on the 'academic freedom' value system, university employees use their work computers for private business even if the ISSP says that the computers are to be used for work activities only. An additional example is found in Ruidhaver et al. [7]. The authors describe how account agents share usernames and passwords though it is forbidden according to the ISSP. The account agents' day-to-day operations involve maintaining customer records and they often have to share passwords to access particular customer accounts. The agents violate the ISS policy to get their job done [7]. The

previous literature also illustrates various examples of value conflicts in relation to ISS in health care [10, 11] and in relation to risk management [19].

Although the notion of value conflicts in relation to ISS has been well documented in the literature, a limited number of current studies emphasize the relationship between value conflicts and the ISSP implementation. While aspects of value conflicts in relation to compliance have been touched upon in the work of Hedstrom et al. [10], they do not explicitly focus on how the analysis of value conflicts may be used in the design and implementation of work-friendly ISSPs. Our study addresses this gap showing the opportunities of systematic value conflicts analysis to attain a work-friendly ISSP implementation and ISS processes integrated in the daily work practice.

3 Theory and Methodology

Values are considered to be motivations for all human acting i.e. values decide what action is relevant to take for a person in a certain situation [20]. According to value theory [20], the value concept is unspecific and quite challenging to study. Values are subjective and often unconscious and sometimes difficult to articulate [21]. Hence, focusing on only visible and espoused values might be misleading in understanding why people behave in certain way.

Shein's model of organizational culture [21] helps to find values that exist on different levels in an organization. The model has earlier been used in studies of ISS culture [5, 14]. We chose to use Shein's model in our study because of its ability to elicit both espoused values and basic assumptions. Such distinction is important because ISS actions such as the implementation of ISSPs as well as users' ISS behaviors might differ from the values that are espoused. Focusing exclusively on users' espoused values could be misleading because, in the context of ISS, users often espouse what they are supposed to do and not what they actually do [6]. Correspondingly, values espoused in an ISSP are high level ISS values which are then implemented in actual ISS procedures [22]. The implemented procedures might fulfill other values than those espoused in the ISSP.

Schein [21] argues that in order to understand what values impact people's behaviors, values should be studied on three levels: (1) artifacts, (2) espoused values, and (3) basic assumptions. *Artifacts* in the context of ISS are related to implemented security measures and processes [5, 14] and also to employees' security behaviors. Artifacts are related to the question '*how things are done*' in an organization. *Espoused values* relate to values and norms expressed within an organization. In relation to ISS, espoused values can be found in an ISSP [14] and they can also be expressed by people in the organization. Espoused values are related to the question '*what is important in the organization*'? *Basic assumptions* are the basic underlying beliefs and values that are often subconscious. Basic assumptions directly impact the artefact level resulting in the observable behavior of employees in their daily work activities [21]. Basic assumptions are related to '*why things in an organization are done in this specific way*'.

3.1 Case Background

The case study was carried out at one of the Swedish Municipality Social Service Divisions, which is responsible for helping vulnerable children and their families. To improve ISS in the organization, the management decided to implement a computer-based IS for all communication and exchange of information. According to this decision all actor groups working in the municipality's social services were obliged to use the implemented IS. It was argued that in this way, ISS in social services would be improved since the IS realized all the prescribed ISS rules and legal requirements. Though all actor groups had to comply with the decision, it was noticed that the care providers at treatment centers did not comply with the new rules. As a result, other actor groups could not access up-to-date information and consequently, their job performance suffered and the managers were concerned about the confidentiality of information and the privacy of the clients.

3.2 Research Methodology

The study was conducted as a qualitative case study [23]. Data was collected in two phases. In the first phase, interviews and project documents from another project [24] were reviewed to create an understanding of the context for the ISS work and to identify relevant actor groups for studying value conflicts. Sixteen group interviews with eight different stakeholder groups (including management, system owners, IT-technicians and five user groups) and five project reports were reviewed. Three stakeholder groups emerged as essential to study value conflicts in relation to the ISSP implementation and use. These were (a) managers and (b) IT-technicians who enforced the new security rules and processes through the information system and (c) care providers at the treatment centers who did not comply with these rules. We also identified '*legal security*' as an important context for ISS in social services. We found that all interviewed people stressed that ISS is important in order to achieve '*legal security*'. Therefore in the second phase we looked for ISS values and work values in the context of '*legal security*'.

In the second phase additional documents (policies, guidelines and work descriptions) were reviewed and in-depth interviews with the three stakeholders groups were conducted. Our data collection was guided by the three levels in Shein's model to values impacting the implementation and use of the ISSP. The semi-structured interviews were based on the following questions: 'how things are currently done to achieve legal security for the clients' as a way to identify artifacts, and 'what values are important to achieve legal security' in order to uncover the espoused values. Basic assumptions were derived from artifacts during the interviews and the analysis by asking 'why things in the organization are done in this specific way'. Each interview lasted approximately 1 to 2 hours. All interviews were recorded and transcribed.

The data was analyzed in four stages. In the *first stage*, espoused ISS values and work values were identified based on the documents and interviews. The analysis resulted in a long list of statements that were then categorized in clusters according to an inductive qualitative categorization [25]. Thereafter, clusters of values were

labeled. The emerging clusters were further validated through discussions with other researchers and representatives from the studied organization. During the *second stage*, security structures and processes as well as care providers' security behaviors (artifacts) were analyzed in order to identify basic assumptions. Basic assumptions could be discovered in the collected data, when the interviewees explained the reasons behind the behaviors and the reasons behind the implemented ISS processes. The identified basic assumptions were then categorized in the same way as espoused values (cfr. stage 1). In *stage three*, the identified espoused values and basic assumptions were analyzed in the context of 'legal security'. The analysis resulted in a conceptual model illustrating ISS values and social workers' values in the context of 'legal security' [see 26]. Once the conceptual model was created, an expert panel consisting of managers and social workers validated and discussed the categories and the relationships between these categories, which led to a few changes in the model. In the last and *fourth stage*, the value categories in the model were compared in order to find value conflicts explaining non-compliant behaviors.

4 Value Conflicts in Social Services

In this section, we present identified value conflicts and their implications on the ISSP use in practice. The analysis of value conflicts is based on artifacts, espoused values and basic assumptions identified in relation to the implementation and use of ISSP in social services. The last section suggests how managers can use value conflict analysis as a starting point for finding workable ISS procedures and processes.

4.1 ISS Values in the Context of Social Work

As mentioned earlier, 'legal security' was the value that had to be achieved in public social services in relation to ISS; thus, values were studied in this context. 'Legal security' is achieved by *secure work processes* and *good ISS*. Secure work processes are based on clarity, transparency, consistency and respect, which means for instance that the clients always know what happens and why, and that decisions are based on correct information that contains both facts and well founded logical interpretations. Good ISS means that information is available, and that the information is correct and protected against unauthorized disclosure (confidentiality) and change (integrity). It also means that it is possible to trace information and users in the system.

We found that *integrity* of the information was an unclear and confusing concept in social services. For managers and IT-technicians, integrity meant protection of information against unauthorized changes, while for care providers, integrity meant that the information was right, complete, not changed, detailed and dated. We found also that for managers and IT-technicians, *good ISS* was important to achieve 'legal security'. The two groups argued that the technical solutions implemented in the system would ensure confidentiality, integrity, availability and traceability and therefore also 'legal security for the clients'. For social workers, both *good ISS* and *secure work processes* were equally important in order to achieve 'legal security'. ISS

values were a part of social workers' work values; however, the new processes enforced by the system come in conflict with the care providers' work processes causing value conflicts. The identified value conflicts and their implications on ISS in practice are described in the following sections.

4.2 The Identified Areas of Value Conflicts

Conflict regarding formality and informality (vc₁). Care providers' work is based on spoken communication, conversations and meetings. Before the computer-based system was implemented, care providers had a close relationship with care officers regarding each child/teenager. The two groups frequently discussed the treatment by phone or e-mail. During these informal discussions, it was possible to communicate the emotional and interpretative dimension related to the treatment. After the system was implemented, the communication between the two groups was supposed to take place exclusively through the system. Care providers experienced such communication as insufficient and too formal because the emotional and interpretative dimension was difficult to communicate through the system. One care provider noted:

'We have lost the close relationship with the care officers and the possibility to communicate the emotional and interpretative dimension. We don't want the teenagers to be just dumped here. We want them to understand that the treatment is meaningful and based on cooperation between the different actors! We want them to feel that we really care.'

Because of the conflict regarding the formality of the communicated information, care providers were still using the informal way of communication while the documentation in the system was done infrequently and resulted in a decreased availability and integrity. Also, it was impossible to trace the information and the treatment in the system.

Conflict regarding what information should be communicated (vc₂). The system had been designed with 'simplicity' in mind. However, care providers argued that it was important to communicate the complete picture with all the details to ensure that the decisions are made on well grounded and logical interpretations. One care provider told us:

'At least two care providers have to be present at a meeting with a teenager! It is very important to document all the details ... not only what the teenager says, but also how he/she says it, the body language and so on. Every detail might be important for the interpretation.'

Care providers experienced the documentation in the system as limiting because it required formal reporting (only facts), while their work was based on interpretation and observations. Thus, to ensure sufficient richness of the information, care providers still used detailed paper-based notes to communicate the information within the group and phone conversations or meetings to communicate information to other actor groups. Consequently, data was entered in the system infrequently causing the same problems as in *vc₁*. Information was also documented and stored in different

places (both on paper and digitally) with a significant risk for loss of integrity and confidentiality.

Conflict about responsibility for communicated information (vc₃). Before the system was implemented, care providers were responsible for presenting a rich picture of the situation, while other stakeholders were responsible for choosing the relevant information and formally document it in the records. After the system was implemented, the care providers were responsible for choosing the relevant information for other actor groups and formally document that information in the IS. One of care providers explained:

'I cannot take responsibility for choosing information for other stakeholders. I don't know what they need. If I miss to communicate some important details, maybe a child will not get the help he/she needs!'

Care providers were not properly informed about what information other groups needed to take the appropriate decisions to ensure proper care and treatment. Consequently, different care providers developed different routines for documentation. As a result, other actor groups experienced that some important information was missing or was too detailed. It was also unclear what the facts were and what constitute the care providers' interpretations.

Conflict about why the information is communicated (vc₄). Before the implementation of the system, written documentation (paper notes) was mainly used as a means of communication between care providers working at a treatment center. The most important goal for the documented information was to ensure proper care and treatment. After the implementation of the system, it became unclear what the main goal of the documentation was. The documentation was not seen as part of care providers' work and they did not feel that the documentation in the new form supported their work. It was rather seen as reports to other actor groups. Because of that, documentation was experienced as extra work; social workers did not feel motivated to do it and consequently, documentation in the system was done infrequently and often paper notes were used instead. These cause the same problems as described in *vc₂*.

Conflict regarding how the documentation should be done (vc₅). The new work processes for documentation, enforced by the system, did not correspond to the work processes the care providers were used to. The system did not support the care providers' work and even some important templates and documents were missing in the system. As a result, confidential information was exchanged via insecure portable devices and saved as Word files on local unprotected hard drivers with a significant risk for diminished integrity and confidentiality.

4.3 Reasons for Value Conflicts and Suggested Solutions

Table 1 summarizes the identified value conflict and their implications on the ISS. Based on the analysis, we suggest possible solutions for workable ISS processes. Space limitations, however, prevent us from going into full details.

Table 1. Possible solutions for workable ISS processes

Value conflicts and implications about:	Reasons:	Possible solution(s):
<i>(vc₁) formality versus informality;</i> <i>consequences:</i> infrequent documentation in the system, decreased availability, integrity and traceability.	<i>Procedures and processes are not reflecting work processes:</i> the new processes for documentation do not allow communication of the emotional and interpretative dimension. <i>Confusion</i> about what integrity of information means.	<i>Negotiation</i> between implementers and users to find a way of communicating the emotional and interpretative dimension through the system. <i>Defining</i> what integrity means for the different actors.
<i>(vc₂) what information should be communicated;</i> <i>consequences:</i> infrequent documentation in the system, information stored at different places, decreased confidentiality, availability, integrity and traceability.	<i>Procedures and processes are not reflecting work processes:</i> the system requires information (only facts) while rich information (both facts and interpretations) is needed in work processes. <i>Procedures are unknown and/or wrongly understood:</i> care providers at the treatment centers are not properly educated about what information other groups need to do their job correctly. <i>Confusion</i> about integrity.	<i>Negotiation</i> between implementers and users to find a way to clearly communicate both facts and interpretations through the system. <i>Education</i> focusing on what information the other groups of social workers need and also on risks related to storage of information in different places. <i>Defining</i> what integrity means for the different actors as in <i>vc₁</i> .
<i>(vc₃) responsibility for the communicated information;</i> <i>consequences:</i> care providers develop different routines for documentation; information is missing or is too detailed; unclear what are facts or what are interpretations; decreased availability and integrity.	<i>Responsibilities unclear or wrongly understood:</i> care providers at the treatment centers are not properly prepared for taking the new responsibilities. They are unaware of and confused about what information other groups need. <i>Confusion</i> about integrity.	<i>Education</i> focusing on explaining the meaning of the new processes and responsibilities and also what information the other groups of social workers need. <i>Defining</i> what integrity means for the different actors as in <i>vc₁</i> .
<i>(vc₄) why the information is communicated</i> <i>consequences:</i> documentation in the system is done infrequently; paper notes are used instead;	<i>Procedures and processes are not reflecting work processes:</i> communication through the system does not support communication between care providers at a	<i>Negotiation</i> between implementers and users to find a way to better support communication between care providers at a treatment center.

decreased availability, confidentiality, integrity and traceability.	treatment center. <i>Procedures are unknown and/or wrongly understood:</i> unclear what the main goal of the documentation is.	<i>Education</i> focusing on how the documentation contributes to achieve 'legal security'.
<i>(vc₅) how the documentation should be done;</i> <i>consequences:</i> confidential information is exchanged via insecure portable devices and stored on local disks; decreased integrity and confidentiality.	<i>Procedures and processes are not reflecting work processes:</i> the system does not support the care providers' work.	<i>Negotiation</i> between implementers and users to find a way to change the system so that it supports the care providers' work. <i>Education</i> focusing on risks related to the storage of information on local disks and the usage of portable devices.

5 Discussion

Based on our empirical results, we will highlight four findings.

1. *It is important to consider different values and value conflicts in order to design workable ISS processes.* We found in our case that care providers considered ISS values as important, but equally important were other values related to their profession as social workers. They could not comply with the implemented ISS procedures because it would mean ignoring important professional values. In our case, the implemented ISSP did not reflect the current practice and resulted in unworkable security procedures. Consequently, when forced to choose between ISS values and professional values, care providers chose their professional values. The problem highlighted in the literature is that implemented ISSPs are often based on standards without considering the unique organizational context [13, 14] and most often, ISS is treated as the most important process on top of other processes and not something that should be interwoven in the existing work practice [10]. We argue that it is a significant limitation in current ISS approaches because ISS is only a small part of the users' complex reality. Lamb et al [27] argue that users play different roles in their work context, utilize multiple applications and interact with a variety of other people often in multiple social contexts. Hence, to create workable ISS processes, we need to understand the different values that come into play in relation to ISS and influence users' ISS behaviors. Identifying value conflicts not only helps to explain many of the tensions being experienced in the ISSP implementation and use, but also clarifies the choices that have to be made in any ISSP design or implementation. Based on our empirical findings and the discussion above we conclude that understanding and analyzing value conflicts is important in order to find workable ISS processes.

2. *It is important to realize that values embedded in the ISSP may differ from values guiding the ISSP implementation.* As described in the case study section, the social services organization implemented ISS values included in the ISSP as part of

the computer-based information system. The managers and IT-technicians thought that implementing these values simply meant a careful design of the computer-based system. They argued that if the integrated system was used for communication and exchange of information, it would ensure compliance with the ISSP. It was assumed that employees in social services were both aware of and aligned with ISS values included in the ISSP. It was true, however, that the implementation of these values by the computer-based system clashed with the care providers' work processes and professional values. Hence, although care providers shared the ISS values included in the ISSP, they did not comply with the new ISS processes. As a consequence, the level of ISS security in the organization was decreased.

Based on our empirical findings, we argue that it is important to realize that values embedded in ISSP may differ from values guiding the ISSP implementation. ISSPs are often formulated by top managers and include high level ISS values while the ISSP implementation is performed by IT-technicians [22]. Because of that, there is a risk for a technical focus in the ISSP implementation where the largely socio-technical and organizational context is not considered [28]. This problem also occurred in the studied organization. People responsible for the implementation of the ISSP did not realize the need for establishing new responsibility structures and new process descriptions. As a result, the new procedures and responsibilities were unknown and/or wrongly understood. The findings are based on results from one case study; however, the situation described in this paper is not limited to the specific case study. Today, many organizations implement the existing security rules in form of computer-based systems. For instance, in the health care sector computer-based systems are often implemented to improve the security for medical records. Thus, lessons learned from this case study may help other organizations to avoid security problems experienced in the studied organization.

3 Value conflict analyses may direct management's attention to new security solutions. The studied organization experienced a problem of care providers' non-compliance with the new ISS rules. This lack of compliance decreased the ISS in the organization and also exposed the organization to significant liabilities, which could potentially have an impact on the viability of the enterprise. The most popular approach suggested in the literature on ISS compliance is based on the deterrence theory and emphasizes the use of sanctions to control employees' non-compliant behaviors [29, 30]. In our case, applying this approach would mean that management enforces the new ISS processes by using sanctions. This would create an environment where the care providers would be forced to work according to unworkable ISS processes and violate their professional values or to act according to their professional values with the risk of the ISS rules being ignored. Another stream of security compliance research suggests value correspondence by making users' to internalize ISS values in their daily work practices [5, 15]. Value correspondence can be achieved by cultivating a security culture and setting up awareness programs [16, 17]. In our case, such approach would solve some of the experienced compliance problems. According to our analysis (see section 4.3) some of the new procedures and mechanisms were unknown or wrongly understood. In this case, management needs to create an understanding for the new processes and change some of the care

providers' values by education and suitable awareness programs. We argue that value conflicts analysis may direct management's attention to *why* concordance between ISS rules and ISS practice is not achieved and point at other security solutions that possibly both improve compliance and consider the current practice. The possible solutions that emerged from our analysis were: education, negotiation and definition of the main ISS concepts. We suggest education in cases where ISS procedures and responsibilities are unknown and/or wrongly understood. Our contribution in relation to the earlier awareness studies is that value conflicts analysis directs attention to specific areas that should be focused on in education and awareness programs. Further, we suggest negotiations in cases where ISS procedures and processes do not reflect work processes. Negotiation means finding the middle ground between implementers and users and in this way find better ways to achieve the same result. Lastly, we suggest defining of key ISS concepts in cases when different actors give different meanings to these concepts.

4 Negotiation involves users in the ISS design and implementation process. The view of the users' ISS compliance is formed by the longstanding technical tradition and the use of traditional technically oriented models and methods [31]. In the technical tradition, people are considered as the biggest threat and for that reason, their behaviors have to be restricted and controlled in order to achieve a high level of ISS. Users are viewed as the 'weakest link' or 'the enemy inside', and they usually have a passive role in the ISS design and implementation process [31]. Users' knowledge about the current practice, their responsibility-taking and intentional actions are not utilized in ISS management. The lack of involvement in the ISS design and implementation process is considered as a problem and often results in users' resistance to implemented ISS rules [3]. We argue that, suggested in this study, negotiation between ISS implementers and users in order to find work-friendly ISS processes is an opportunity to involve users in the ISS design and implementation process and consequently decreases their resistance to implemented ISS rules.

6 Conclusion and Future Research

In this paper we have analyzed value conflicts between ISS values and work values existing in a social services organization. In our case study, new ISS processes were approved by management and were implemented without considering the consequences on the users' daily work. The poor implementation resulted in non-compliance behaviors and a decreased level of ISS security. Based on the analysis of value conflicts that emerged in relation to the implementation and use of the ISSP, we suggest solutions that should result in more work-friendly ISS processes and an ISSP implementation that both improve compliance with ISS rules and consider the current practice.

In summary, the paper offers four key findings: 1) it is important to consider different values and value conflicts in order to find workable ISS processes; 2) it is important to realize that values embedded in the ISSP may differ from values guiding the ISSP implementation; 3) value conflict analyses can direct management's

attention to new security solutions; 4) negotiation involves users in the ISS design and implementation process.

Findings presented in this paper are useful for expressing an ISS policy in an informal way. Future research needs to investigate more thoroughly how to map an informal specification into a formal policy, part of which has to be declared by means of the underlying information system and then automatically enforced in the organization. More studies are also needed about which features of modern security enforcement systems could be appropriate in this case.

References

1. Ernst, Young: Moving beyond compliance. Global Information Security Survey (2008)
2. Mattia, A., Dhillon, G.: Applying Double Loop Learning to Interpret Implications for Information Systems Security Design. In: The IEEE Systems, Man & Cybernetics Conference, Washington, DC, October 5-8 (2003)
3. Lapke, M., Dhillon, G.: Power relationships in information systems security policy formulation and implementation. In: The 16th Annual European Conference on Information Systems (ECIS), Galway, Ireland (2008)
4. Mishra, S., Dhillon, G.: Information systems security governance research: a behavioral perspective. In: The 1st Annual Symposium on Information Assurance, Academic Track of 9th Annual NYS Cyber Security Conference, New York, USA (2006)
5. Thomson, K.L.: Information Security Conscience: a precondition to an Information Security Culture. In: 8th Annual Security Conference, Las Vegas, NV, USA, April 15-16 (2009)
6. Ramachandran, S., Rao, V.S., Goles, T.: Information Security Cultures of Four Professions: A Comparative Study. In: Proceedings of the Forty First Hawaii International Conference on System Sciences 2008, Big Island, Hawaii (2008)
7. Ruighaver, A.B., Maynard, S.B., Chang, S.: Organisational security culture: Extending the end-user perspective. *Computers & Security* 26(1), 56–62 (2007)
8. von Solms, R., von Solms, B.: From policies to culture. *Computers & Security* 23(4), 275–279 (2004)
9. Albrechtsen, E., Hovden, J.: The information security digital divide between information security managers and users. *Computers Security* 28(6), 476–490 (2009)
10. Hedström, K., Kolkowska, E., Karlsson, F., Allan, J.P.: Value conflicts for information security management. *The Journal of Strategic Information Systems* 20(4), 373–384 (2011)
11. Vast, E.: Danger is in the eye of the beholders: Social representations of Information Systems security in healthcare. *Journal of Strategic Information Systems* 16, 130–152 (2007)
12. Baskerville, R., Siponen, M.: An information security meta-policy for emergent organizations. *Logistics Information Management* 15(5/6), 337–346 (2002)
13. Siponen, M., Wilson, R.: Information security management standards: Problems and solutions. *Information and Management* 46, 267–270 (2009)
14. Vroom, C., von Solms, R.: Towards information security behavioural compliance. *Computers & Security* 23(3), 191–198 (2004)
15. Thomson, K.L., von Solms, R., Louw, L.: Cultivating an organizational information security culture. *Computer Fraud and Security* (10), 7–11 (2006)

16. Siponen, M.: A Conceptual Foundation for Organizational Information Security Awareness. *Information Management & Computer Security* 8(1), 31–41 (2000)
17. Furnell, S.M., Gennatou, M., Dowland, P.S.: A prototype tool for information security awareness and training. *Logistics Information Management* 15(5), 352–357 (2002)
18. Kolkowska, E.: A Value Perspective on Information System Security - Exploring IS security objectives, problems and value conflicts. Orebro University, Orebro (2009)
19. Sasaki, R., Hidaka, T., Moriya, M., Taniyama, H., Yajima, K., Yaegashi, Y., Kawashima, Y., Yoshiura, H.: Development and applications of a multiple risk communicator. In: *Sixth International Conference on Risk Analysis*, pp. 241–249. WIT Press (2008)
20. Mumford, E.: *Values, Technology and Work*. The Hague Martinus Nijhoff Publishers (1981)
21. Schein, E.: *The corporate culture survival guide*. Jossey-Bass Publishers, San Francisco (1999)
22. Dhillon, G.: *Principles of information systems security: text and cases*. Wiley Inc., Hoboken (2007)
23. Myers, M.D.: *Qualitative research in business & management*. Sage Publications, London (2009)
24. Lagsten, J.: *Utvärdera Informationssystem: Pragmatiskt perspektiv och metod*. Linköping University, Linköping (2009)
25. Silverman, D.: *Interpreting qualitative data. Methods for analyzing talk, text and interaction*, 2nd edn. Sage, London (2001)
26. Kolkowska, E.: Lack of compliance with IS security rules: value conflicts in Social Services in Sweden. In: *8th Annual Security Conference, Las Vegas, USA, April 15-16 (2009)*
27. Lamb, R., Kling, R.: Reconceptualizing users as social actors in information systems research. *MIS Quarterly* 27(2), 197–235 (2003)
28. Hedström, K., Dhillon, G., Karlsson, F.: Using Actor Network Theory to Understand Information Security Management. In: *The 25th Annual IFIP TC 2011, Brisbane, Australia, September 20-23 (2010)*
29. Straub, D., Nance, W.: Discovering and Disciplining Computer Abuse in Organizations: A Field Study. *MIS Quarterly* 14(1), 45–60 (1990)
30. Kankanhalli, A., Teo, H.H., Tan, B.C., Wei, K.K.: An Integrative Study of Information Systems Security Effectiveness. *International Journal of Information Management* 23(2), 139–154 (2003)
31. Siponen, M.: An analysis of the traditional IS security approaches: implications for research and practice. *European Journal of Information Systems* 14, 303–315 (2005)

When Convenience Trumps Security: Defining Objectives for Security and Usability of Systems

Gurpreet Dhillon¹, Tiago Oliveira², Santa Susarapu¹, and Mário Caldeira³

¹ School of Business, Virginia Commonwealth University, Richmond, VA, USA
{gdhillon, susarapusr}@vcu.edu

² ISEGI, Universidade Nova de Lisboa, Portugal
toliveira@isegi.unl.pt

³ ISEG, Technical University of Lisbon, Portugal
caldeira@iseg.utl.pt

Abstract. Security and usability of systems continues to be an important topic for managers and academics alike. In this paper we propose two instruments for assessing security and usability of systems. These instruments were developed in two phases. In Phase 1, using the value-focused thinking approach and interviews with 35 experts, we identified 16 clusters of *means* and 8 clusters of *fundamental* objectives. In phase 2 drawing on a sample of 201 users we administered a survey to purify, ensure reliability, and unidimensionality of the two instruments. This resulted in 15 means objectives, organized into four categories (*minimize system interruptions and licensing restrictions, maximize information retrieval, maximize system aesthetics, and maximize data quality*) and 12 fundamental objectives grouped into four categories (*maximize standardization and integration, maximize ease of use, maximize system capability, and enhance system related communication*). Collectively the objectives offer a useful basis for assessing the extent to which security and usability has been achieved in systems.

Keywords: security values, usability values, value focused-thinking, qualitative methods, instrument development, quantitative methods.

1 Introduction

Consider a situation where Alice has to set up her friend's new computer. Alice sets up a limited user account, changes the file permissions for the entire user class, thus allowing *write* and *modify* access for all non system partitions. Alice goes a step further and installs a freeware *SuRun* so that in case her friend called her again, she could run certain programs without necessarily asking for administrative rights. Alice also installed *Returnil*, a software suite that allows automatic recovery in case of problems. All in all, Alice considered this to be a rather secure and a usable arrangement. Alice's friend however encountered significant problems with the set up. Following a system update, SIW (System Information for Windows) kept popping up warning windows and later hung up. With little computing knowledge Alice's

friend tried uninstalling the virtual machine. With some luck she was able to delete the folder, which stopped the conflict messages from popping up. However in the process she probably left her computer open for several vulnerabilities.

So, where does one draw a line between security and usability? Alice thought that she was probably doing a pretty good job, but by imposing her own values in configuring a system, she ended up creating several vulnerabilities without even realizing that they existed. Within organizations it is common to see such problem occur and the solution perhaps resides in addressing user expectation and inferring authorizations based on designations [1]. While the literature has made several calls for balancing security and usability, these have not been adequately heeded to (e.g. see [2, 3]). In majority of the cases where security and usability has been considered, it's been an afterthought at best with developers design systems and later realizing that security and usability considerations had not been adequately addressed. Such development duality typically results in a haphazard system development (see [4, 5]). A definition of a common set of security and usability objectives would to a large extent elevate the development duality problem by presenting objectives that must be achieved. Such objectives would also help in providing a strategic direction for secure and usable systems development.

In this paper, following Keeney [6], Dhillon and Torkzadeh [7], and Torkzadeh and Dhillon [8], we use value-focused thinking to define security and usability objectives. The study was undertaken in two phases. Phase 1 involved a qualitative definition of value-based objectives. Phase 2 helped in developing a parsimonious set of security and usability objectives. The two phases and a final set of objectives are discussed in the rest of the paper.

2 Value Focused Security and Usability Objectives

As stated earlier, methodologically this research is based on Keeney's [6] 'value focused thinking' approach. Keeney suggests that most decision-making methods are based on alternative thinking practices where choices are made only from a limited list of available alternatives. The alternative based approach is constrained by the limits imposed by decision-makers in the process of identifying constraints and subsequently alternatives. As a consequence, individuals tend to forget what they really want to achieve. Since achieving an objective is the primary reason for being involved in any decision situation, Keeney contends that one should remain focused on the bottom-line objectives, which makes decisions meaningful and of value, instead of making choices among current alternatives. Value focused thinking is proposed as a method by Keeney, to address the most fundamental question - what do we want to do and why. Research conducted by Keeney (e.g. see [6, 9]), has attempted to expose underlying values in a wide array of decision contexts. The inherent argument is that the value thinking process can help researchers and managers alike to be proactive and hence create more alternatives instead of being limited by available choices.

Value focused thinking consists of two main steps in eliciting and framing values: (1) conduct interviews and construct a list of what they want in the decision context, (2) convert these statements into a common format of objectives (an object and a preference). In terms of modeling the means and fundamental objectives, a network hierarchy can also be put together. In the context of our research, this two-step method is applied in order to assess values attached by users, to IS Security and Usability. Thirty-five end-users of IS/IT services were contacted from employees of five large businesses in the US. The businesses represented the following industries – IT consulting, Hotel and Casino, Banking, Education and Training. The interviews formed the basis for eliciting the values.

Construct a List of What Users Want. The best way to find out what users value most is to ask them. Also, it is better to ask as many users as possible because different users may have different values and they may express them differently. However a difficulty lies in the latency of these values. In many cases users' values are hidden under the surface. Keeney recommends several stimulation techniques to surface these latent values. We chose a combination of two techniques to identify the latent values. The first method used was a wish list. Each interviewee was asked to express what their needs were in terms of security and usability of systems they used within their organizations. The second method, which augments the simple wish list method, was the probing technique. In order to expand the wish list and whenever subjects are having problem articulating what they want, interviewer posed several probing questions prepared beforehand. The list of probing questions included: "If you did not have any constraints, what would your objectives be?" "What needs to be changed from the status quo?" "How do you evaluate security and usability of systems?" "What do you expect in terms of security and usability?" "How do they tell if security and usability of systems is good or bad?" Besides asking the interviewees to generate a wish list, we also asked them to generate a list of problems and shortcomings in security and usability of systems they used. The basic idea behind asking problems and shortcomings was to generate objectives by articulating their concerns. The thirty-five interviews generated three hundred and thirty seven wishes/problems/ concerns.

Convert Statements into Objectives. These statements are converted to objectives, using a verb (direction of change) plus an object (target of change) format. Some statements on the list are compound sentences, which produce more than one objective, and some statements were being repeated by several users. For example, one user wishes "to be educated in moving between different applications and wants help when he gets lost." Two objectives can actually be derived from this wish: (1) ease of navigation through the application and (2) enhance system training quality. To eliminate these ambiguities and redundancies, two researchers reviewed each item on the list independently. This review and refinement produced one hundred and thirty objectives in a common form of a verb plus an object.

In ensuring security and usability, users wanted to achieve these one hundred and thirty objectives. However, these objectives are not adequately articulating values yet, and also include duplication. The objectives were then categorized in order to surface the meanings, and the values attached to cluster the objectives. The categorization resulted in twenty-four clusters of objectives.

As a next step of framing values out of objectives, twenty-four objectives were classified into two categories: means objectives and fundamental objectives. The criterion of classification is whether an objective is an intermediate one, i.e. is it a means to achieve another objective or is it a final and a fundamental one in terms of security and usability. As a result, eight fundamental objectives were identified. The means objectives, a total of sixteen, are presented in Table 1 and fundamental objectives in Table 2. The tables do not show all the individual objectives. These are available from the authors on request.

Table 1. Means objectives

Means Objectives (16 clusters, 91 items)	
Clarify & improve system documentation e.g.: Ensure easy access to system documentation	Maximize system access e.g.: Define role-based external access
Improve system search capability e.g.: Ensure semantic based search features	Maximize system efficiency e.g.: Ensure process fairness
Maximize data quality e.g.: Enhance data integrity	Maximize system esthetics e.g.: Enhance visualization of system security
Maximize database and system access e.g.: Ensure web access to the system	Maximize system integrity e.g.: Maximize system adaptability
Maximize disaster recovery e.g.: Ensure data availability	Maximize system maintainability e.g.: Ensure hardware robustness
Maximize productivity e.g.: Ensure automated password retrieval	Maximize system reliability e.g.: Maximize process execution accuracy
Maximize security & privacy e.g.: Decrease restrictiveness of system	Maximize task efficiency e.g.: Maximize automation of manual tasks
Maximize self-efficacy in training e.g.: Enhance system training quality	Minimize system interruptions e.g.: Minimize system down-time

Table 2. Fundamental objectives

Fundamental Objective (8 clusters, 59 items)	
Enhance system related communications e.g.: Ensure exception reports go to management	Maximize system administration functionality e.g.: Enhance connectivity at affordable price
Improve data organization e.g.: Ensure data archival functionality	Maximize system capability e.g.: Enhance application features
Maximize ease of use e.g.: Ensure ease of navigation through application	Maximize system integration e.g.: Ensure functionality is designed into system
Maximize standardization of system features e.g.: Enhance customizable interfaces	Maximize user requirements elicitation e.g.: Ensure system functionality meets requirements

3 Quantitatively Derived Parsimonious Set of Security and Usability Objectives

3.1 Method

In Phase 1, 150 items that influence information systems (IS) usability were developed. These items were based on the total set of 130 objectives identified in phase 1. The additional 20 items were added to ensure that all objectives were well represented in the survey instrument. These items were grouped into two categories of means and fundamental objectives. The means objectives contain 91 questions (items) grouped in 16 clusters (constructs). The fundamental objectives present 59 items grouped in 8 constructs. The large number of items found in both objective sets may have led to redundancy, but it helped content validity, since we were drawing on a large universe of possible items [10].

Based on items found in Phase 1, we developed a questionnaire. A five-point Likert-type scale was used, with a range from one (strongly disagree) to five (strongly agree). The respondents were asked to express agreement with 150 questions pertaining to the following context statement - *“In order to respond to the questions below, think of any system that you may be using or are familiar with. What would your ideal state be in terms of achieving your objectives?”* The survey was administered to graduate and undergraduate students in a European University. We obtained a sample of 201 (30.3% male, 69.7% female) respondents. The respondent rate was 66.3%. The respondents were mature students with work experience in a variety of professions, such as, banking, sales, healthcare, information systems, engineering, education among others areas. The age of the participants ranged from 18 and 50. All participants had experience with security and usability of IS, thus being qualified to answer this survey.

The analysis of the data was undertaken with several goals: purification, reliability, and unidimensionality. The following three steps were used in the elimination process:

1. We eliminated the items if their corrected item-total correlation (the correlation of each item with the sum of the other items in its category) was less than 0.5, because, according to Churchill [11], all items that belong to the same domain of the concept (construct) should be highly inter-correlated.
2. We eliminated the items if the reliability of the remaining items was at least 0.9. Cronbach's α was computed to see if additional items could be eliminated without substantially lowering reliability.
3. A factor analysis was undertaken with the remaining items for each group to eliminate items that were not factorially pure [12]. This means that we eliminated items that had a loading greater than 0.3 on more than one factor.

The purification of the items was done before the factor analysis, to produce less dimensions and not to confound the interpretation of the factor analysis [11]. This methodology provides brevity and simplicity of the factor structure.

3.2 Data

Means Objectives. We performed the item procedure, described before, to purify the means objectives category. First, corrected item-total correlation of less than 0.5

suggests the elimination of 29 items. Second, the reliability analysis does not eliminate any item. Finally, the factor analysis suggests an elimination of 47 more items.

After the elimination process, 15 items of the means objectives category were obtained. In Table 3, we present the results of the factor analysis using varimax rotation for the retained items. Bartlett's test of sphericity was 1278.4 ($p < 0.001$). This means that the data contains enough common variance to perform a factor analysis. Kaiser-Meyer-Olkin (KMO) measures the adequacy of the sample; KMO is 0.88 ($KMO \geq 0.80$ is good [13]), which reveals that the matrix of correlation is adequate for the factor analysis. The results of the factor analysis revealed four factors with eigenvalues greater than one. These factors explain 67.5% of the variance contained in the data.

Table 3. Factor analysis of means objectives in Phase 2 (n=201)

	F1	F2	F3	F4	Corrected Item-Total Correlation
Minimize system interruptions and licensing restrictions					
Minimize unnecessary system lock outs & time outs	0.73				0.69
Minimize system interruptions	0.73				0.64
Minimize application licensing restrictions	0.71				0.64
Minimize the total cost of ownership	0.63				0.63
Minimize system down-time	0.55				0.55
Maximize information retrieval					
Maximize efficiency of system tasks		0.74			0.74
Maximize task efficiency		0.73			0.71
Maximize system efficiency		0.71			0.72
Maximize database and system access		0.62			0.65
Maximize system esthetics					
Ensure color combinations are visually appealing			0.81		0.66
Ensure good application display			0.68		0.66
Maximize system esthetics			0.59		0.60
Maximize data quality					
Enhance data integrity				0.74	0.61
Increase timely application data access				0.68	0.59
Increase ease in editing and updating of application data accurately				0.56	0.53
Eigenvalue	5.77	1.70	1.38	1.26	-
% Variance	38.5%	11.3%	9.2%	8.4%	-

Note: loadings greater than 0.3 are reported; the items are grouped by highest factor loading and presented by descending order.

The four factors found were easily interpreted, they are: *minimize system interruptions and licensing restrictions* (five items), *maximize information retrieval* (four items), *maximize system esthetics* (three items), and *maximize data quality* (three items). The range of loadings is respectively: 0.55-0.73, 0.62-0.74, 0.59-0.81, and 0.56-0.74. All the factors have a loading greater than 0.5. This indicates that our analysis employs a well-explained factor structure.

The range of corrected item-total correlation varies between 0.55 to 0.69 for *minimize system interruptions and licensing restrictions*, 0.65 to 0.74 for *maximize information retrieval*, 0.60 to 0.66 for *maximize system esthetics*, and 0.53 to 0.61 for *maximize data quality*.

The reliability for each construct was: 0.84 for *minimize system interruptions and licensing restrictions*, 0.86 for *maximize information retrieval*, 0.80 for *maximize system esthetics*, and 0.75 for *maximize data quality*. The overall reliability for the 15 item scale was 0.88. This reveals that reliability exceeds the suggested cutoff value of 0.70 [14].

Table 4. Factor analysis of fundamental objectives in Phase 2 (n=201)

	F1	F2	F3	F4	Corrected Item-Total Correlation
Maximize standardization, integration and user requirements					
Maximize standardization of system features	0.86				0.78
Maximize functional standardization	0.82				0.77
Maximize system interoperability	0.64				0.67
Maximize automated internal controls	0.61				0.65
Maximize ease of use					
Maximize ease of use		0.85			0.74
Maximize ease of system use		0.77			0.73
Maximize ease of system navigation		0.56			0.59
Maximize system capability					
Enhance explanatory features in the system			0.75		0.72
Enhance geographic location features			0.73		0.69
Enhance e-commerce features			0.62		0.61
Enhance system related communications					
Minimize user interaction with system developers				0.87	0.81
Minimize users' interaction with technical personnel				0.87	0.81
Eigenvalue	5.17	1.62	1.26	1.13	-
% Variance	43.0%	13.5%	10.5%	9.4%	-

Note: loadings greater than 0.3 are reported; the items are grouped by highest factor loading and presented by descending order.

Fundamental Objectives. To purify the fundamental objectives category we used the same item purification procedure. The first criteria was to eliminate items below 0.5, it allowed us to eliminate 17 of the 59 items obtained in Phase 1. The second criteria, reliability analysis, did not eliminate any items. Finally, the factor analysis suggested the elimination of 30 more items.

Subsequent to the elimination process, the fundamental objectives scale included 12 items. First, Bartlett's test of sphericity was 1292.3 ($p < 0.001$). These factors explain 76.5% of the variance contained in the data. The KMO is 0.82 ($KMO \geq 0.80$ is good [13]), which reveals that the matrix of correlation is adequate for factor analysis. This reveals that the data contains enough common variance to perform the factor analysis. Four factors with eigenvalues greater than one is obtained (Table 4), and the interpretation of each factor was not difficult, i.e.: *maximize standardization, integration and user requirements* (4 items), *maximize ease of use* (3 items), *maximize system capability* (3 items), and *enhance system related communications* (2 items). The ranges for factor loading were, respectively, 0.61-0.86, 0.56-0.85, 0.62-0.75, and 0.87-0.87. All the factors have a loading greater than 0.5. This indicates that our analysis employs a well-explained factor structure.

The range of corrected item-total correlation for each item varies between: 0.61 to 0.78 for *maximize standardization, integration and user requirements*, 0.59 to 0.74 for *maximize ease of use*, 0.61 to 0.72 for *maximize system capability*, and 0.81 to 0.81 for *enhance system related communications*. The reliability scores were 0.87, 0.83, 0.82, and 0.89 respectively for each construct. The overall reliability for the 18 item scale was 0.88. The reliability exceeds the suggested cutoff value of 0.70 [14].

In short, the results obtained in Phase 2 present good reliability and validity measures for both instruments developed (means objectives: 4-factor with 15 items; fundamental objectives: 4-factor with 12 items).

4 Discussion

The findings from our research present an interesting mix of security and usability objectives that any software developer would find useful. The fundamental objectives identified include: *maximize standardization and integration, maximize ease of use, maximize system capability, enhance system related communication*. Typically system developers tend to focus on one or the other set of objectives. For instance past research has typically suggested that perceived ease of use effects perceived usefulness and hence behavioral intention to use [15]. However the measures are not entirely useful to a typical system developer (*viz.* constructs such as perceptions of internal control, computer anxiety, playfulness etc). From a security and usability perspective perhaps ease of system navigation and the general perception of easy to use seem more logical.

Another important aspect as always, is related to system related communications. In the literature various proposals have been made. These have ranged from development of *hybrid managers* [16] who can help bridge the gap between technical system developers and actual users to the development of intrinsic competencies for

harnessing technology [17]. While all these assertions may present significant theoretical opportunities, typically in organizations we are still to see adequate management of interactions between users and the technical staff. Inability to deal with such relationships results in systems getting abused or not properly used. And thus pose significant security challenges.

The importance of standardization and integration in security and usability cannot be underestimated. A casual review of various security and usability standards itself suggests a plethora of options. In the usability community although ISO standards such as ISO 9241 1995 exist, there is lack of consensus with respect to the conformance methods. Dzida [18] notes, "If a product is claimed to meet a standard, the procedure used in testing the product against the requirements should be specified to guarantee reproducibility of results. Some standards prescribe a certain test method, some recommend a method, and some inform the reader that the procedure used in testing is a matter of negotiation between the parties involved". In information security, while ISO 27799 exists, there are equally other competing standards (*viz.* SSE-CMM among others). Perhaps one of the reason for the inadequacy of existing standards and an existence of a large set is because of a lack of core objectives that need to be achieved in managing security and usability. More often than not the standards seem to be "cobbled" together to fit a purpose. Our research has identified four rather interesting standardization requirements - standardization of features, functional standardization, interoperability and automated internal controls. As a case in point simply consider academic university websites across institutions. Perhaps some functional and feature standardization would come in handy as would access and availability of information. Failure to do so not only makes it difficult to navigate systems but; opens up institutions to several vulnerabilities (e.g. see website breaches at Utkal University India, St Louis University USA among others).

Our research has found that fundamental objectives for security and usability can be achieved if there is a corresponding appreciation of the means to achieve the fundamental objectives. Means objectives identified in this study include: *minimize system interruptions and licensing restrictions, maximize information retrieval, maximize system aesthetics, maximize data quality.*

In our study we found that the higher licensing costs and poor quality of systems and data results in bypassing legal software and many of the controls. This can have serious consequences on the integrity of systems. As a consequence, virus and malware problems are also known to creep in. Grabosky and Smith [19] has argued that proper guardianship helps in preventing such vulnerabilities. Guardians are also known to facilitate usability of system. Retrieval of information from systems has also been a well-researched topic area and sits at the cusp of security and usability dimensions. Griffith and Jakobsson [20] for example note that mother's maiden name, a usual means to retrieve data from financial institutions, can actually be deduced with great accuracy from public records. Some progress has however been made by adding personal knowledge questions for information retrieval, but more so for fallback authentication. From a security and usability perspective enough thought has not gone into the strategic aspects of information retrieval and their relationship to

security and usability. Our research indicates it to be an important objective for consideration.

Another important aspect, as identified in our study, pertains to data quality. In the literature poor data quality has been known to have two implications. First, the security of an enterprise gets compromised (see Redman [21]). This is because security is directly linked to the accuracy of data. Second, usability of the system gets questioned. If the system and the data therein is not useful [22], is out of context, there is typically a loss of ownership. This results in significant security problems.

Theoretical and Practical Contributions. The major theoretical contribution of the security and usability objectives presented in this paper is their intertwined nature. Typically security and usability have been treated as separate constructs. At best researchers have pondered about security implications of low usability systems or the implications of highly secure systems on lack of usability (see [1, 2, 21, 22]). While both the contentions are worthy of investigation, they fall short of providing a strategic direction for secure and usable system development. We believe that our research provides a theoretical framework for addressing security and usability. The major tenants of our theoretical contribution are:

- Well-grounded security and usability objectives that are based on the values of individuals. Value based objectives are considered much better for strategic planning relative to the alternative based objectives (see [23]).
- Our value proposition combines security and usability. While in the literature calls for aligning the two have been made [2], there has been practically no follow up research. By combining the two constructs we have in many ways presented a well-aligned set of security and usability objectives.

At a practical level, research presented in this paper offers requirement objectives that system developers should use to design security and usability into the systems. Typically security has been considered as an afterthought in the design process [7]. And usability has been addressed in an iterative manner. While system developers seem to develop their own processes in addressing security and usability concerns, a structured framework is however a preferred approach. We believe that the guidance provided by the security and usability objectives described in this paper forms a solid, theoretically grounded, empirically derived basis for the range of development tasks.

5 Conclusion

This paper examines the combination of security and usability of systems in two phases. In Phase 1, we developed value-focused security and usability objectives. A qualitative approach revealed 150 objectives, 91 means objectives and 59 fundamental objectives, grouped respectively into 16 and 8 means and fundamental objectives. A quantitative approach was developed in Phase 2, with the aim of purification, reliability and unidimensionality, from which a parsimonious set of security and usability objectives were derived. 15 means objectives were obtained, grouped in four constructs, which are: *minimize system interruptions and licensing*

restrictions, maximize information retrieval, maximize system aesthetics, and maximize data quality. In terms of fundamental objectives, 12 fundamental objectives were obtained, grouped in four constructs, respectively: *maximize standardization and integration, maximize ease of use, maximize system capability, and enhance system related communication.* We believe that this paper offers a good basis for better understanding of security and usability objectives. Finally, with further research the instruments presented in this paper could be further validated.

References

1. Yee, K.P.: Aligning security and usability. *IEEE Security & Privacy* 2, 48–55 (2004)
2. DeWitt, A.J., Kuljis, J.: Aligning usability and security: a usability study of Polaris. In: *Proceedings of the Second Symposium on Usable Privacy and Security*, pp. 1–7. ACM, Pittsburgh (2006)
3. Frøkjær, E., Hertzum, M., Hertzum, M., Hornbæk, K.: Measuring usability: are effectiveness, efficiency, and satisfaction really correlated? In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 345–352. ACM, The Hague (2000)
4. Baskerville, R.: Information systems security design methods: implications for information systems development. *Computing Surveys* 25, 375–414 (1993)
5. Dhillon, G.: *Managing information system security*. Macmillan, London (1997)
6. Keeney, R.L.: *Value-focused thinking*. Harvard University Press, Cambridge (1992)
7. Dhillon, G., Torkzadeh, G.: Value-focused assessment of information system security in organizations. *Information Systems Journal* 16, 293–314 (2006)
8. Torkzadeh, G., Dhillon, G.: Measuring factors that influence the success of Internet commerce. *Information Systems Research* 13, 187–204 (2002)
9. Keeney, R.L.: The value of Internet commerce to the customer. *Manage. Sci.* 45, 533–542 (1999)
10. Boudreau, M.C., Gefen, D., Straub, D.W.: Validation in information systems research: A state-of-the-art assessment. *MIS Quarterly* 25, 1–16 (2001)
11. Churchill, G.A.: Paradigm for Developing Better Measures of Marketing Constructs. *Journal of Marketing Research* 16, 64–73 (1979)
12. Weiss, D.J.: Factor analysis and counseling research. *Journal of Counseling Psychology* 17, 477–485 (1970)
13. Sharma, S.: *Applied Multivariate Techniques*. John Wiley & Sons, Inc., New York (1996)
14. Nunnally, J.C.: *Psychometric Theory*. McGraw-Hill, New York (1978)
15. Venkatesh, V.: Determinants of perceived ease of use: Integrating control, intrinsic motivation, and emotion into the technology acceptance model. *Information Systems Research* 11, 342–365 (2000)
16. Earls, M.J., Skyrme, D.J.: Hybrid managers — what do we know about them? *Information Systems Journal* 2, 169–187 (1992)
17. Dhillon, G.: Organizational competence for harnessing IT: A case study. *Information & Management* 45, 297–303 (2008)
18. Dzida, W.: International usability standards. *ACM Computing Surveys* 28, 173–175 (1996)
19. Grabosky, P., Smith, R.: Telecommunication fraud in the digital age: The convergence of technologies. In: Wall, D.S. (ed.) *Crime and the Internet*. Routledge, London (2001)

20. Griffith, V., Jakobsson, M.: Messin' with Texas Deriving Mother's Maiden Names Using Public Records. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 91–103. Springer, Heidelberg (2005)
21. Redman, T.C.: The impact of poor data quality on the typical enterprise. *Communications of the ACM* 41, 79–82 (1998)
22. Arts, D.G.T., de Keizer, N.F., Scheffer, G.J.: Defining and improving data quality in medical registries: A literature review, case study, and generic framework. *Journal of the American Medical Informatics Association* 9, 600–611 (2002)
23. Leon, O.G.: Value-focused thinking versus alternative-focused thinking: Effects on generation of objectives. *Organizational Behavior and Human Decision Processes* 80, 213–227 (1999)

Security-by-Contract for the OSGi Platform

Olga Gadyatskaya, Fabio Massacci, and Anton Philippov

DISI, University of Trento, Italy
{name.surname}@unitn.it

Abstract. The natural business model of OSGi is dynamic loading and removal of bundles or services on an OSGi platform. If bundles can come from different stakeholders, how do we make sure that one's services will only be invoked by the authorized bundles? A simple solution is to interweave functional and security logic within each bundle, but this decreases the benefits of using a common platform for service deployment and is a well-known source of errors. Our solution is to use the Security-by-Contract methodology (S×C) for loading time security verification to separate the security from the business logic while controlling access to applications. The basic idea is that each bundle has a *contract* embedded into its manifest, that contains details on functional requirements and permissions for access by other bundles on the platform. During bundle installation the contract is matched with the platform security policy (aggregating the contracts of the installed bundles). We illustrate the S×C methodology on a concrete case study for home gateways and discuss how it can help to overcome the OSGi security management shortcomings.

1 Introduction

The Open Services Gateway Initiative (OSGi) framework [1] is one of the most flexible solutions for the deployment of pervasive services in home, office, or automobile environments. OSGi-compatible implementations constitute the backbone of many recent proposals for embedded systems [2] or other industry-based services. The OSGi services are also the basic building blocks for service mash-ups extending the classical “smart homes” scenarios to richer settings [8].

The OSGi framework redefines the modular system of Java by introducing *bundles*: JAR files enhanced with specific metadata. The *service* layer connects bundles in a dynamic way with a publish-find-bind model for Java objects. An OSGi-based system has several advantages over the traditional JAR modules because it provides a robust integrated environment where bundles can be published and exported to be used by other bundles, handles bundle versioning for every new deployment and maintains the bundle lifecycles. Moreover, the bundles can be updated dynamically at run-time without restarting the system and seamlessly to other bundles.

As a result, an OSGi platform is expected to be highly dynamic. All pervasive and mash-up applications expect that bundles can be installed, updated or removed at any time depending on business needs, and also they can collaborate arbitrarily in order to ensure enhanced composite services.

From a security perspective, the possibility of bundle interactions is a threat for bundle owners. Since bundles can contain sensitive data or activate sensitive operations (such as locking doors of somebody’s house), it is important to ensure that the security policy of each bundle owner is respected by other bundles. However, such aspects have been only partially investigated.

How do we make sure that one’s services are invoked only by authorized bundles? A simple solution is to rely on service-to-service authentication to identify the services and then interleave functional and security logic into bundles, for example, by using aspect-oriented programming [11]. However, this decreases the benefits of using a common platform for service deployment and significantly hinders evolution and dynamicity: any change to the security policy would require redeployment of the bundle (even if its functionalities are unchanged). Vice versa, any changes in the bundle’s code would require redeployment of security.

Our solution is to use the Security-by-Contract methodology [3,5] for loading time security verification in order to separate security and the business logic while achieving a sufficient protection of bundles among themselves. S×C’s basic idea is that each bundle will have a *contract* embedded into its manifest file. The contract contains details on the functional requirements and lists access permissions for other bundles on the platform. During installation of bundles the contract is extracted and matched with the platform security policy aggregating the contracts of all installed bundles. Thus, after the check we can be sure (under reasonable assumptions) that the incoming bundle respects the security requirements of other bundles.

Overall contribution of the paper is twofold. First, we improve the OSGi security by introducing a way for bundle providers to define their security policies and enforcing them on the platform. Second, we extend the S×C paradigm to a permission-based security system, opening a way for it to be adopted further for other platforms.

The rest of this paper is organized as follows. Section 2 introduces the S×C scheme for OSGi, Section 3 presents the smart home service gateway case study and discusses the security and functionality challenges, Section 4 presents the formal model of the OSGi system, the contract notation and the checks that the S×C framework performs to ensure security. Section 5 evaluates the proposed solution, Section 6 overviews the related work and Section 7 concludes the paper.

2 The S×C Architecture

We assume at least a high-level understanding of the main notions of the OSGi platform such as *bundles* (the OSGi components made by developers), *services* (plain old Java objects connecting bundles in a dynamic fashion by the means of publish-find-bind model), *lifecycle* API (the API to install, start, stop, update, and uninstall bundles) and *modules* – the layer that defines how a bundle can import and export code.

The S×C framework consists of two main components: the ClaimExtractor and the PolicyChecker. The verification workflow is described on Figure 1. Informally,

the S×C process starts when a new bundle B is loaded. The ClaimExtractor component then accesses the manifest file, retrieves the information about imported and exported packages and obtains the bundle contract. Then the ClaimExtractor reads the `permissions.perm` file, which contains local bundle permissions, extracts permissions requested by the bundle B and related to services retrieval, packages importing, requirements of bundles, etc., and combines this information into the overall “security claims and needs” of the bundle. Then the PolicyChecker component receives the result from the ClaimExtractor and matches it with the security policy of the platform, that aggregates the security policies of all the installed bundles, and with the functional state of the platform (installed bundles, running services, etc.). If the PolicyChecker failed on either of the checks, the bundle is removed from the platform. Otherwise, it is installed and the security policy of the platform is updated by including the security requirements of B .

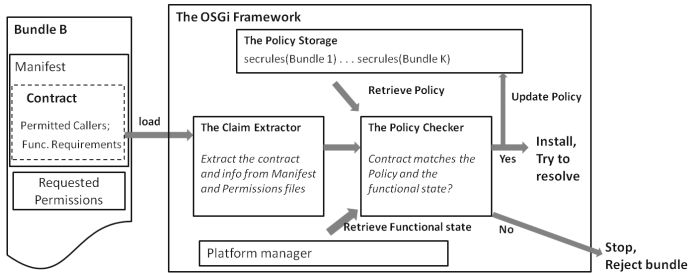


Fig. 1. The S×C Workflow

The S×C checks will be run in case of bundle code update or bundle policy update. These checks, however, are variations of the installation scenario. Thus, in the sequel we will focus only on the installation scenario as the most representative one.

In terms of technical realization, the S×C framework can be easily integrated with the OSGi framework as a bundle, provided it is granted the permissions.

3 The Running Example

We consider as a case study an OSGi platform deployed as a service gateway in a smart home. A security-unaware user, typically a resident of the smart home, can download and install untrusted bundles onto the platform. The bundles are providing various services on the platform and can interact generating an added value for the user. If the framework can host multiple third-party bundles which can freely register services, the platform owner has to make sure that there are no security or functionality problems of the different bundles installed by the end user (who most likely does not even know what is a bundle and just sees the web interfaces of the services). Thus, a threat scenario under investigation is a

case when a bundle gains unauthorized access to the sensitive data of another bundle (*security threat*), or a bundle is malfunctioning due to unavailability of some external entity (*functionality threat*).

The running example was shared by an OSGi Alliance¹ member Telefónica². Let us consider Alice, the smart home resident, and a telecom provider, the owner of the platform. Alice can download bundles for entertainment (news RSS feeds, media bundles from TV providers) or even bundles with traditional Internet content (like Facebook or Twitter), as new TV sets are used today for all these purposes. For the lack of space we use fictional names in the case study. The full version of the case study and more details about the S×C solution for the OSGi framework are available in the companion technical report [4].

Alice, a beginner stock market player, downloads and installs bundle *A* from provider *FSM.com* that provides her with an interface of the stock market operations. This bundle includes service S_A that retrieves updates about the stock prices. However, Alice later finds and installs another stock market bundle *B* from *BH.fr* provider, that provides a service for prices information retrieval S_B and a service S_{fr} that allows Alice to transfer money from her stock market account (registered on *BH.fr*) to her Happy Farm account on Facebook (*FB.com*). Thus, Alice also installs Happy Farm bundle *F*.

The bundle providers want to ensure that their security policies related to bundles and services usage are enforced on the Alice's platform. Their requirements are as follows:

FSM.com: Access to S_A service is allowed only for bundles signed by *FSM.com*.

BH.fr: Access to S_B service is allowed only for bundles signed by *BH.fr*. Only bundles signed by *BH.fr* can import the package containing S_B . Access to S_{fr} service can be granted only for bundles signed by *FB.com* or by *BH.fr*.

The OSGi platform at Alice's smart home has to ensure that the requirements of each provider are respected. Let us now discuss how the unmodified OSGi platform itself can enforce the requirements of the providers and why this approach is not satisfactory.

The OSGi Framework Background. Let us now present the relevant OSGi platform details [1]. An OSGi bundle is a JAR file that includes the *manifest.mf* file (*manifest file* in the sequel) containing the necessary OSGi metadata: the symbolic name of the bundle, its version, the dependencies and the provided resources. Some packages of a bundle can be *exported* (accessible for other bundles on the platform). A bundle also typically includes an *activator* (used for bootstrapping when the bundle is started) and a file with security permissions requested by the developer.

The OSGi system maintains the evolving lifecycles of the bundles. The bundle must first be *installed*. It can be *resolved* when all its *dependencies* are resolved. Bundles can depend on external entities (by requiring other bundles, an

¹ <http://www.osgi.org/>

² <http://www.telefonica.com/>

execution environment, a library, etc.). A bundle can start only after its dependencies were resolved. Bundles can express their dependencies as *requirements on capabilities*. Capabilities are attribute sets in a specific namespace and requirements are filter expressions that assert the attributes of the capabilities. A requirement is satisfied when there is at least one capability that matches the filter.

Bundles can interact through two complementary mechanisms: the export and import of packages and the service registration/lookup facility. A *service* is a normal Java object registered under a Java interface with the *service registry*. Bundles can register services, search for them, and receive notifications when the system state changes. A *service interface* is the specification of the service's public methods. A developer creates a service by implementing its service interface and registering it with the service registry. When requesting a service, a bundle specifies the name of the service interface and, optionally, a filter to narrow the search. In response, the framework first sends `ServiceReference` objects of the services that satisfy the search filter. The actual service object can then be acquired by passing the `ServiceReference` to the platform, provided the caller has the `ServicePermission[ServiceReference, GET]` permission.

Security of the OSGi platform is based on the Java 2 security architecture. Each bundle is associated with a set of permissions, that are queried at runtime. The OSGi specification defines `ServicePermission`, `BundlePermission` and `PackagePermission`, which are used for getting/registering a service, importing/exporting bundles and packages respectively. The platform can authenticate code by download location or by signer (digital signature). The `ConditionalPermissionAdmin` service manages the permissions based on a comprehensive conditional model.

A bundle has a set of *local permissions* defined by the developer in the file `permissions.perm` (*permissions file* in the sequel). These are the actual detailed permissions needed by this bundle to operate. A framework also provides an administrative service to associate a set of *system permissions* with a bundle. The bundle's *effective permissions* are the intersection of the local permissions and the system permissions. That is, a bundle cannot get more permissions than its local permissions set. Thereby, a bundle developer can limit the possible permissions of a bundle, but she cannot require a minimum set of necessary permissions to be granted and she cannot directly influence the set of system permissions granted to the bundle.

Security Challenges. A confidentiality attack can be realized by the bundle *A* of provider *FSM.com* getting access to the sensitive stock market prices service S_B of provider *BH.fr*. This might happen if *A* imports the package containing the service S_B definition, requires the bundle *B* (thus importing all its exported packages), or tries to get a reference to this service from the service registry and then get access to the object referenced.

The current OSGi security management suggests to address this security threat using the permissions system. Import of a package or a require-bundle action can be granted if the requiring bundle has corresponding permissions.

Simple reviewing of the manifest file and permissions file of the bundle A can report about a (potential) attempt to interact with the bundle B . However, there is no convenient and simple way for the owner of the bundle B , the $BH.fr$ provider, to declare which other bundles are allowed to import its packages.

Package importing can be guarded by the permissions mechanism. Currently only the platform owner (the telecom provider) can define and manage policies in the `ConditionalPermissionAdmin` service policy file. The $BH.fr$ provider might contact the telecom provider to ask him to set the required permissions, or its bundle B , being granted the necessary permissions, can add new permissions to the `ConditionalPermissionAdmin` policy file. These approaches are organizationally cumbersome and costly, as they require the operator to push the changes to its customers before any downloads of $BH.fr$ bundles, even if some customers have no intention of using them.

Another solution, that is traditional for mobile Java-based component systems, could be to ask Alice each time a specific permission is needed. But Alice is not the owner of the bundles to make such decisions, nor is she interested to do so.

The $S \times C$ paradigm, on the other hand, enables the bundle providers with a way to specify in the bundle contracts the necessary authorizations. The framework can then collect these authorizations from the bundles and incorporate them in the $S \times C$ policy on demand.

Service usage is more tricky though. Again, the necessary authorizations for the service usage (more precisely, GET permissions for service retrieval) can be delivered within bundle contracts and incorporated into the policy file of the system. But the invocations of the methods within a service, once the necessary reference is obtained, are not guarded by the permission check, and usually the security checks are placed directly within the service code, thus mixing the security logic with the execution logic.

Functionality Challenges. Let us now consider a more complex scenario.

Example 1. *Alice wants to install the Sims add-on from the EA.com provider. This add-on is packaged into the bundle C and it will provide an integration of the Happy Farm account with her the Sims account. The functional requirement of the EA.com provider is the following: “The bundle C can be installed if and only if the F bundle is available on the platform and provides the Happy Farm service S_F .”*

The requirement in Example 1 means that bundle C can be installed only if the service S_F is already provided on the platform. This requirement prevents the denial of service by the Sims bundle, which can cause a restart of the whole system since the bundles are running on top of a single JVM. This functional requirement is, in fact, unsupported by the current OSGi specification. Requirements/capabilities model cannot provide guarantees on the provided services (except that their definition exists on the platform).

In the following sections we present our solution in detail, starting with the formal model of the current OSGi specification in the next section.

4 The OSGi Platform Formal Model

The entities on the OSGi platform are bundles and services, but the formal model also takes into account the lifecycle of bundles, as it is an explicit part of the OSGi platform.

Let $\Delta_{\mathcal{B}}$ be a domain of symbolic bundle names, $\Delta_{\mathcal{S}}$ be a domain of symbolic service names and $\Delta_{\mathcal{P}}$ be a symbolic domain of package names. Let also $\Delta_{\mathcal{L}}$ be a domain of location strings for the bundles and Δ_{Sign} be a domain of signer names. We also define a set of local permissions requested by each bundle in its permissions file by $\text{Permissions}_{\text{bundle}}$, where each single permission perm is a pair $\langle \text{Target}, \text{Action} \rangle$.

Definition 1 (Bundle). *A bundle B is a tuple $\langle B, \text{state}(B), \text{exports}_{\mathcal{B}}, \text{imports}_{\mathcal{B}}, \text{Permissions}_{\mathcal{B}}, \text{Location}_{\mathcal{B}}, \text{Signer}_{\mathcal{B}} \rangle$, where $B \in \Delta_{\mathcal{B}}$, $\text{state}(B) \in \{\text{Installed}, \text{Resolved}, \text{Starting}, \text{Active}, \text{Stopping}\}$ is the current state in which the bundle resides at the moment, $\text{exports}_{\mathcal{B}} \subseteq \Delta_{\mathcal{P}}$ and $\text{imports}_{\mathcal{B}} \subseteq \Delta_{\mathcal{P}}$ are the sets of packages exported and imported by B correspondingly, $\text{Permissions}_{\mathcal{B}}$ is the set of local permissions of bundle B , $\text{Location}_{\mathcal{B}}$ is the download location and $\text{Signer}_{\mathcal{B}}$ is the signer (the provider) of the bundle B .*

Uninstalled state is not considered, as the bundle is not functioning in this state and cannot be transferred to other states, so this state is equivalent to deletion of a bundle.

We define an “is defined in” relation \diamond for packages, bundles and service interfaces. Let B be a bundle, S be a service interface and P be a package. We will denote by $P \diamond B$ the fact that P is a package defined in the bundle B and by $S \diamond P$ the fact that the service interface S is defined in the package P . For defining locations of bundles we define a \vdash relation (“comes from”), we will denote as $L \vdash B$ the fact that bundle B comes from location $L \in \Delta_{\mathcal{L}}$. Also for locations we can define a notion of location inclusion, we will denote as $L_1 \subseteq L_2$ if the string location L_1 includes the string location L_2 as a prefix (without wildcards).

Definition 2 (OSGi Platform). *The platform Θ is a tuple $\langle \mathcal{B}, \mathcal{S}, \mathcal{R} \rangle$ where \mathcal{B} is a set of bundles on the platform, $\mathcal{S} \subseteq \Delta_{\mathcal{S}}$ is a set of services on the platform, and $\mathcal{R} \subseteq \mathcal{B} \times \mathcal{S}$ is a service provision relation such that for each service $S \in \mathcal{S}$ there exists only one bundle $B \in \mathcal{B}$ such that the pair $(B, S) \in \mathcal{R}$, $\text{state}(B) \in \{\text{Active}, \text{Starting}, \text{Stopping}\}$ and $S \diamond P$ such that $P \diamond B$ and $P \in \text{exports}_{\mathcal{B}}$.*

Thus, in the model we consider that a service can be provided only by a bundle in an appropriate state that exports a package containing the definition of this service. We will denote the fact that bundle B can provide service S (exists package P such that $S \diamond P$ and $P \in \text{exports}_{\mathcal{B}}$) as $S @ B$.

We want to ensure that bundles interact on the platform in compliance with the pre-defined security policies set by the bundle owners. Thus, we start with the definition of bundle interaction. Informally, two bundles interact if one of them imports an exported package from another, or consumes a service provided by the other bundle.

Definition 3 (Bundle Interaction). Let $B_1, B_2 \in \mathcal{B}$. We say that B_1 interacts with B_2 , denoted $B_1 \bowtie B_2$ if at least one of the following conditions is satisfied:

- *Exists $P \in \text{exports}_{B_1} \cap \text{imports}_{B_2}$ such that $P \diamond B_1$ – there exists a package exported by B_1 and imported by B_2 ;*
- *Exists a local permission $\text{perm} = \langle \text{GET}, S \rangle \in \text{Permissions}_{B_2}$ where S is a service such that exists a package $P: S \diamond P$ and $P \in \text{exports}_{B_1}$ – there exists a local permission of bundle B_2 to get a service reference from bundle B_1 .*

This definition captures a *potential direct control flow among bundles*.

Functionality guarantees on the OSGi platform can vary. An interesting scenario was discussed in Example 1, where a bundle wants to have some functional requirements fulfilled prior to be loaded. The capabilities approach currently explored on the OSGi platforms is purely static and declarative. We want to enhance it with the dynamics of evolving platforms and with the guarantees given by the framework itself rather than by (potentially untrusted) bundles.

In Example 1, the C bundle wants to be installed on the platform only if a specific service is already provided there. While the presence of the service interface definition can be (limitedly) ensured by the capabilities approach, only the platform itself can assure that the service is indeed provided, or a certain bundle is in a desired state, or that a competitor's bundle is not installed at all. If later, when the bundle will already be installed, the platform will evolve such that the desired service will be unregistered or an undesired bundle appear, the bundle can be notified about it through the event system and take the actions it needs to protect itself (be removed, stop, notify the provider, etc.).

There is also another interesting problem that can be considered. The primary purpose of the requirement-capabilities model is to provide an explicit assertion about the environment before a bundle becomes active and its code starts to run. This prevents bundles that cannot run because they are not suitable for a given environment from becoming active, or even installed, when this header is used by a management system. The S×C framework can become the management system that will assure bundles they will never enter even the Installed state on a platform that is not suitable for them.

Contracts and Policy. The claim of a bundle (sufficient to cover the security and functionality issues discussed above) can be easily extracted from the bundle's manifest file and permissions file. Thus, the ClaimExtractor component duties will be to extract this information. The policy of a bundle is a new component specified in the contract that requires a permission notation and a notation for functional requirements.

Definition 4. Let B be a bundle. The Contract_B is a tuple $\langle \text{sec.rules}_B, \text{func.rules}_B \rangle$, such that:

- sec.rules_B is a set of permissions of the form $\langle \text{Action}, \text{Target}, \text{Authorized entity} \rangle$, that specifies the security policy on the usage of B 's packages and services, where

- Action $\in \{\text{IMPORT}, \text{GET}\};$
- Target $\in \bigcup_{P \circ B} P \cup \bigcup_{S @ B} S;$
- Authorized entity $\in \Delta_B \cup \Delta_C \cup \Delta_{\text{Sign}};$

• **func.rules_B** is a set of functional requirements of B of the form $\langle \text{Desired state}, \text{Flag}, \text{State} \rangle$, that specifies the requests of the bundle for functionality available on the platform, where

- Desired state $\in \{\text{Present}, \text{Not present}\};$
- Flag $\in \{\text{Bundle}, \text{Package}, \text{Service}\};$
- State differs in the following fashion:
 - Flag = Bundle, then State $\in \{\text{Installed}, \text{Resolved}, \text{Starting}, \text{Active}, \text{Stopping}\};$
 - Flag = Package, then State $\in \{\text{Present}, \text{Exported}\};$
 - Flag = Service, then State $\in \{\text{Present}, \text{Provided}\};$

Using the above notations bundles can express various security and functional requirements on other bundles on the platform. Bundles can be installed on the platform if and only if all their security and functional requirements are satisfied and their behavior is compliant with the policies of all other bundles on the platform.

We propose the bundle contract to be delivered within its manifest file by using the possibility to define new manifest file headers in the common header syntax. The newly defined headers processed by the S×C manifest file parser (the ClaimExtractor), are **sxc-secrules** and **sxc-funcrules**. The first header specifies the bundle's **sec.rules** separated by commas (elements of permissions are separated by colons). The second header denotes the bundle's **func.rules** separated by commas (elements of requirements are separated by colons).

The security policy of the platform is defined as follows.

Definition 5 (Security Policy of the Platform). For a platform Θ its security policy $\text{Policy}_\Theta = \bigcup_{B \in \mathcal{B}} \text{sec.rules}_B$.

Example 2. Let us consider the running example from Section 3. The bundles A , B and F are installed on the platform. The security policy Policy_Θ of the Alice's platform equals to $\{\text{sec.rules}_A, \text{sec.rules}_B, \text{sec.rules}_F\}$. The contracts of the bundles can be derived from their requirements Thus, $\text{sec.rules}_F = \{\emptyset\}$.

$\text{sec.rules}_A = \{\langle \text{GET}, S_A, FSM.com \rangle\}$. $\text{sec.rules}_B = \{\langle \text{GET}, S_B, BH.fr \rangle, \langle \text{IMPORT}, P_B, BH.fr \rangle, \langle \text{GET}, S_{fr}, BH.fr \rangle, \langle \text{GET}, S_{fr}, FB.com \rangle\}$.

For a platform Θ its *functional state* is at any given moment of time defined by the platform itself: the installed bundles and provided services.

The S×C Checks.

Definition 6. Let Θ be an OSGi platform and B is a loaded bundle. We say Θ can host B securely iff the following conditions are satisfied:

- **Stable Security.** For all bundles $A \in \mathcal{B}$ if $A \bowtie B$ then a corresponding permission for B exists in sec.rules_A , and if $B \bowtie A$ then a corresponding permission for A exists in sec.rules_B .

• **Stable Functionality.** All functional requirements described in func.rules_B are satisfied by Θ .

We note that the stable functionality property may not hold for some other bundle A immediately after the installation of bundle B on the platform. However, A will be notified about the situation and can take appropriate actions.

These are some of the checks (full list can be found in [4]) to be executed by the PolicyChecker. If B imports a package P of A , the check is “ $(\text{IMPORT}, P, B) \in \text{sec.rules}_A$ ” (the source of information is the manifest and the permissions files of B). If B requires a bundle A to be present on the platform in Active state ($(\text{Present}, \text{Package}, P, \text{Exported}) \in \text{func.rules}_B$), the check is “in the current functional state exists bundle $A \in \mathcal{B}$ such that $\text{state}(A) = \text{Active}$ ”. It can be easily demonstrated (proof by cases) that if the necessary checks for a new bundle B are performed by the S×C framework, then the platform can host B securely.

We can note here that the permissions file could be a weak source of information, as it may only require AllPermission, letting the system to define the upper bound of permissions for the bundle. This problem can be solved by awareness of the developers that their bundles will be rejected if the required permissions will be too demanding.

5 Evaluation

The evaluation of the proposed framework was conducted in collaboration with industry experts from Telefónica. Two top-level criteria were defined: effective usage (which includes applicability and human effort), and specific industrial criteria (level of automation).

Applicability. The OSGi framework presents no obstacles for the implementation of such a system as a bundle.

Human Effort. It is expected that the basics of S×C methodology will require a relatively low effort for a developer to learn how to create contracts. We estimate that the deployment of S×C will require very little effort for the operator of a network of home gateways. Installing a bundle (such as the S×C framework) on an OSGi platform is typically a routine task that presents few challenges. The S×C deployment should be transparent to and require no effort from the users of a gateway, except from perhaps providing confirmation for a system update.

Automation. The S×C is expected to work on a fully automated manner, inspecting bundle contracts and enforcing contract policies without need for user interaction. The generation of contracts, however, cannot be automated in the current framework. As a future development, it would be interesting to find ways to automate contract generation, or at least to have automated tools that guide developers in the process.

Overall, the current S×C specification for OSGi platforms looks like a promising starting point, which has some immediate applications as well as some very interesting research lines which will need to be studied in depth. One significant upside of the S×C methodology is the fact that it’s optional and backwards

compatible, which means that service providers do not need to immediately incorporate contracts into each running service as soon as S×C is adopted, but can introduce them gradually as new bundle versions are released. This allows careful planning of S×C contracts and better distribution of the workload.

6 Related Work

Though the OSGi technology is gaining popularity today, there are not so many papers dedicated to the OSGi security. Moreover, these papers are not applicable to the scenarios and threats we considered³.

The proposal by Parrend and Frénot [10] on installation time verification for the OSGi bundles is the closest to our current work. The authors advocate the installation time static analysis of bundle code that could replace the time-consuming run-time checks for given permissions. Their Component-based Access Control (CBAC) mechanism allows to parse the bytecode of the loaded bundle and check that all the sensitive methods it invokes are allowed. The testing of the CBAC tool concluded that the overhead of the installation time verification is insignificant in comparison with the run-time checks. However, the drawbacks of the approach are the same as were concluded for the OSGi framework: the bundle providers are still not entitled to defining their own policies.

There are a number of permission-based security frameworks for service platforms outside of the OSGi community. We can mention the works of Enck et al. [6] and Nauman and Khan [7] on the loading time mobile code certification for Android. While strengthening the security of the framework, these approaches, in contrast to our solution, leave the non-trivial task of defining the policy for the user and can result in non-functioning software. The needs of the Android applications to be able to regulate how they interact with other applications were advocated by Ongtang et al. [9]. They have proposed Saint – an extension of the Android platform that governs installation time permission assignment and controls the run-time use of permissions.

The Security-by-Contract paradigm that had inspired our proposal was investigated and implemented by Bielova et al. for mobile Java-based devices [3] and by Dragoni et al. for the Java Card platforms [5]. Our current work improves previous approaches by adapting the S×C scheme to the systems with permission-based security management, thus allowing it to be easily adopted on similar platforms (e.g., Android, Chrome).

7 Conclusions

In this paper we have presented a Security-by-Contract paradigm for the OSGi platforms. We gave an overview of the OSGi framework and described how the the OSGi platforms can benefit from the S×C approach. The paper contains the

³ While here we present only the most related work, a more thorough review can be found in the full version of the paper [4].

following contributions: (1) the proposal how to enable the bundle providers with ability to effectively express their security and functional requirements on the platform; (2) a formal model of an OSGi platform and the notations for security and functional requirements of bundles; (3) the S×C architecture for load time verification on OSGi.

The main benefits that the S×C approach can bring to OSGi platforms are the following. From the security aspect, the bundle providers can now specify the authorizations for access to their bundles, packages and services. The policies can be updated easily and the update does not require an interaction from the platform owner, an access to the framework policy file or an update of the execution logic of the bundle. For the functionality aspect, the bundle providers have now a more powerful tool for expressing their functional requirements than the requirement/capability model of OSGi. The contracts can express requirements on the current state of the platform (including requirements on the states of the bundles or certain services provision, or absence of the competitor's resources).

Acknowledgements. We would like to thank Telefónica for sharing the case study and evaluating our proposal. This work was partially supported by the EU under grants EU-FP7-FET-IP-SecureChange (grant n. 231101), FP7-IST-NoE-NESSOS (grant n. 256980) and EU-FP7-ICT-IP-ANIKETOS (grant n. 257930).

References

1. The OSGi Alliance. OSGi service platform core specification. Version 4.3 (2011)
2. Belimpasakis, P., Michael, M., Moloney, S.: The home as a content provider for mash-ups with external services. In: CCNC 2009, pp. 1–5 (2009)
3. Bielova, N., Dragoni, N., Massacci, F., Naliuka, K., Siahaan, I.: Matching in security-by-contract for mobile code. *Journal of Logic and Algebraic Programming* 78(5), 340–358 (2009)
4. Capelastegui, P., Gadyatskaya, O., Massacci, F., Philippov, A.: Security-by-Contract for the OSGi framework. Technical Report DISI-12-002, DISI, University of Trento, Italy, <http://www.disi.unitn.it/~gadyatskaya/osgi.html>
5. Dragoni, N., Lostal, E., Gadyatskaya, O., Massacci, F., Paci, F.: A load time Policy Checker for open multi-application smart cards. In: Proc. of POLICY 2011, pp. 153–156 (2011)
6. Enck, W., Ongtang, M., McDaniel, P.: On lightweight mobile phone application certification. In: CCS 2009, pp. 235–245. ACM, New York (2009)
7. Nauman, M., Khan, S.: Design and Implementation of a Fine-grained Resource Usage Model for the Android Platform. In: IJAIT (2010)
8. Ngu, A., Carlson, M., Sheng, Q., Paik, H.: Semantic-based mashup of composite applications. *IEEE Tran. on Services Computing* 99, 2–15 (2010)
9. Ongtang, M., McLaughlin, S., Enck, W., McDaniel, P.: Semantically rich application-centric security in Android. In: Proceedings of ACSAC 2009, pp. 340–349 (2009)
10. Parrend, P., Frénot, S.: Component-Based Access Control: Secure Software Composition through Static Analysis. In: Pautasso, C., Tanter, É. (eds.) SC 2008. LNCS, vol. 4954, pp. 68–83. Springer, Heidelberg (2008)
11. Phung, P., Sands, D.: Security policy enforcement in the OSGi framework using aspect-oriented programming. In: COMPSAC 2008, pp. 1076–1082 (2008)

Cyber Weather Forecasting: Forecasting Unknown Internet Worms Using Randomness Analysis ^{*}

Hyundo Park¹, Sung-Oh David Jung², Heejo Lee¹, and Hoh Peter In¹

¹ Korea University, Seoul, Korea

{hyundo95, heejo, hoh_in}@korea.ac.kr

² Trinity International University, IL, USA

zsjung@tiu.edu

Abstract. Since early responses are crucial to reduce the damage from unknown Internet attacks, our first consideration while developing a defense mechanism can be on time efficiency and observing (and predicting) the change of network statuses, even at the sacrifice of accuracy. In the recent security field, it is an earnest desire that a new mechanism to predict unknown future Internet attacks needs to be developed. This motivates us to study forecasting toward future Internet attacks, which is referred to as CWF (Cyber Weather Forecasting). In this paper, in order to show that the principle of CWF can be realized in the real-world, we propose a forecasting mechanism called FORE (FOrecasting using REgression analysis) through the real-time analysis of the randomness in the network traffic. FORE responds against unknown worms 1.8 times faster than the early detection mechanism, named ADUR (Anomaly Detection Using Randomness check), that can detect the worm when only one percent of total number of vulnerable hosts are infected. Furthermore, FORE can give us timely information about the process of the change of the current network situation. Evaluation results demonstrate the prediction efficiency of the proposed mechanism, including the ability to predict worm behaviors starting from 0.03 percent infection. To our best knowledge, this is the first study to achieve the prediction of future Internet attacks.

Keywords: Forecasting, Internet worm, Randomness check, Regression analysis, Reliability check.

1 Introduction

In the recently security field, various intelligent Internet attacks are being developed and they cause fatal damages on the Internet. In cases of previous fatal Internet incidents, we experienced that an Internet worm has an extremely high destructive force and devastates the Internet. Since the destructive power of a worm

* This work was partially supported by Seoul City R&BD program WR080951 and the National Research Foundation of Korea (NRF) grant funded by the Korean government (MEST) (2009-0086140).

is caused by its propagation speed, faster worm propagation techniques have been used in an Internet attack. For example, when a bot master constructs a botnet with compromising other hosts, a bot master can use a worm propagation technique. As a result, it is crucial to cope with a worm at an early stage of a worm propagation.

So far, many researchers have been made a commitment to develop worm detection approaches. Though their efforts, we can decide whether our network is under a worm propagation situation or not. In general, approaches to detect a worm are classified into two categories: signature based and anomaly based. In general, a signature based worm detection approach is widely used on most systems. The reason is that these approaches have the advantage of low false-positive rates. However, they cannot handle an unknown worm properly. Unlike a signature based worm detection approach, an anomaly based worm detection approach detects an unknown worm [1, 4, 5, 9, 10, 14–16]. They enable us to recognize even an unknown attack by examining the network situation whether it is normal or not. In order to do that, the most of detection approaches evaluate particular metrics as numeric values, representing the network situation. Unfortunately, when the worm just began its existence on the Internet, most anomaly worm detection approaches cannot detect it because they need at least a certain amount of attack traffic to decide whether the network situation satisfies the condition of an abnormal network situation or not. Moreover, the result of an anomaly detection mechanism is based on a binary decision, ‘true’ or ‘false’ (whether the network is under an attack or not). With the only result of an anomaly detection approach, we cannot observe the changing process of the network situation under a worm propagation. Observing the change of the network situation, from being benign to being a worm propagation recognized by an anomaly detection approach, will be very useful to cope with a worm. In other words, if we can observe the process of the change of the network situation before recognizing as a worm propagation (e.g., after one hour, the situation will turn out to be in worm propagation), it will be very useful to cope with a worm distinctly.

The concept of forecasting is to estimate future statuses by examining and analyzing current information [3]. There are several examples of forecasting such as a sales/stock forecasting in the business or a weather forecasting in the meteorology. Thus, we have a hypothesis that forecasting future statuses of the network situation by analyzing the current status of the network situation can be accomplishable. Forecasting future trends of the network situation is the concept of *CWF* (Cyber Weather Forecasting). If we analyze current trends of a worm propagation, we can estimate future trends of a worm propagation. The reason is that the number of infected hosts and attack packets generated by them is continuously increased but not decreased as time passes, as shown in cases of Cod-Red and Slammer [2, 13, 17, 18]. In their studies, we divide the process of a worm propagation into three stages as follows;

- *SP (Starting Point)*: this stage is very early stage of a worm propagation and the symptom caused by a worm propagation is a subtle sign.
- *PS (Processing Stage)*: a worm propagation is in progress and the symptom caused by a worm propagation is getting clearer but it is still a subtle sign.

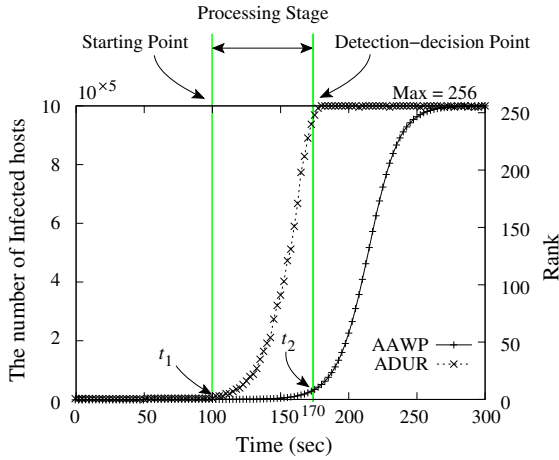


Fig. 1. Three phases of an early worm detection mechanism; Starting Point, Processing Stage and Detection-decision Point.

- *DP (Detection-decision Point)*: the symptom caused by a worm propagation is clear enough to recognize a worm propagation situation by an anomaly worm detection approach.

Fig. 1 shows that ADUR (Anomaly Detection Using Randomness check), proposed by Park *et al.* as one of early worm detection approaches based on the threshold, detects the worm at an early stage of the worm propagation [9, 10]. In Fig. 1, the worm propagation is based on the AAWP (the Analytical Active Worm Propagation) model, one of the popular models for the Internet scale worm propagation [2]. As shown in Fig. 1, ADUR has three phases (*SP*, *PS* and *DP*) under a worm propagation. It is pretty much the same as other worm detection approaches based on the threshold. In Fig. 1, the rank value, estimated from ADUR, is maintained in low values under the benign network situation or the very early worm propagation stage (before *SP*). But the symptom of a worm propagation becomes clearer under *PS* (after *SP* until *DP*) and the value increases more and more. Finally, when the symptom caused by a worm propagation becomes clear, the value goes over the threshold at *DP*. When our network situation is under *SP* or *PS*, *DP* will be the future network situation. Nevertheless, *CWF* forecasts *DP* with analyzing the current trend of the change of the network traffic under *PS*. To forecast *DP*, *CWF* decides whether the current network situation is *SP* or not. If our network situation is *SP*, our network situation turns to *PS* and *CWF* forecasts *DP* with the time from the current time until *DP* as shown in Fig. 1. In Fig. 1, at time t_1 , the current network situation is *SP* and the *CWF* forecasts the future network situation with $\Delta t = t_2 - t_1$. As time passed, t_1 will be closed to t_2 and *CWF* still forecasts with Δt until *DP*.

In this paper, we propose a new conceptual model of *CWF*, and we show that *CWF* can be realized in the real-world with applying statistical theories.

To show that *CWF* can be accomplished in the real-world, we provide the value, the prediction of attack situation by analyzing the values taken from ADUR. The evaluation results of our forecasting approach, called FORE (Forecasting using REgression analysis), show that FORE can predict the Internet worm 1.8 times faster than ADUR does, irrespective of the number of vulnerable hosts or the worm scan rate. Furthermore, FORE continuously forecasts Δt under *PS*. This result shows that predicting and alerting an unknown attack is possible and this predicted information will be invaluable for a security agencies.

This paper is organized as follows. In Section 2, we give an in-depth account of *CWF*. In Section 3 and Section 4, we explain the ADUR and FORE mechanism. Section 5 demonstrates the effectiveness of the FORE with the evaluation results. Finally, we conclude with a summary of contribution and discuss future works.

2 Cyber Weather Forecasting

In this paper, to help us better understand *CWF*, we explain the limited *CWF* within which predicts a worm propagation.

We can get information of a current network situation related to a worm propagation using an early worm detection mechanism. Nevertheless, we always have curiosities toward the future network situation, such like how and when the current network situation will be changed to the worm propagation network situation in the future. Unfortunately, since most early worm detection mechanisms merely estimate the network situation with the binary decision, such as ‘true’ or ‘false’, we cannot investigate how the network situation is being changed during the period from triggering the worm propagation to detecting it. As a result, developing the new model which can continuously report a possibility of the worm epidemic before when the early worm detection mechanism detects the worm is earnestly desirable, even at the sacrifice of accuracy.

In order to satisfy our desires, we propose the model of *CWF* in this paper. *CWF* is the conceptual model that forecasts the change of the future network situation. *CWF* forecasts when the current situation will be clearly changed to the worm epidemic situation and reports the changing process of the network situation during the period from triggering the worm to detecting it. Therefore, *CWF* helps us to get the information of the future network situations (reporting the trend of worm propagation under *PS* and forecasting the time of *DP* with Δt)

In the view points of forecasting the future Internet worm, Nazario *et al.* proposed a mathematical model for a vulnerability’s “wormability”, the potential for the use of the vulnerability in worm propagation [8]. The model provides time intervals between the publication of the vulnerability and the current time; and the model measures the possibility of several vulnerability announcements in the near future along with explanations of why some vulnerabilities are inherently not “wormable”. On the other hand, Sanguanpong *et al.* proposed the approach to minimize the damage due to worm infection in enterprise networks [12]. The authors predict the number of infected nodes by fuzzy decision, thus benefiting to the study of worm behaviors.

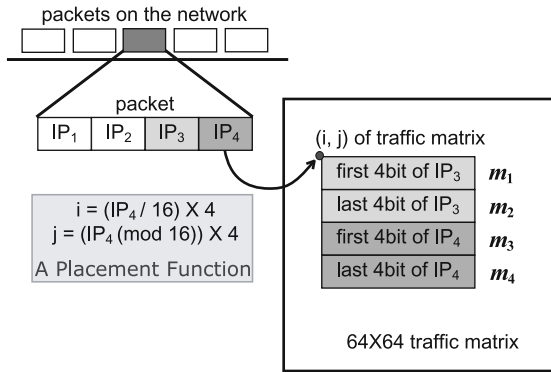


Fig. 2. Constructing matrix M by mapping a packet to submatrix m

Unfortunately, these studies cannot be included in *CWF*. Since previous studies predict the worm propagation using theoretic models, related to previous worm trends and worm propagation behaviors, and it is absolutely impossible to find the point when the unknown worm propagation will be triggered on the Internet, these studies are useful in analyzing already known worms only.

To realize *CWF* in the real-world, two types of information which are used to predict the worm should be reported to us as follows:

- *BP* (Branch Point) : reporting the starting time of forecasting the worm (i.e., *SP*).
- *RT* (Remaining Time) : reporting the remaining time from the current to the future worm epidemic situation (i.e., Δt).

In Fig. 1, if the worm propagation is triggered on the Internet, *CWF* senses the change of the network situation with the tiny symptom of the worm propagation, even at the sacrifice of accuracy. Thus, *CWF* reports *BP* (as *SP*) and *RT* (as Δt)

3 ADUR (Anomaly Detection Using Randomness Check): The Context

Since we propose a forecasting model called FORE which analyzes the randomness of the network traffic, we first explain ADUR in concrete. ADUR, proposed by Park *et al.* [9, 10], is the anomaly worm detection mechanism using randomness checks of the network traffic. In order to do that, ADUR uses the matrix operation after constructing the traffic matrix. In order to check the randomness of the network traffic, ADUR uses the well-known binary matrix rank test, proposed by Marsaglia *et al.* [6, 7].

Since one prominent characteristic of Internet worms is a random selection of subsequent targets, ADUR examines the random distribution of destination addresses in the network traffic. ADUR consists of four steps: 1) traffic matrix construction, 2) XOR operation, 3) rank calculation, and 4) randomness

checkup. In order to obtain more accurate attack information such as a randomness of either entering or departing worm traffic from the network, ADUR classifies traffic data into two categories based on their direction: incoming and outgoing. ADUR constructs the traffic matrix, a binary square matrix, to check a randomness of traffic, because checking for randomness of a binary square matrix is easily accomplished by measuring the rank of the matrix. As well, ADUR uses a XOR operation to diminish the effect of normal traffic on rank values.

First, ADUR constructs the traffic matrix as Fig. 2. To represent the traffic information (the binary information of third and fourth octets of IP address) on the traffic matrix, ADUR finds the location, i and j of the traffic matrix, with the fourth octet of IP address of a packet and overwrites the binary information of third and fourth of octets of IP address of a packet on the submatrix (4×4) in the traffic matrix (64×64). As a result, a randomness of traffic can be properly expressed on the matrix. In previous works, ADUR considers the network environment as IPv4 when ADUR constructs the traffic matrix. To apply ADUR to the IPv6 network environment, we should adopt a new method to find the location, i and j of the traffic matrix, and the function to find the location should satisfy three conditions (it should be easy to compute the index for any given IP address, difficult to find an IP address that maps to a given index value, and difficult to spoof the IP address without the index value being changed). As a result, under the IPv6 network environment, we can adopt a cryptographic hash function to find the location. In this paper, we consider the network environment as IPv4.

Second, since the legitimate traffic can reduce accuracy of the worm detection, a simple XOR operation is very efficiently used to remove the legitimate traffic on the traffic matrix.

$$R(M'_t) = R(M_t \oplus M_{t-1}) \quad (1)$$

where M_t is the traffic matrix at time t . Eq. (1) presents the XOR operation on a sequence of matrices. Let M'_t denote the result of the XOR operation of two consecutive matrices, M_t and M_{t-1} . Then, the XOR operation eventually removes most of legitimate traffic in the traffic matrix M'_t because legitimate traffic lives longer than one time unit so the portion of legitimate traffic is eliminated by the XOR operation. This effectiveness of the XOR operation will be constant even when the network state goes the busy state (such like the traffic volume increases highly) caused by huge legitimate users (flash crowds) [11].

Third, the common dimension of the row space and the column space of a matrix is called the rank. A straightforward method to compute a rank of a matrix is to count the number of non-zero rows after applying the Gaussian elimination to the matrix. The rank of the matrix is equal to the number of non-zero rows (or equivalently, the number of leading 1's) on the matrix. In Eq. (1), $R(M'_t)$ is the rank value of M'_t .

Finally, when the worm propagates on the Internet, the traffic has randomness. Thus, ADUR detects the worm with the rank value of the traffic matrix. The rank value of the 64×64 random matrix exceeds 60 with the probability which

is greater than 99.999%. This implies that, if the 64×64 binary matrix is a random matrix then the rank has a high probability of being greater than 60. To summarize, the rank of the matrix can be used to determine the randomness of distribution of the elements on the matrix. This 64×64 matrix will undoubtedly be used when a /24 network is monitored [10]. When we monitor the network larger than a /24 network, we should enlarge the size of matrix [9]. However, in this paper, we use a 64×64 binary matrix as early research to predict unknown worms.

ADUR effectively detects the Internet worm at an early stage of worm propagation. It is shown that ADUR is highly sensitive so that the worm epidemic can be detectable quickly, *e.g.* three times earlier than the infection of 90% vulnerable hosts [9, 10] which is evaluated from AAWP model [2]. And when ADUR detects the worm epidemic, the number of infected hosts was only one percent of the total number of vulnerable hosts. It means that ADUR can detect the worm epidemic before the fast spread phase of the worm propagation, represented by Zou *et al.* [18].

4 FORE (FOrecasting Using REgression Analysis) : The Proposed Model

Here we propose a forecasting model called FORE. FORE finds BP when the worm emerges from the Internet and predicts RT .

FORE consists of three steps: 1) time series analysis, 2) linear regression analysis, and 3) reliability analysis. In these three steps, step 1) and step 2) are used to estimate RT , and step 3) is used to find BP . FORE does not use out-rank values (of out-coming traffic matrix of ADUR) but in-rank values (of in-coming traffic matrix of ADUR) because out-rank values are only useful to detect the worms originating from the monitored network.

4.1 Time Series Analysis

In time series analysis, a moving average can be used for smoothing the scattered in-rank data. Eq. (2) shows the moving average of rank values

$$\bar{R}_i = \left(\sum_{j=i-t}^i R_j \right) / t \quad (2)$$

where R_i , t , and \bar{R}_i are the in-rank value at time i , the time interval to calculate average, and the average of in-rank values from $i - t$ to i , respectively.

4.2 Linear Regression Analysis

Linear regression analysis yields the equation of the fitted line of the in-rank values of the incoming traffic matrix. As seen in Eq. (3), the equation is used to

predict RT . The best feature of worm propagation behavior is that the number of infected hosts does not decrease but increases in case that the self-propagation worm spreads over the Internet [2, 13, 17, 18]. Therefore, the equation of the fitted line of the in-rank values is used for calculating the in-rank values at a specific time and vice versa.

$$R_i = \alpha + \beta T_i \tag{3}$$

where R_i and T_i are the in-rank value at time tick i , and the time at i , respectively.

$$\beta = \left(\sum_{j=i-t}^i (R_j - \bar{R}) (T_j - \bar{T}) \right) / \left(\sum_{j=i-t}^i T_j - \bar{T} \right) \tag{4}$$

$$\alpha = \bar{R} - \beta T_i \tag{5}$$

where \bar{R} and T are an average of results of the moving average of in-rank values on time interval t and the average of time on time interval t , respectively. We can get α and β using Eq. (5) and Eq. (4) at time tick i . With Eq. (3) based on α and β , FORE predicts when the rank value goes over the threshold. In this paper, since the threshold of ADUR is 60, R_i of Eq. (3) will be 60 so FORE can forecast RT (as the value of $\Delta t = T_i - t_i$ where t_i is the current time).

4.3 Reliability Analysis

In the reliability analysis, the coefficient of determination measures the reliability of the equation of the fitted line, and measuring the reliability represents the trust level of predicting the detection decision point when ADUR detects the worm. Added to that, the reliability is used to find the BP to start predicting the worm.

$$C_i = \left(\sum_{j=i-t}^i (\widehat{R}_j - \bar{R})^2 \right) / \left(\sum_{j=i-t}^i (R_j - \bar{R})^2 \right) \tag{6}$$

where C_i and \widehat{R}_i are the reliability of the fitted line and the in-rank value on the fitted line after applying the moving average of original in-rank values, respectively. As shown in Fig. 3, in Stage 1 (before BP), since rank values merely oscillate, the reliability of the similarity between the fitted line (obtained using Eq. (3)) and in-rank values is very low (not over 0.8 as shown in Fig. 3). In contrast, in Stage 2 (during PS), since the number of infected hosts and attack packets generated by the worm is continuously increased but not decreased as time passes, the reliability of the similarity between the fitted line and in-rank values is very high (over 0.8 as shown in Fig. 3). According to this phenomenon, FORE finds BP using this difference among the reliability values (as C_i in Eq. (6)).

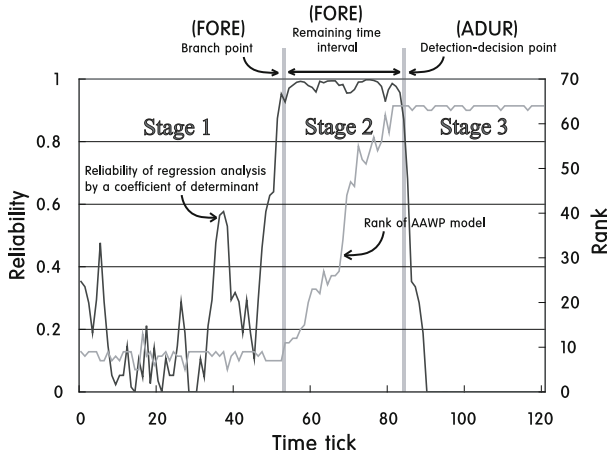


Fig. 3. The estimated branch point using reliability analysis

All these analyses are performed using the in-rank values within the contiguous time interval; the time interval exists between the previous time and the current time. The previous time must be continuously selected while reflecting network environment.

In this section, first, FORE decides whether the current time is *BP* or not using the reliability analysis. If the current time is *BP*, FORE finds the fitted line and predicts *RT* using regression analysis. In the following section, we show evaluation results of FORE where FORE predicts the worm effectively. Thus, it will be confirmed that CWF can be accomplishable.

5 Evaluation Results

FORE is evaluated using the real worm traffic by generating as many as the number of infected hosts, being estimated using the AAWP model. In the model, it is assumed that the number of vulnerable hosts is one million, and the monitored network is a /24 network. In order to evaluate the efficiency and reliability of FORE, time-series analysis and regression analysis are employed using real traffic dump data; and the data is comprised of packets captured on the backbone of a university campus network in July 29, 2004, a /16 campus network traffic trace where the average number of clients and packets per second are 132 and 1,847 during the time spam of 187 seconds. And the most busy /24 network trace for background traffic is used to evaluate FORE during 180 seconds. The traffic matrix is constructed with one second time unit. The time unit of the moving average and the regression analysis is 10 seconds.

As shown in Fig 3, the worm propagation is divided by three parts: the Stage 1, the Stage 2 and the Stage 3.

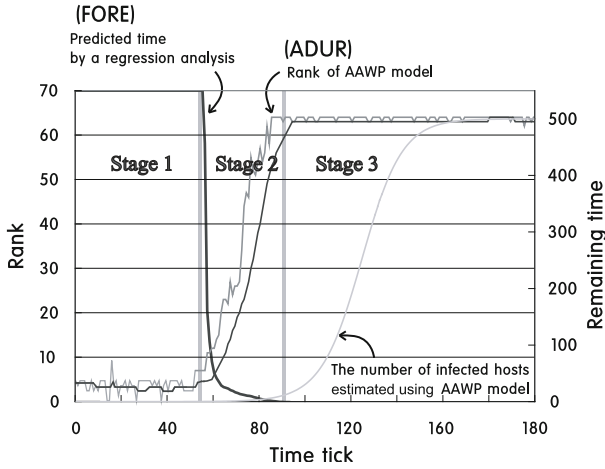


Fig. 4. Trajectory of the remaining time predicted by FORE

- Stage 1 : from the start time to *BP* (*i.e.*, before *SP* in Fig. 1).
- Stage 2 : from the branch point to *DP* (*i.e.*, during *PS* in Fig. 1).
- Stage 3 : after *DP* (*i.e.*, after *DP* in Fig. 1).

FORE estimates in-rank values using reliability analysis. As a result, FORE senses the tiny change of the network traffic by the worm, and decides the end of the Stage 1 and the start of the Stage 2 with *BP* which is drawn by the reliability analysis. Following the start of the Stage 2, ADUR checks whether the in-rank value goes over the threshold or not. If the in-rank value goes over the threshold, ADUR decides the end of the Stage 2 and the start of the Stage 3 using randomness checks. In our evaluations, the threshold of reliability is 0.8 drawn by experimental results. As shown in Fig. 3, FORE decides whether the current network situation is *BP* (as *SP* in Fig. 1) or not.

Fig. 4 plots in-rank values of ADUR, results after applying the moving average to in-rank values of ADUR, and remaining time values estimated by FORE. In Fig. 4, ‘Predicted time by a regression analysis’ shows the trajectory of the predicted *RT* as time passes. Since the moving average requires a contiguous time period (as 10 seconds in Fig. 4) to calculate the average of in-rank values, *RT* value does not present the correct *RT* at the early stage of Stage 2. But, after the early stage of Stage 2 (as after 63 time tick in Fig. 4), *RT* presents a correct value.

In Fig. 4, FORE starts to predict *RT* when only 0.03 percent of the total vulnerable hosts are infected. At this time, the number of infected hosts is merely 306 of the total number of vulnerable hosts. Furthermore, ADUR detects the worm when 1 percent of total vulnerable hosts are infected. In the same manner, the number of infected hosts is 10306 of total vulnerable hosts when ADUR detects the worm.

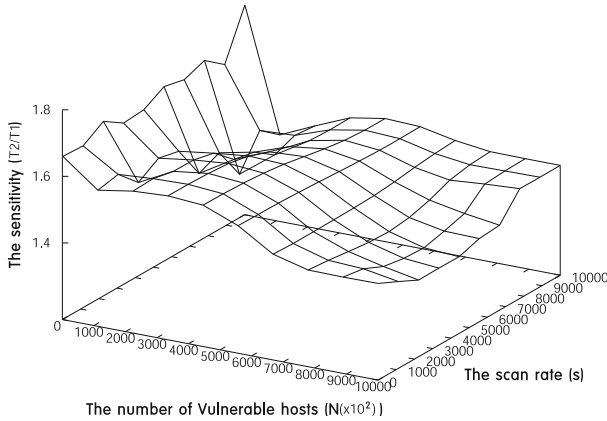


Fig. 5. Comparison of the sensitivity of FORE comparing with ADUR

Fig. 5 shows the FORE’s sensitivity, measured under the worm spreading state which is generated by AAWP model, where T_1 , T_2 , s , and N are BP by FORE, DP by ADUR, the worm scan rate per second, and the number of vulnerable hosts on the Internet, respectively. In Fig. 5, the faster the worm scan rate, the higher sensitivity FORE has. Likewise, the smaller the number of vulnerable hosts make FORE more sensitive. Fig. 5 shows that FORE predicts the Internet worm 1.8 times faster than ADUR does, irrespective of the number of vulnerable hosts or the worm scan rate.

6 Conclusions

For the purpose of realizing *CWF*, the FORE mechanism is proposed and evaluated for predicting worm epidemics with analyzing dynamics of a randomness in the network traffic. The evaluation results show that FORE responds 1.8 times faster than the early detection mechanism (e.g., ADUR) against unknown worms. As these results address, this study shows the incorporation of forecasting into the detection of unknown worms (e.g., zero-day worms). Furthermore, using FORE, we can observe the change of network situation before when the early detection mechanism (e.g., ADUR) detects the worm propagation situation with binary decision, ‘true’ or ‘false’. To the best of our knowledge, this is a new direction to predict the future worm epidemics based on the current state of the monitored network traffic. In addition to the worm prediction, we are sure that FORE can be applied to predict many other Internet attacks, if the attack changes the situation of the network traffic with time, unlike a normal network situation. And if the attack detection mechanism estimates the network situation with the value and the threshold, we are sure that FORE can be applicable for predicting future attacks in an earlier stage than any detection mechanism.

References

- [1] Berk, V.H., Gray, R.S., Bakos, G.: Flowscan: Using sensor networks and data fusion for early detection of active worms. In: SPIE AeroSense, vol. 5071, pp. 92–104 (2003)
- [2] Chen, Z., Gao, L., Kwiat, K.: Modeling the spread of active worms. In: IEEE INFOCOM (2003)
- [3] InvestorWords.com: What is forecast? definition and meaning, <http://www.investorwords.com/2038/forecast.html>
- [4] Jung, J., Paxson, V., Berger, A.W., Balakrishnan, H.: Fast portscan detection using sequential hypothesis testing. In: IEEE Symp. on Security and Privacy, pp. 211–225 (2004)
- [5] Leckie, C., Kotagiri, R.: A probabilistic approach to detecting network scans. In: IEEE Network Operations and Management Symposium (NOMS) (April 2002)
- [6] Marsaglia, G.: Diehard: a battery of tests of randomness, <http://stat.fsu.edu/~geo/diehard.html>
- [7] Marsaglia, G., Tsay, L.H.: Matrices and the structure of random number sequences. *Linear Algebra and Its Applications* 67, 147–156 (1985)
- [8] Nazario, J., Ptacek, T., Song, D.: Wormability: A description for vulnerabilities. Arbor Networks (October 2004)
- [9] Park, H., KIM, H., Lee, H.: Is early warning of an imminent worm epidemic possible? *IEEE Network* 23(5), 14–20 (2009)
- [10] Park, H., Lee, H., Kim, H.: Detecting unknown worms using randomness check. *IEICE Trans. on Communications* E90-B(4), 894–903 (2007)
- [11] Park, H., Li, P., Gao, D., Lee, H., Deng, R.H.: Distinguishing between FE and DDoS Using Randomness Check. In: Wu, T.-C., Lei, C.-L., Rijmen, V., Lee, D.-T. (eds.) *ISC 2008*. LNCS, vol. 5222, pp. 131–145. Springer, Heidelberg (2008)
- [12] Sanguanpong, S., Kanlayasiri, U.: Worm damage minimization in enterprise networks. *Int'l Journal of Human–Computer Studies* 65(1), 3–16 (2007)
- [13] Staniford, S., Paxson, V., Weaver, N.: How to own the internet in your spare time. In: *USENIX Security Symposium*, pp. 149–169 (August 2002)
- [14] Tong, X., Wang, Z.: A novel anomaly detection algorithm and prewarning technology of unknown worms. *Communications in Computer and Information Science* 163, 164–171 (2011)
- [15] Whyte, D., Kranakis, E., van Oorschot, P.: DNS-based detection of scanning worms in an enterprise network. In: *Network and Distributed System Security Symposium, NDSS* (2004)
- [16] Whyte, D., van Oorschot, P., Kranakis, E.: ARP-based detection of scanning worms within an enterprise network. In: *Annual Computer Security Applications Conference (ACSAC)*, pp. 5–9 (2005)
- [17] Wu, J., Vangala, S., Gao, L., Kwiat, K.: An efficient architecture and algorithm for detecting worms with various scan techniques. *NDSS* (February 2004)
- [18] Zou, C.C., Gong, W., Towsley, D., Gao, L.: The monitoring and early detection of internet worms. *IEEE/ACM Transaction on Networking* 13(5), 961–974 (2005)

Incentive Compatible Moving Target Defense against VM-Colocation Attacks in Clouds

Yulong Zhang¹, Min Li¹, Kun Bai², Meng Yu¹, and Wanyu Zang¹

¹ Computer Science Department, Virginia Commonwealth University
401 W. Main Street, Richmond, VA 23284, USA
{zhangy44, lim4, myu, wzang}@vcu.edu

² IBM T.J. Watson Research Center
19 Skyline Dr., Hawthorne, NY 10532, USA
kunbai@us.ibm.com

Abstract. Cloud computing has changed how services are provided and supported through the computing infrastructure. However, recent work [1] reveals that virtual machine (VM) colocation based side-channel attack can leak users privacy. Techniques have been developed against side-channel attacks. Some of them like NoHype remove the hypervisor layer, which suggests radically changes of the current cloud architecture. While some other techniques may require new processor design that is not immediately available to the cloud providers.

In this paper, we propose to construct an incentive-compatible moving-target-defense by periodically migrating VMs, making it much harder for adversaries to locate the target VMs. We developed theories about whether the migration of VMs is worthy and how the optimal migration interval can be determined. To the best of our knowledge, our work is the first effort to develop a formal and quantified model to guide the migration strategy of clouds to improve security. Our analysis shows that our placement based defense can significantly improve the security level of the cloud with acceptable costs.

Keywords: Cloud computing, Moving target defence, VM migration, VCG mechanism.

1 Introduction

Cloud computing [2] is becoming a major trend in computing services with its inspiring features of elastic “data anywhere” and “computing anywhere”. Generally, there are three types of cloud computing services: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS). Among them the IaaS is the most fundamental one, where a cloud user owns a virtual machine (VM) and purchases virtual power to execute as needed, just like running a virtual server. A typical example of public IaaS is the Amazon Elastic Computing (EC2) Services [3]. Although IaaS offers cost-efficiency and ease-of-use to cloud users, there are significant security concerns that need to be addressed when considering moving critical applications and sensitive data to the clouds. Recent work [1] reveals the problem of side-channel based attacks through virtual machine colocation, showing that the adversaries can map

the internal VM-placement of the cloud and mount cross-VM side-channel attacks by placing malicious VMs on the victim’s physical server.

Many software level approaches [10] and hardware modification methods [13,7] have been developed against the side-channel attacks. Unfortunately, the software approach cannot perfectly cover the new advances in attack models, and the hardware approach, modifying the cloud physical platform, is far from practical for commercial clouds. Using commercial off-the-shelf (COTS) components, the *Shamir’s secret sharing* approach [12] can make the attack much harder. We can split our secret D into k pieces and store them into k VMs¹. Using Shamir’s secret sharing, D can be easily constructed from all k pieces but knowledge less than k pieces reveals no information about D . Note that the secret sharing might be controlled by either a single party or multiple parties [2]. Thus, the attacker will be forced to use *brute-force-attacks* to achieve colocation with multiple target VMs, according to [11]. Now, the defense problems becomes how to protect the k pieces from being all captured by the attacker.

The methods to securely live-migrate VMs [4,8] can make it much harder for adversaries to locate the target VMs [6]. As shown in Figure 1, the secret, “helloworld”, is divided into two pieces and held by two VMs, initially as VM_0 and VM_2 . An attacker can launch VMs collocated with the benign VMs with some probabilities. For example, if VM_3 is controlled by an attacker, the attacker will be able to extract partial secret, “hello”, out of VM_0 . Although the splitting of “helloworld” into “hello” and “world” can prevent the attack from reconstructing the secret from VM_3 , we should at the same time guarantee that VM_5 can never be taken over by the same attacker. The fact that the attack would simultaneously attempt to launch lots of VMs to enumerate the colocations with both VM_0 and VM_2 motivates the migration of the VMs, e.g. moving VM_0 to VM_1 . Unfortunately, there is currently no formal model to guide the dynamic migration strategy for clouds. Similarly, quantifying the benefits of dynamism still remains an open problem.

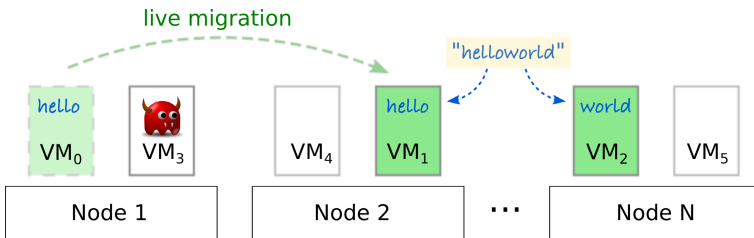


Fig. 1. Shamir’s secret sharing in cloud environment and the moving target defense through VM-migration

Intuitively, the cloud provider can offload the choice to cloud users, letting them migrating at free will. Or the cloud provider can migrate VMs according to the security demand - those VMs with higher security demand of dynamism have higher chances

¹ It is not necessary to have k identical VMs. We can have one service VM and $k - 1$ secret holding VMs that can be very small and cost little.

to be migrated. At first glance, this is reasonable. However, as long as some of the VMs sharing the secret are migrated, other VMs can take the free ride of the achieved dynamism. Since any rational cloud user will try to maximize their benefit², he/she could report a lower preference of dynamism than the actual one, pinning its hope on other VMs honest movements to reach the security goal. So in this case, the game equilibrium is that no one would truthfully report the security valuation and migrate.

Our Goal and Contributions: To solve the aforementioned problems, the goals of this paper are (1) to model the migration benefit and cost, (2) to provide instructional criteria for the migration strategy of the cloud provider, and (3) to motivate the cloud users to migrate in a incentive-compatible way. In order to address these issues, we develop a migration strategy based on the Vickrey-Clarke-Groves(VCG) mechanism^[9] in game theories, which can maximize the social welfare given the individuals are all “selfish”.

The contributions of this paper are as follows. To the best of our knowledge, (1) this is the first work to address VM-colocation based attacks in clouds using moving target defense; (2) our work is the first effort to apply VCG game and realize incentive compatible migration in cloud; (3) we offer two criteria about how to make the strategy acceptable by rational cloud users, and how to determine the optimal time interval of migrations, (4) our analytical evaluation shows that our defense approach is practical for commercial clouds.

This paper is organized as follows. In Section^[2] we review some of the fundamental concepts of game theory, especially the VCG mechanism. In Section^[3] we present our proposed method. In Section^[4] we evaluate the security and practicability of our scheme. We conclude the work in Section^[5].

2 Preliminaries

2.1 Assumptions and Notations

We have the following assumptions: (1) the cloud controller and the migration process are all secure; (2) for simplicity, each migration of a VM will result in a constant cost in terms of service interruption and consume the same amount of resources; (3) the cloud provider has enough CPU, network bandwidth, and other resources to perform arbitrary migration; (4) the cloud provider has sufficient resources as the reward, e.g., extra memory or CPUs, to motivate the players to migrate, which will be further discussed later; and (5) a node’s destination of migration is randomly chosen, as long as the cloud has free space. In reality, cloud provider would take performance and efficacy into consideration during the migration. For example, VMs with frequent connections should be placed close to each other. In this paper, for simplicity, we only measure security in the goal function, but a commercial cloud can freely modify the goal function as needed, where our game model remains the same.

The incentive problem of cloud migration can be modelled as a game. The specific game discussed here is a finite player, single round, simultaneous action, incomplete

² Note that even if all the VMs sharing the secret are controlled by one user, because different VMs may have different purposes, each VM should be treated as an independent party with individual interest.

information game, with payoffs depending on the final result of players' actions. In the cloud migration problem, the game players, P_1, P_2, \dots, P_n , are n VMs sharing a secret. The cloud provider (game mediator) is denoted as P_t , who doesn't participate in the game but operates the game. Players have actions which they can perform at designated times in the game, and as a result they receive payoffs. The actions of the players are denoted as $X = (x_1, x_2, \dots, x_n)$, and specially $X_{-i} = (x_1, x_2, \dots, x_{i-1}, x_{i+1}, \dots, x_n)$. The players have different pieces of information, on which the payoffs may depend; each player has his/her individual strategy to maximize his or her payoff. Each player $P_i \in \{P_1, P_2, \dots, P_n\}$ can decide whether to accept the migration request from P_t or not, so $x_i = \{accept, refuse\}$. For those who agree to migrate, the direct payment is \bar{p}_i , which measures the cost of downtime due to live migration. Here we assume the cost of downtime is a constant value for each P_i . As a result of migration, all the players get benefited. We measure the benefit as $v_i(X)$, where the valuation function v_i quantifies P_i 's real security demand of the overall dynamism. However, under the assumption of incomplete information, P_i can decide to report a fake valuation function v'_i at will. Finally, the utility function of each player is $u_i = v_i - p_i$.

2.2 Incentive Compatible Game

While our ultimate goal is to design a migration strategy to fulfil some security requirements, the problem is that in real-world cloud environment most of the cloud users are not willing to migrate their VMs unless it is necessary, because the migration will result in service interruption. So the main purpose of our work is to design an incentive compatible mechanism to mitigate this problem. Following Nisan's work [9], the terms "mechanism" and "incentive compatible" are defined as :

Definition 1. (Mechanism) [9] Given a set of n players, and a set of outcomes, A , let V_i be the set of possible valuation functions of the form $v_i(a)$ which player i could have for an outcome $a \in A$. A mechanism is a function $f : V_1 \times V_2 \times \dots \times V_n \rightarrow A$. Given the valuations claimed by the players, f selects an outcome, and n payment functions, p_1, p_2, \dots, p_n , where $p_i : V_1 \times V_2 \times \dots \times V_n \rightarrow R$.

Definition 2. (Incentive compatible) [9] If, for every player i , every $v_1 \in V_1, v_2 \in V_2, \dots, v_n \in V_n$, and every $v'_i \in V_i$, where $a = f(v_i, v_{-i})$ and $a' = f(v'_i, v_{-i})$, then $v_i(a) - p_i(v_i, v_{-i}) \geq v_i(a') - p_i(v'_i, v_{-i})$, then the mechanism is incentive compatible.

Specially, among those incentive compatible mechanisms, the Vickrey-Clarke-Groves (VCG) mechanism is the mostly used one. The VCG mechanism generally seeks to maximize the social welfare of all players in one game, where the social welfare is calculated as $\sum_{i=1}^n v_i$. So the goal function of VCG is $argmax_{a \in A} \sum_{i=1}^n v_i$. The VCG mechanism and the rule to design VCG mechanisms [9] are defined as:

Definition 3. (Vickrey-Clarke-Groves mechanism) [9] A mechanism, consisting of payment functions p_1, p_2, \dots, p_n and a function f , for a game with outcome set A , is a Vickrey-Clarke-Groves mechanism if $f(v_1, v_2, \dots, v_n) = argmax_{a \in A} \sum v_i(a)$ (f maximizes the social welfare) and for some functions h_1, h_2, \dots, h_n , where $h_i : V_{-i} \rightarrow R$ (h_i does not depend on v_i), $\forall v_i \in V, p_i(v_i) = h(v_{-i}) - \sum_{j \neq i} v_j$.

Definition 4. (Clarke pivot rule) [9] The choice $h_i(v_{-i}) = \max_{b \in A} \sum_{j \neq i} v_j(b)$ is called the Clarke pivot payment. Under this rule the payment of player i is $p_i(v_1, v_2, \dots, v_n) = \max_b \sum_{j \neq i} v_j(b) - \sum_{j \neq i} v_j(a)$, where $a = f(v_1, v_2, \dots, v_n)$.

3 VM-Migration Based Moving Target Defence

3.1 The Optimal Number of Moving VMs

At first glance, it appears that maximum dynamism can be achieved if all VMs migrate, but in this section we will disprove it and find the optimal number of moving VMs. As denoted in Section 2, each player has the unique valuation function $v_i(X)$, where $X = (x_1, x_2, \dots, x_n)$ is the vector of actions of all players. Intuitively, v_i has a positive correlation with the randomness or diversity of the VM placement. Suppose there are m physical nodes as the available candidates, and γ of them are randomly chosen as the migration destination, of which the capacities (the maximum number of extra VMs one node can host using its current free space) are $C = c_1, c_2, \dots, c_\gamma$. If k out of n VMs are randomly selected to be migrated ($k < \sum_{i=1}^\gamma c_i$), the number of possible placements, $N(m, \gamma, n, k, C)$ can be derived using the *Balls In Bins* analysis.

For selecting γ ones out of the m nodes, and selecting k ones out of the n VMs, the numbers of possible schemes are given respectively:

$$S(m, \gamma) = \binom{m}{\gamma}, T(n, k) = \binom{n}{k} \tag{1}$$

The generating function for placing k distinguishable VMs (balls) in γ distinguishable nodes (bins) is

$$GF = \prod_{i=1}^\gamma \sum_{j=0}^{c_i} \frac{x^j}{j!} \tag{2}$$

where the coefficient of $\frac{x^k}{k!}$, denoted as $\theta(\gamma, k, C)$, is the number of the total number of possible placements.

Specially, if $c_1 = c_2 = \dots = c_\gamma = \bar{c}$, Equation 2 reduces to be

$$GF = (1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots + \frac{x^{\bar{c}}}{\bar{c}!})^\gamma \tag{3}$$

Finally, the total number possible VM-placements is

$$N(m, \gamma, n, k, C) = S(m, \gamma)T(n, k)\theta(\gamma, k, C) \tag{4}$$

Given the m, γ, n and C , we can use Equation 4 to find k_{opt} , which is the optimal number of moving VMs corresponding to the largest N . To make it tangible, an example curve illustrating the $k - N$ relationship is shown in Figure 2. Since $S(m, \gamma)$ does not contribute too much in the magnitude of N , we simply set $\gamma = m$ (so $S(m, \gamma) = 1$); for demonstration, we set the total VMs as $n = 25$ and the capacity of each destination node as $\bar{c} = 4$. From the curve we can learn that:

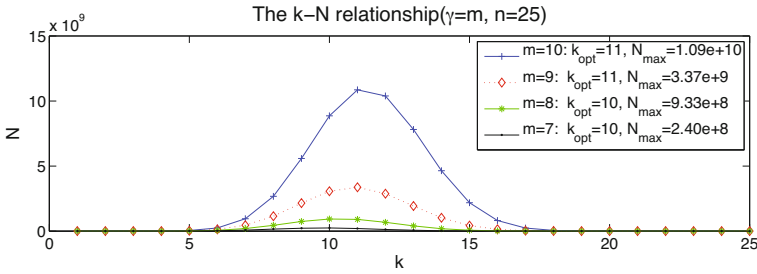


Fig. 2. The k-N relationship when $n=25$, $\bar{c} = 4$, and $m=\gamma$ varies from 10 to 7

1. It is not true that a larger k is better, which means that the intuitive strategy of migrating VMs as many as possible is incorrect.
2. A larger γ , namely more destination nodes, significantly increases the number of possible placement schemes.
3. Compared to γ , N is even more sensitive to k . N can reach the order of 10^9 when k is still in the order of 10^1 , which implies the huge potential of VM migration as a moving target defense strategy.

On the one hand, the optimal k is given by

$$k_{opt} = \operatorname{argmax}_k N(m, \gamma, n, k, C) \tag{5}$$

and in reality, only cloud provider controls the complete information of m, γ, n, k, C , so it is the cloud provider’s responsibility to calculate the k_{opt} .

On the other hand, if we assign 1 to *accept* and 0 to *refuse*, we have the actual number of moving VMs, k_{real} , as:

$$k_{real} = \sum_{i=1}^n x_i \tag{6}$$

In economics, the valuation function should reflect the security preference of the cloud users. Since this is not the main focus of this paper, we simply model the preference as a linear function of the system diversity:

$$v_i(X)|_{m,\gamma,n,C} = \lambda_i N(m, \gamma, n, \sum_{i=1}^n x_i, C) + \eta_i \tag{7}$$

where λ_i and η_i are determined by the i th cloud user’s preference.

Once the cloud provider decides the security level, and thus the value of k , the rest is to motivate the randomly chosen k VMs, denoted as Ω , to get migrated. To satisfy the Clarke pivot rule defined in Definition 4, we design the payment function for each $P_i \in \Omega$ as:

$$p_i(X) = \sum_{j \neq i} v_j(X_{-i}) - \sum_{j \neq i} v_j(X) + \bar{p}_i \tag{8}$$

where \bar{p}_i is the constant cost due to service interruption as assumed. We will further introduce the determination of \bar{p}_i in Section 3.2

The second part of Equation 8, $-\sum_{j \neq i} v_j(X)$, is actually a positive reward from cloud provider to the i th VM. It can be any form with monetary value equalling to it, like extra CPU scheduling credit, bonus VM life, extra network bandwidth and so on, depending on the need of P_i . The first part, $\sum_{j \neq i} v_j(X_{-i})$, is the monetary charge of P_i . If P_i chooses *accept*, \bar{p}_i is the actual migration cost; if P_i chooses *refuse*, \bar{p}_i is in the form of monetary charge. Given that everyone else accepts migration, if P_i decides to migrate, we have $k_{real} = k_{opt}$; if P_i refuses to do so, $k_{real} = k_{opt} - 1$.

Theorem 1. (The criterion for the migration decision) *The mechanism with the valuation function shown in Equation 7 and the payment function shown in Equation 8 is individually rational if and only if*

$$\sum_{i=1}^n \lambda_i(N(m, \gamma, n, k_{opt}, C) - N(m, \gamma, n, k_{opt} - 1, C)) > \bar{p}_i$$

Proof. To prove that the mechanism is individually rational, we should show that the mechanism has a utility of at least zero for each of the players. For those selected to be migrated:

$$\begin{aligned} u_i(X) &= v_i(X) - p_i(X) = \sum_{i=1}^n v_i(X) - \sum_{j \neq i} v_j(X_{-i}) - \bar{p}_i \\ &= \sum_{i=1}^n \lambda_i(N(m, \gamma, n, k_{opt}, C) - N(m, \gamma, n, k_{opt} - 1, C)) - \bar{p}_i \end{aligned} \tag{9}$$

Only when the value of the above equation is larger than 0, P_i can achieve individual rationality. As a matter of fact, Theorem 1 provides a criterion for cloud provider to decide whether a migration is cost-efficient.

Theorem 2. (Incentive Compatibility) *The mechanism with the above valuation and payment functions is incentive compatible and thus motivates the players to truthfully reveal their security preference.*

Proof. To get to the proof of incentive compatibility, we must show that for any given i , X_{-i} and the P_i 's choice $x_i = \textit{accept}$ or $x_i = \textit{refuse}$:

$$u_i(X = \textit{accept} \cup X_{-i}) \geq u_i(X' = \textit{refuse} \cup X_{-i}) \tag{10}$$

The utility of P_i for X is given by

$$u_i(X) = v_i(X) - p_i(X) = \sum_{i=1}^n v_i(X) - \sum_{j \neq i} v_j(X_{-i}) - \bar{p}_i \tag{11}$$

Likewise,

$$u_i(X') = \sum_{i=1}^n v_i(X') - \sum_{j \neq i} v_j(X_{-i}) - \bar{p}_i \tag{12}$$

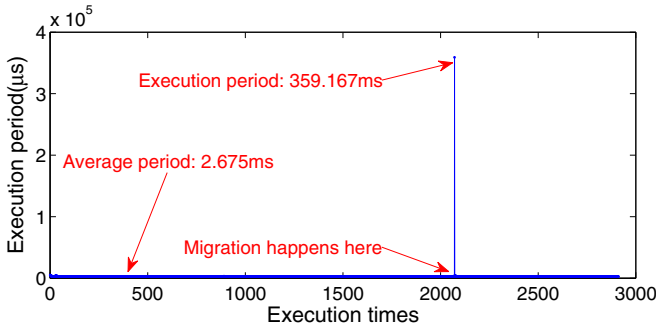
So we have

$$\begin{aligned}
 u_i(X) - u_i(X') &= \sum_{i=1}^n v_i(X) - \sum_{i=1}^n v_i(X') \\
 &= \sum_{i=1}^n \lambda_i(N(m, \gamma, n, k_{opt}, C) - N(m, \gamma, n, k_{opt} - 1, C)) \quad (13)
 \end{aligned}$$

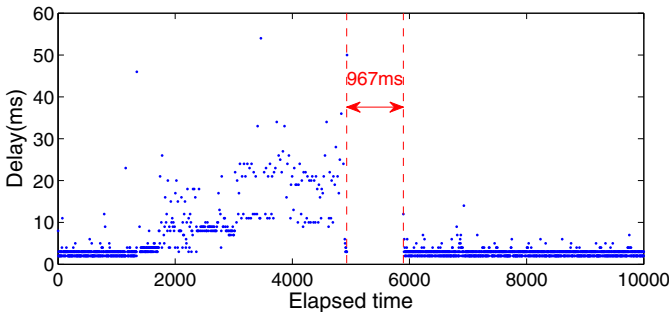
According to Equation 5, the value of Equation 13 is always non-negative. Therefore, the mechanism is incentive compatible.

3.2 The Constant Cost of Migration

In Equation 8 we introduced the \bar{p}_i , which is the constant cost of service interruption each time P_i is migrated. According to [8], \bar{p}_i is determined by the initial memory size of a VM and the applications' memory access pattern. For demonstration, we illustrate the downtime due to migration in Figure 3. The platform specifications are listed in Table 1.



(a) Impact on CPU execution time of a certain task due to migration. As illustrated in the graph, the execution downtime is around 360ms.



(b) Migration impact on response delay of web server. As illustrated in the graph, the web service downtime due to migration is 967ms.

Fig. 3. The influence of migration to VM computation and web service

and the VM that we migrate is of 1 vcpu, 1 GB memory and 4 GB disk. To test the influence to the cpu execution, we keep the VM executing “for (i=0; i<999999; i++); gettimeofday(&time, NULL);” and recording the execution period. As shown in [3a](#), the migration will bring in an execution delay of around 350ms. To test the influence to the network service, we keep measuring the VM’s response delay of GET requests. As shown in [3b](#), there will be a 967ms downtime to the web server. P_i and each P_i can estimate the \bar{p}_i respectively, according to the service downtime.

Table 1. Platform specifications of migration downtime experiment

Hardware (source and destination nodes):	VMM:
CPU: Intel Xeon x5650 2.66GHz \times 2	Xen version: 4.0.1-21326-02-0.5
RAM: 8GB DDR3 1333MHz \times 3	Dom0: openSUSE 11.3 x86_64
Ethernet: Broadcom NetXtreme II BCM5709	Dom0 kernel: 2.6.34.7-0.7-xen

3.3 Defense Timeline

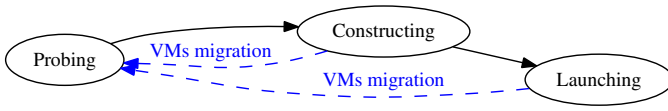
Although the migration can confuse the attackers and force them to restart the attack procedures, with time going by the attackers can still figure out the new VM placement as well. To solve the problem, we have to keep migrating the VMs, and how to determine the optimal time interval becomes a new problem. Following the model in [5i](#), the State Transition Diagram (STG) of attacks is shown in [Figure 4a](#). More specifically, the attackers have to firstly probe the placement of the VMs, for example, by continually creating and stopping their probing VMs and testing the colocation with target VMs; then, it takes some time to construct the attacks; and there is still a period before the attackers successfully gathering all the information. Any distortion of the VM placement, for example, by migrating any of the VMs to other places, will reset the attack to the initial state.

As shown in [Figure 4b](#), suppose the total time needed for a successful attack is t_1 , the time interval between any two migrations t_2 should satisfy $t_2 < t_1$, which offers the cloud provider the criterion to determine migration intervals. If $t_2 \geq t_1$, there is a highly chance for the attackers to fully extract the secret. In reality the constructing time and the launching time of the attacks could be treated as constants, while the probing time has a linear relationship with the N , which explodes with the increasing of γ as shown in [Figure 5d](#).

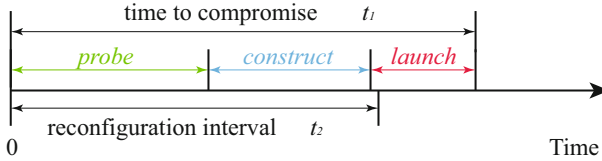
4 Evaluation

4.1 Security

We can never guarantee that our defense strategy or algorithm is agnostic to the adversaries. So when we design a moving target defense scheme, we must ensure that the adversaries are impossible or difficult to recover our actual internal infrastructures even if the strategy is not secret. In the mechanism designed here, we assume that the forms



(a) The State Transition Diagram (STG) of attacks.



(b) Attack and defense lifetime.

Fig. 4. The timeline of attacks and moving target defense (VM migration)

of the valuation function and the deliberate payment function are known to the public. However, in reality the individual security preference is kept as a privacy of each cloud user, thus the adversaries have no way to get λ_i and η_i . Even if some of these parameter-pairs are exposed, as long as the adversaries cannot obtain the information of all the VMs in the cloud, our strategy based on the overall social welfare is always secure. Furthermore, during the generation of the migration scheme, γ are randomly chosen and the capacity $C = \{c_1, c_2, \dots, c_n\}$ is a secret of cloud provider only, thus according to Equation 5 the final migration scheme is only known to cloud provider.

Another indicator of one moving target defense scheme’s capability is the number of the total target space. As demonstrated later in Figure 5d, N can increase rapidly with a small increase of γ . With the exploding number of possible VM placements, the adversaries are extremely difficult to enumerate all the possibilities.

We note that there might be information leakage during VM migration. But since the design of migration methods is not within the scope of this paper, we prospect more secured VM migration mechanisms.

4.2 Practicability

As shown in Figure 5 compared with \bar{c} , γ contributes more on the value of both k_{opt} and N_{max} . Empirically, $k_{opt} \approx \gamma$; theoretically, for any $c_1, c_2, x \in \mathbb{N}$ and $c_1 < c_2$ we have $\frac{x^{c_1}}{c_1!} > \frac{x^{c_2}}{c_2!}$ and $\lim_{c \rightarrow \infty} \frac{x^c}{c!} = 0$, so according to Equation 3 γ has a significantly larger influence on the value of k_{opt} . Consequently, for the real-world cloud providers, it is pretty easy to determine the optimal strategy by assigning $k_{opt} = \gamma$ without complex algorithms, which makes this approach scalable.

Another potential concern is that whether the cloud users are willing to split a secret into multiple VMs, which is the precondition of our VM migration based defense. As shown in Table 2 the small instance has a price of 1/4 of the price of the large one, and offers 1/4 of the compute units, 1/4.4 of the memory and 1/5.3 of the storage.

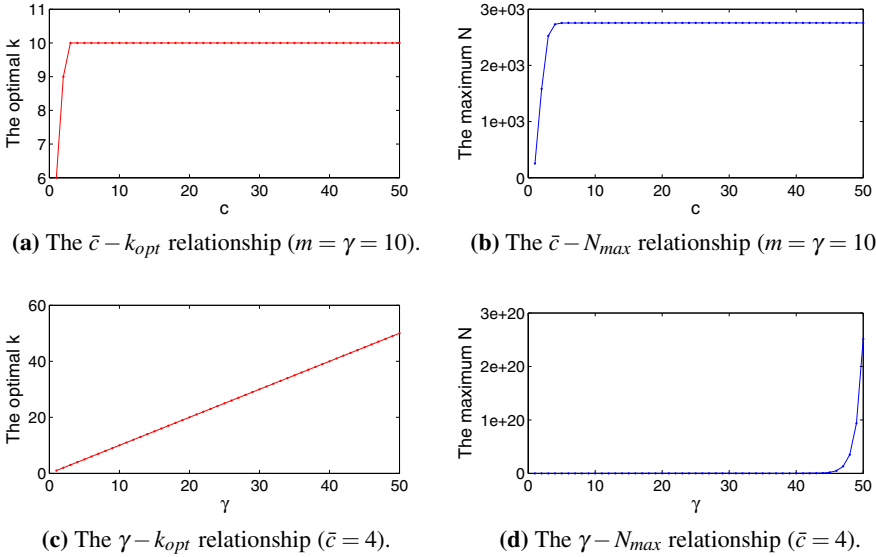


Fig. 5. The influence to k_{opt} and corresponding N_{max} by adjusting \bar{c} or γ in Equation 5

Table 2. Amazon EC2 Linux/UNIX instances pricing and specifications (US East, up to January 2012) [11]

Instance types	Price	Memory	Compute Units	Storage	API name
Small	\$0.085 per hour	1.7 GB	1	160 GB	m1.small
Large	\$0.34 per hour	7.5 GB	4	850 GB	m1.large

Therefore splitting a secret from one large VM into four small VMs won't largely increase the cost. Compare with the risk of being attacked through covert channels, the cloud users will be willing to endure the cost due to transferring computation from few large VMs to numerous small VMs.

5 Discussion and Conclusion

Our work can be extended in different avenues. First, only security goal is considered here for simplicity, but the valuation and payment functions can be easily adjusted to include the performance considerations. Second, so far we have only considered the non-cooperative game. Actually, there might be some connections between VMs on cloud, forming a cooperative game which is a competition between coalitions of players rather than between individual players. In the future, we will develop the mechanism in regard to the coordination game. Furthermore, in the next step, we will extend the mechanism to suite into the asymmetric information game, because in reality some VMs

are offering services to other VMs (thus knowing some statistics of the latter), and some VMs may even be controlled by attackers with some knowledge of other VMs.

To summarize, we proposed an incentive compatible moving target defense of cloud VM-colocation attacks, based on the VCG mechanism. When the migration is acceptable by rational users and how to determine the time interval are discussed. Our analysis shows that this defense strategy is practical to be applied to commercial clouds.

Acknowledgments. This work was supported in part by NSF Grants CNS-1100221 and CNS-0905153.

References

1. Amazon web services, <http://aws.amazon.com>
2. Workshop on Secret Sharing and Cloud Computing. MEXT, Institute of Mathematics for Industry, Kyushu University, Japan (2011)
3. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I., Zaharia, M.: Above the clouds: A Berkeley view of cloud computing. Tech. Rep. UCB/EECS-2009-28, EECS Department, University of California, Berkeley (February 2009)
4. Danev, B., Masti, R.J., Karame, G.O., Capkun, S.: Enabling secure vm-vtpm migration in private clouds. In: Proceedings of the 27th Annual Computer Security Applications Conference, ACSAC 2011, pp. 187–196. ACM, New York (2011)
5. Evans, D., Nguyen-Tuong, A., Knight, J.: Effectiveness of moving target defenses. In: Jajodia, S., Ghosh, A.K., Swarup, V., Wang, C., Wang, X.S. (eds.) Moving Target Defense. Advances in Information Security, vol. 54, pp. 29–48. Springer, Heidelberg (2011)
6. Kant, K.: Configuration management security in data center environments. In: Jajodia, S., Ghosh, A.K., Swarup, V., Wang, C., Wang, X.S. (eds.) Moving Target Defense. Advances in Information Security, vol. 54, pp. 161–181. Springer, Heidelberg (2011)
7. Keller, E., Szefer, J., Rexford, J., Lee, R.B.: Nohype: virtualized cloud infrastructure without the virtualization. SIGARCH Comput. Archit. News 38, 350–361 (2010)
8. Liu, H., Xu, C.-Z., Jin, H., Gong, J., Liao, X.: Performance and energy modeling for live migration of virtual machines. In: Proceedings of the 20th International Symposium on High Performance Distributed Computing, HPDC 2011, pp. 171–182. ACM, New York (2011)
9. Nisan, N., Roughgarden, T., Tardos, E., Vazirani, V.: Introduction to Mechanism Design (for Computer Scientists). Cambridge University Press (2007)
10. Page, D.: Defending against cache-based side-channel attacks. Information Security Technical Report 8(1), 30–44 (2003)
11. Ristenpart, T., Tromer, E., Shacham, H., Savage, S.: Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In: Proceedings of the 16th ACM Conference on Computer and Communications Security, CCS 2009, pp. 199–212. ACM, New York (2009)
12. Shamir, A.: How to share a secret. Commun. ACM 22, 612–613 (1979)
13. Wang, Z., Lee, R.B.: New cache designs for thwarting software cache-based side channel attacks. SIGARCH Comput. Archit. News 35, 494–505 (2007)

Give Rookies A Chance: A Trust-Based Institutional Online Supplier Recommendation Framework

Han Jiao¹, Jixue Liu¹, Jiuyong Li¹, and Chengfei Liu²

¹ Data and Web Engineering Lab,
School of Computer and Information Science,
University of South Australia,
SA 5095, Australia

{han.jiao,jixue.liu,jiuyong.li}@unisa.edu.au

² Faculty of Information and Communication Technologies,
Swinburne University of Technology,
Melbourne, VIC3122, Australia
cliu@swin.edu.au

Abstract. Trust and reputation systems are widely adopted to help online consumers choose trustworthy suppliers. One problem is that consumers are only attracted by high-reputation holders, which disadvantages the newcomers entering the market, even they provide better goods or services. In this paper, we propose an online supplier selection model in which online consumers interact with a web institution, where online suppliers are registered in, and the web institution will recommend trustworthy suppliers according to a trust-based algorithm. This new recommendation model finds a balance point between minimizing the defective interactions as well as granting opportunities to newcomers. In addition, it saves consumers' efforts to look for trustworthy suppliers. We propose a framework to help realize the new interaction mode. We also use experiments to show its effectiveness.

Keywords: trust, trustworthiness, reputation, customer feedback, agent selection, game theory.

1 Introduction

Nowadays, using reputation-based methods become the main stream to help determine trustworthiness in the online environment [1]. Reputation is derived by scoring and integrating certain attributes of a reputation holder to reflect the global view of the reputation holder. Online consumers give their trust to a supplier by referencing its reputation in relation to the reputations of others. Many reputation-based models using various methods to calculate and compare reputations were studied [2].

Reputation models were improved to better represent the trustworthiness of a supplier. Meanwhile, online suppliers try their best to become high reputation

holders. All these activities make the whole market towards more trustworthy. However, one serious problem is that online consumers are only attracted by high reputation holders. Newcomers without any historical transactions are hard to be admitted by the market, even though they provide better goods and services. In other words, trust and reputation systems, while providing references to help determine trustworthiness, also place an entry barrier that blocks rookies to enter. This problem is also interpreted to be setting initial reputations for newcomers, since the initial reputation gives online consumers hints to judge trustworthiness. In [10,14], it is termed reputation bootstrapping.

Some studies have been conducted on this issues, releasing preliminary results [10,11,12]. In this paper, we solve this problem by designing an institutional recommendation model, between online consumers and suppliers, to grant newcomers some chances without introducing extra risks of potential fraudulence. We first introduce a new concept “reputation utility”, based on which we use game theory to construct a model that bridges reputation and trust formation. Then, we propose a framework for a web institution, where online suppliers are registered in, to automatically recommend interactive partners for online consumers. The core selection algorithm is trust-based. We make an implementation of the proposed framework and use experiments to evaluate it.

2 Related Works

2.1 Trust, Trustworthiness and Reputation

The ubiquity of trust issues offers nothing but more diversified opinions that distract the mainstream understanding of this old concept, since human consciousness came into being. The evolution of trust definition gradually clears the masks on the mystery to result in a notion admitted by the most. Gambetta [4] defined trust as a subject possibility that individual A expects B to perform a given action which A’s welfare depends, which is named “reliability trust” in Jøsang et al’s survey paper [5]. Appreciation must be given to Mayer et al. for rewarding their seminal work on an integrative trust model in 1995 [3], in which several basic notions related to trust were explicitly distinguished. In that work, trust is defined to be “the willingness of a party to be vulnerable to the actions of another party based on the expectation that the other will perform a particular action important to the trustor, irrespective of the ability to monitor or control that other party”. They stressed on vulnerability and regarding trust as willingness to take risk. Definition given by McKnight and Chervany in [6] is of generality, saying that “trust is the extent to which one party is willing to depend on something or somebody in a given situation with a feeling of relative security, even though negative consequence are possible”. This is called “Decision trust” in the work of Jøsang et al [5]. Our paper adopts the definition in [6].

Many research works do not explicitly distinguish the concepts of trust and trustworthiness, though they are two related, yet different concepts [3] [7]. Trust is a kind of mental activity in a trustor, while trustworthiness is an attribute

of a trustee that justifies why a certain amount of trust is worthy to be given to that trustee. Researchers use several dimensions to represent trustworthiness. The most famous classification is proposed in [3] and later adopted in [8], where ability, benevolence and integrity are the three recognized. As the measurement of the extend be to trusted, trustworthiness is like a label stuck on trustees that shines signals to attract attentions. However, a trustor has the "final vote" to decide who would be trusted.

The concept of reputation originates from a contradictory situation where a complete knowledge set on a specific object is unattainable since a full investigation and evaluation is beyond the resource of an individual. It is very natural to consider solving the contradictory situation by relying on aggregated power from multiple individuals. Reputation is the mutual knowledgeable perception that integrates various knowledge sources to reflect a global view of the target. In [3], reputation is defined as "what is generally said or believed about a person's or thing's character or standing", which also conforms to the definitions given in [2] and [8]. Basically, reputation is an objective and collective measurement to evaluate a supplier. It is given by the public and is common knowledge held by the whole community. In a situation that one individual has no private knowledge on the other, reputation can be considered as a collective measure representing trustworthiness. This awareness is particularly important in the Web environment, where two online suppliers without any prior interactive experiences is an common case. In this paper, instead of opening the box to check each dimension (ability, benevolence and integrity), we regard reputation as the only indicator of trustworthiness. Further, we use consumer feedback as the base to calculate reputation. This will be discussed in a later section.

2.2 Trust for Newcomers

The problem solved in this paper is to promote newcomers in the reputation system without introducing high risks of defective transactions. Currently, there are only a few research focusing on this issue, releasing the outcomes with limitations.

In [11], authors propose that participants with unknown reputation may acquire reputation through the endorsement of other trusted participants and the endorsee's actions directly affect the endorser's credibility. However, it may not be possible for a newcomer to find enough suppliers who are willing to give endorsements, as a newcomer is usually a new competitor to existing suppliers. In [10], Zaki and Athman interpret the trust for newcomers with another term, "reputation bootstrapping", which means to set newcomers a proper initial values. They use both adaptive and assigned approaches to achieve the goal. In the adaptive approach, a reputation community asked their participant to disclose its rate of maliciousness and use an aggregation method (for example, average weighted sum) to calculate an initial reputation for a newcomer. This reputation is adaptive to the latest status of the whole environment since it refers to other existing reputation holders. However, there is no solid evidence showing the average malicious rate is related to any newcomer's initial reputation. In addition, if no enough rate of mali-

sciousness can be collected, the system can use assigned approach to process, which either asks the newcomer to buy reputation with authenticated credentials or evaluate the newcomers for a while (fake trading to see its performance) and then decide its reputation. Realization of these approaches is difficult, as no one wants to expose their malicious rate; the value of an authenticated credential is hard to measure; and a reputation holder can perform differently during and after an evaluation period. In [12], authors similarity of profiles as the base to determine a newcomer's reputation. They designed models for trust mirroring and trust teleportation to address the issue. But profiles are usually contains privacies and are not easy to obtain and compare in real cases. It may be possible that a newcomer even has no profile since it is very new to the business world. Newcomers can also provide fake profiles to make them look better.

In general, determining newcomer's trustworthiness is still an open issue. In our opinion, this is not a essential problem. The final purpose is that how to get the newcomers involved in without introducing extra risks for defective transactions, which is the problem we will solve.

3 Reputation Utility and Online Trust Game

An interaction occurs after a consumer trusts a supplier. Assuming that a consumer has no private knowledge about a supplier, its trust is derived by referencing the supplier's reputation. The purpose that we introduce the concepts of reputation utility and online trust game is to clarify the relationship between reputation and trust formation.

3.1 Reputation Utility

Reputation utility (RU) is the benefit that reputation brings to its holder. To better understand it, let us use E-commerce as an example. In a certain market, customers who once bought goods from an E-commerce seller may leave feedbacks. These accumulated feedbacks can be viewed as a kind of reputation by other buyers. It is natural that a seller with nice customer comments has more chances to attract more prospective buyers. In other words, better reputation gains more market shares. This is an evidence that reputation has utilities.

Definition 1. *Reputation Utility (RU) is the benefit that a certain reputation value brings to its holder. Given a supplier whose reputation is r , we use $\Phi(r)$ to represent the reputation utility of r , and name this function as Reputation Utility Function (RUF).*

In a reputation-based environment, reputation is changed after certain types of events, such as transactions, customer feedbacks or time durations. The events that can cause a reputation change is termed to be reputation alteration event (RAE). Based on the definition of RU and RAE, we define the concept of marginal reputation utility (MRU).

Definition 2. *Marginal reputation utility (MRU) is the amount of reputation utility change per occurrence of a RAE. It has two components, $\phi_+(r)$ and $\phi_-(r)$, representing the effects of decrease and increase of RAE on MRU. The two functions are called Positive MRU Function and Negative MRU Function.*

Positive marginal reputation utility (PMRU) is the reputation utility change per unit for reputation increase. The negative marginal reputation utility (NMRU) is for reputation decrease. MRU is the change amount of reputation utility per combined RAE, which should be distinguished from the change of reputation itself.

Definition 3. *Given a certain reputation system S , its reputation change policy can be represented by two functions $\eta_+(r, e_+)$ and $\eta_-(r, e_+)$. Given a supplier whose reputation is r , $\eta_+(r, e_+)$ stands for the reputation change amount for reputation increase given the Event e_+ occurs, and $\eta_+(r, e_-)$ is the reputation change amount for reputation decrease when Event e_- occurs. We name two functions as Positive Reputation Change Function and Negative Reputation Change Function, respectively.*

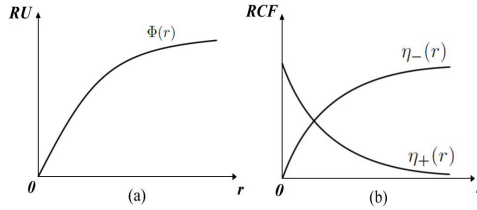


Fig. 1. RU Curve and MRU Curves

The above three concepts are co-related with each other. Their relationship can be represented by the following two equations:

$$\phi_+(r) = \Phi(r + \eta_+) - \Phi(r) \tag{1}$$

$$\phi_-(r) = \Phi(r) - \Phi(r - \eta_-) \tag{2}$$

As reputation goes up, the utility of each reputation unit is diminishing. Consider that there is a supplier who has already constructed a good reputation, then the incremental of his reputation score cannot bring him significant utility anymore. This corresponds to the fact that as the reputation goes higher, each reputation unit increase brings less benefit to its holder, which that as two reputation values go higher, the difference of the two reputation holders is not as remarkable as when the reputations are low. This phenomenon in Economics is termed the Law of Diminishing Marginal Utility [13], which makes the reputation utility function as shown in Fig. 1(a). In addition, reputation is hard to gain but easy to lose. Therefore, η_+ should be a decreasing function and η_- is an increasing one, as shown in Fig. 1(b).

3.2 Online Trust Game

Online transactions, being full of remoteness and anonymity, leave participants good possibility to cheat. In a reputation-based circumstance, cheating behaviors usually mean getting an immediate advantage at the expense of lost prospective benefit. The lost prospective benefit is reflected in the change of reputation utility. In our opinion, the interaction between a consumer and a supplier can be modeled by a game, in which each side selects a strategy that maximize their payoffs. This game model describes the gains and losses of each side when selecting different strategies. It creates the linkages between the reputation utility and trust formation. Based on the game theory, we now use a two-player two-stage sequential game to describe an online interaction shown in Fig. 2.

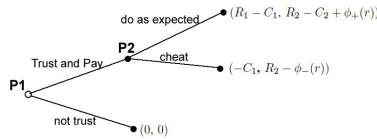


Fig. 2. Online Trust Game

The game has two players: P1 is the consumer and P2 is the supplier (reputation holder). At the beginning stage, P1 decide whether to trust P2. If not, the game finishes and yields to zero payoffs for both players. Otherwise, the game comes to P2’s turn to decide whether to act as expected or to cheat. By acting as expected, P1’s requirements are fulfilled. So P1 get benefit R_1 and pay cost C_1 . As rewards, P2 gets the benefit R_2 plus an increment of reputation utility $\phi_+(r)$. But P2 must pay C_2 as cost. If P2 cheats, P1 loses its payment (C_1) and P2 gets the benefits R_2 without any cost. As punishment, an extra immediate payoff ($\phi_-(r)$) due to the cheating behaviors. However, P2’s reputation will decrease which results in a decrease of reputation utility.

We assume that the two players both have complete and perfect information. We use backward induction to find the conditions that P1 trusts P2. P1 knows that if the game goes to the second stage, P2 will choose to act in the way that brings more benefit to P2. Only when $R_2 - C_2 + \phi_+(r) > R_2 - \phi_-(r)$, P2 chooses to act as expected, which means that only in that case P1 can trust P2. Therefore, the condition that P1 trusts P2 is as follows:

$$C_2 < \phi_+(r) + \phi_-(r) \tag{3}$$

Because of Formula (1) and (2), the above formula can be refined as follows:

$$C_2 < \Phi(r + \eta_+) - \Phi(r - \eta_-) \tag{4}$$

The above condition tells us that whether P1 trusts P2 depends on numeric relationship between the production cost of the trading goods and the sum of the PMRU and NMRU at the current reputation. In a mature market of a certain products, the cost of each product can be considered as a fixed value (assume that the production technologies are mature and innovation can not be made in a short time). Therefore, the condition for trusting an web supplier is that the sum of PMRU and NMRU of the provider must be greater than a constant value. This intermediate conclusion solve the problem of filtering trustworthy web suppliers.

4 Trust-Based Auto-selection Framework and Algorithms

Assume that there are n suppliers in a reputation system, whose reputations are r_1, r_2, \dots, r_n and who are providing the similar goods or services. We first normalize their reputations. Then we use reputation utility functions to transform reputation values into benefits that reputations bring. After that, we use the trust game model introduced before to filter out trustworthy suppliers. Within all the trustworthy suppliers, we use an auto-selection algorithm to decide a supplier to interact.

4.1 Normalization

In a specific reputation system, reputation scores can be scaled into a fix range $[0, 1]$. A threshold r_o value must be decided to filter out the suppliers who are not considered by consumers. The threshold stands for the psychological baseline of consumers. The suppliers whose reputations are lower than the threshold will never be considered. Given a group of existing reputation scores r_1, r_2, \dots, r_n , the highest of which is r_{max} , for a supplier r_i , its normalized reputation R_i is calculated by the formula that follows:

$$R_i = \begin{cases} \frac{r_i - r_o}{r_{max} - r_o} & \text{if } r_i > r_o \\ 0 & \text{if } r_i \leq r_o \end{cases} \quad (5)$$

After normalization, primitive reputations r_1, r_2, \dots, r_n are transformed into R_1, R_2, \dots, R_n . If there are m candidate suppliers whose primitive reputation is beyond r_o , The normalized reputations can be represented as $(0, 0, \dots, R_1, R_2, \dots, R_m)$, where the number of "0" is $m - n$. Truncating the 0 values from the list, we get the final normalized reputations $R = (R_1, R_2, \dots, R_m)$.

4.2 Utilization

Sometimes, even we normalized all the reputations, it may still not be able distinguish reputation holders (as the normalized reputations are still close to

each other). To solve this problem, we define a Utility Amplification Function (UAF) $\varepsilon(R)$ that is monotone increasing on its definition domain $[0, 1]$ and whose second derivative must be greater than 0. For example, $\varepsilon(R) = R^n$ where $n > 1$ would satisfy the condition. UAF is optional to use. It helps only when the normalized reputations fail to show clear differences.

Let us now assume that the volume of market demand in a certain duration is V ; the average unit cost of interaction for provider is c ; the average unit price for consumer is p . Our framework defines the Reputation Utility Function (RUF) as follows:

$$\Phi(r_j) = \frac{\varepsilon(R_j)}{\sum_{i=1, i \neq j}^n \varepsilon(R_i) + \varepsilon(R_j)} \times V \times (p - c) \times \rho \tag{6}$$

Notice that function $f(x) = \frac{x}{S+x}$, where S is a constant value, has the similar monotonicity and concavity as the function shown in Fig. 1 (a). Therefore we use it as the base to form our RUF. For reputation holder r_j , $\sum_{i=1, i \neq j}^n \varepsilon(R_i)$ is a constant value. In fact, $\frac{\varepsilon(R_j)}{\sum_{i=1}^n \varepsilon(R_i)}$ can be viewed as the market share that reputation holder j seizes. Because reputation is not the only factor impacting market share, we use ρ ($0 \leq \rho \leq 1$) to represent the weight that reputation can impact. They are, then, multiplied by V and $(p - c)$ to get the profit that the reputation r_j brings to its holder. By calculating each reputation holder's utility, we finish the second step.

4.3 Trustworthiness Determination

Although in the normalization step we have removed some “unacceptable” reputation holders, the rest are just qualified to be candidates. We have to refine the list based on the conditions specified in Formula (4), which is derived from the online trust game model.

We denote Positive and Negative Reputation Change Functions as $\eta_+(r, e_+)$ and $\eta_-(r, e_-)$. For each reputation holder j , its trustworthiness determination logic is as follows:

1. Suppose that in the next interaction it finishes consumer's request as expected, its primitive reputation will be $r_j + \eta_+(r_j)$. We re-normalize all the reputations to get a new $R' = (R_1, R_2, \dots, R_m)$. Based on Formula (6), we calculate its reputation utility and denote it as Φ_+ .
2. Roll back everything to the current status. Then suppose that in the next interaction it cheats, its primitive reputation will be $r_j - \eta_-(r_j)$. Re-normalize and re-calculate its reputation utility, denoted as Φ_- .
3. If $c < \Phi_+ - \Phi_-$ stands, reputation holder j is trustworthy. Otherwise, mark it as untrustworthy.
4. Roll back everything to current status.

The above logic can be implemented by the following pseudo code.

Pseudo code for filtering trustworthy suppliers

```

01> program TrustworthinessCheck(input)
02> begin
03>   {go through the input array}
04>   int i;
05>   double Phi1, Phi2;
06>   for i=0 to input.length-1
07>     input[i].rep=input.rep + ETA1(input[i].rep);
08>     input=Normalize(input);
09>     Phi1=PHI(input[i].rep);
10>     input[i].rep=input.rep - ETA1(input[i].rep);
11>     input[i].rep=input.rep - ETA2(input[i].rep);
12>     input=Normalize(input);
13>     Phi2=PHI(input[i].rep);
14>     if ((Phi1-Phi2)>=c) then
15>       output.append(input[i].ID, input[i].rep);
16>     end if
17>   next i
18>   return output;
19> end.

```

(The input is an array containing IDs and primitive reputations of all the candidate suppliers. The output is an array with IDs and reputations only for trustworthy suppliers. Function ETA1() and ETA2() is the two reputation change functions. Function PHI() is for calculating the reputation utility. Index of array is from 0.)

4.4 Recommend Supplier

Every trustworthy supplier has chance to to be recommended, although their chances are different based on their reputations. We use the following formula to determine its possibility to be selected in a given consumer request:

$$P_j = \frac{\varepsilon(R_j)}{\sum_{i=1}^n \varepsilon(R_i)} \times \rho + \frac{1 - \rho}{k} \quad (7)$$

Reputation impacts the market share, as well as other factors, so a weight number ρ is deserved. It is assumed that the impact of other factors together follows a even distribution among trustworthy suppliers. Therefore each supplier deserves $\frac{1}{k}$ where k is the total number of trustworthy suppliers from last step. The following pseudo code implements the logic:

Auto-selection Algorithm

```

01> program AutoSelect(input)
02>   int total=0;
03>   int i,j;
04>   for i=0 to input[i].length-1
05>     total=input[i].possibility*10^mu+total;
06>   next i
07>   int ranNum;
08>   ranNum=random(0, total-1);
09>   int sum1=0;
10>   for i=0 to input.length-1
11>     sum2=input[i].possibility*10^mu+sum1;
12>     if ranNum>=sum1 and ranNum<sum2 then
13>       return input[i].ID;
14>     exit;

```

```

15>     else
16>         sum1=sum2;
17>     end if
18> next i
19> end

```

(The input is an array containing IDs and possibility of all the supplier. The output is the ID of the selected supplier. The function $round(a, b)$ returns the value of a with b decimal digits and μ is a parameter of accuracy. The function $random(x, y)$ returns a random integer from x to y . This random number generator follows the even distribution, which means that the chances of the generated number to be any one in the defined range are equal.)

When a supplier quits from the system, all the reputation must be re-normalized so that the market share will be re-allocated. When a new supplier joins, its initial primitive reputation will be equal to that of the lowest trustworthy supplier. This makes it seize a little market share and once it cheats, it falls into the untrustworthy group, which protects the market from defective transactions.

5 Implementation and Experiments

5.1 Environment Settings and Implementation

In an E-commerce market, the total volume is 10000 units. The unit cost is 200 dollars and the unit price is 300 dollars. Positive feedback rate in the last 100 transactions is used to measure reputation. The utilization function is defined as $U = R_{positive} - 4 \times R_{negative}$, in which $R_{positive}$ is the positive feedback rate and $R_{negative}$ is the negative feedback rate. After each transaction, the reputation holder will have $R_{positive}$ and $R_{negative}$ and therefore have different utility. We choose $\varepsilon(R) = R^3$ as the utility amplification function. The possibility of each trustworthy suppliers to be selected is computed as follows:

$$P_j = \frac{U_j^3}{\sum_{i=1}^n U_i^3} \times \rho + \frac{1 - \rho}{k} \quad (8)$$

k is the total trustworthy suppliers that has to be decided by going through the four steps described in the last section. ρ is the weight number and later will be given different values in the experiments.

5.2 Experiment

Assume that 20 suppliers in the market, whose reputations are randomly generated by a normal distribution with mathematical expectation of 90 and variance of 10, as shown in Table 1.

In each round, there will a consumer coming for only one goods item. After each 100 transactions, one supplier will quit and a newcomer will join. The newcomer's real capability (the ability that they actually gain positive feedbacks from consumers) also follows the a normal distribution with mathematical expectation of 90 and variance of 10. The way to decide who quits consists of three

Table 1. Settings for Experiment 1

Supplier	Reputation	Supplier	Reputation	Supplier	Reputation	Supplier	Reputation
Supplier01	0.96	Supplier02	0.93	Supplier03	0.94	Supplier04	0.88
Supplier05	0.88	Supplier06	0.98	Supplier07	0.74	Supplier08	0.98
Supplier09	0.99	Supplier10	0.83	Supplier11	0.78	Supplier12	0.90
Supplier13	1.00	Supplier14	0.98	Supplier15	0.89	Supplier16	0.97
Supplier17	0.99	Supplier18	0.79	Supplier19	1.00	Supplier20	0.89

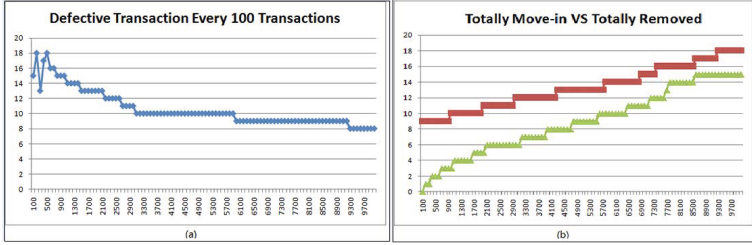


Fig. 3. Online Trust Game

considerations. One is that whether a supplier has been marked as untrustworthy in the system. If so, it will quit. The second consideration is that a supplier with bigger market share should have less possibility to quit because it is doing well. The last consideration is a natural possibility that every supplier may quit for their private reasons. Bearing these in mind, the possibility of supplier j quits is $P_{leave}^j = \rho_1 \times P_{ut} + \rho_2 \times P_{nature} + (1 - \rho_1 - \rho_2)(1 - P_{nt} - P_{nature} * \frac{P_j - P_{min}}{P_{max} - P_{min}})$. P_{ut} is a fixed possibility to quit for untrustworthy suppliers. P_{nature} is a fixed possibility to quit for supplier’s private reasons. P_j is the possibility that a supplier j will be selected to interact in each round. ρ_1 and ρ_2 are the weights. P_{max} and P_{min} are the highest and lowest possibilities of all the suppliers to be selected in a transaction. Fig. 3(a) tells that the number of defective transaction in every 100 transactions goes lower. This shows that the general quality of transactions in the the whole system becomes better. Fig. 3(b) tells the relation between totally moved-in cheaters and total removed cheaters. At the starting point, the system detects several untrustworthy suppliers. As the market moves, they are removed out, but the new untrustworthy cheaters keep on moving in. The two lines converge to each other, which means that the total difference between totally move-in untrustworthy and totally removed untrustworthy suppliers was getting smaller. They finally overlap. Although there are some untrustworthy newcomers moving in, they are removed out quickly as market moves.

6 Conclusion and Future Work

Online consumers are always attracted by high reputation holders, which lifts up the market entrance level for newcomers. In this paper, by introducing the concepts of reputation utilities, we use game theories to construct an online trust game, which links the reputation utility with trust formation. We then proposed

a trust-based auto-selection framework to help consumers select trustworthy online suppliers while balancing the newcomers' involvement. We also make a preliminary implementation of the framework and use an experiment to show its effectiveness.

Our next step is to use real world data to make further examination on the proposed framework. Another possible job is to explore whether there is a general relationship between reputation utility and reputations, which helps to estimate the reputation utility function more precisely.

References

1. Wang, Y., Vassileva, J.: A Review on Trust and Reputation for Web Service Selection. In: Proceedings of the 27th International Conference on Distributed Computing Systems Workshops (ICDCSW 2007). IEEE Computer Society, Washington, DC (2007)
2. Ruohomaa, S., Kutvonen, L., Koutrouli, E.: Reputation Management Survey. *ARES*, 103–111 (2007)
3. Roger, C., Mayer, J.H., Davis, F.: An Integrative Model of Organizational Trust. *Academy of Management Review* 20(3), 709–734 (1995)
4. Gambetta, D.: Can we trust trust? In: Gambetta, D. (ed.) *Trust: Making and Breaking Cooperative Relations*, pp. 213–238. Basil Blackwell, Malden (1990)
5. Jøsang, A., Ismail, R., Boyd, C.: A Survey of Trust and Reputation Systems for Online Service Provision. *Decision Support Systems* 43(2), 618–644 (2007)
6. McKnight, D.H., Chervany, N.L.: The Meanings of Trust, technical Report MISRC Working Paper Series 96-04, University of Minnesota, Management Information Systems Research Center (1996)
7. Gefen, D., Benbasat, I., Pavlou, P.A.: A Research Agenda for Trust in Online Environments. *Journal of Management Information Systems* 24(4), 275–286 (2008)
8. Gefen, D.: Reflections on the Dimensions of Trust and Trustworthiness Among Online Consumers. *ACM SIGMIS Database* 33(3) (2002)
9. Artz, D., Gil, Y.: A Survey of Trust in Computer Science and the Semantic Web. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web* 5(2) (2007)
10. Malik, Z., Bouguettaya, A.: Reputation Bootstrapping for Trust Establishment among Web Services. *IEEE Internet Computing* 13(1), 40–47 (2009) doi:10.1109/MIC.2009.17
11. Maximilien, E.M., Singh, M.P.: Reputation and Endorsement for Web Services. *SIGecom Exchanges* 3(1), 24–31 (2002)
12. Skopik, F., Schall, D., Dustdar, S.: Start Trusting Strangers? Bootstrapping and Prediction of Trust. In: Vossen, G., Long, D.D.E., Yu, J.X. (eds.) *WISE 2009*. LNCS, vol. 5802, pp. 275–289. Springer, Heidelberg (2009)
13. Samuelson, P., Nordhaus, W.: *Economics*, 19th edn. McGraw-Hill/Irwin (2009)
14. Jiao, H., Jixue Liu, J., Li, J., Liu, C.: A Framework for Reputation Bootstrapping Based On Reputation Utility and Game Theory. In: *IEEE TrustCom 2011*, pp. 344–351 (2011)

A Game-Theoretic Formulation of Security Investment Decisions under Ex-ante Regulation*

Giuseppe D'Acquisto¹, Marta Flamini², and Maurizio Naldi³

¹ Garante per la protezione dei dati personali, Roma, Italy
g.dacquisto@garanteprivacy.it

² Università telematica internazionale UNINETTUNO, Roma, Italy
m.flamini@uninettunouniversity.net

³ Università di Roma Tor Vergata, Roma, Italy
naldi@disp.uniroma2.it

Abstract. Data breaches represents a major source of worries (and economic losses) for customers and service providers. We introduce a data breach model that recognizes that breaches can take place on the customer's premises as well as on the service provider's side, but the customer bears the economic loss. In order to induce the service provider into investing in security, a regulatory policy that apportions the money loss between the customer and the service provider is introduced. A game-theoretic formulation is given for the strategic interaction to the customer and the service provider, where the former sets the amount of personal information it releases and the latter decides how much to invest in security. The game's outcome shows that shifting the burden of the money loss due to data breaches towards the service provider spurs its investment in security (though up to moderate levels) and leads the customer to be more confident, but the apportionment must not be too unbalanced for a Nash equilibrium to exist. On the other hand, changes in the probability of data breach of both sides do not affect significantly the service provider's behaviour, but cause heavy consequences on the customer's confidence.

Keywords: Privacy, Data breach, Game theory, Security economics, Security investments.

1 Introduction

Customers of networks and information systems are continually asked to provide their personal data, often in return for enhanced services or discounts. However, those data may fall prey to malicious users. Data breaches occur everyday on any link of the information chain: on the customer's premises, over the network, on the legitimate information recipient. Security is therefore an outstanding

* The support of the Euro-NF Network of Excellence is gratefully acknowledged by the third author. The paper reflects the personal opinion of the authors and cannot be regarded as an official position of the Garante on the subject.

concern in today's networks and information systems. The personal data may be used by malicious third parties for frauds, causing significant losses of money to customers.

Service providers can reduce data breaches by investing in security. The relationship between incremental investments and data breaches has been explored in [1], where the possibility of identifying an optimal level of investment has been determined. The relevance of choosing an optimal level of investment has been shown in [2] also, where it has been recommended that future research should explore what will happen with changes in consumer demand.

But service providers may have no incentives to invest in security. In a peer-to-peer context, where the presence of a service provider is not considered, it has been shown that the presence of negative externalities requires a regulatory intervention to avoid a large social cost [3]. In more general terms, ICT security can be regarded as a public good, and its provision has to be safeguarded through regulatory intervention at some superseding level of governance [4].

In this paper, we propose an ex-ante regulatory intervention, which apportions the expected money loss resulting from a data breach between the customer and the service provider. Such damage sharing policy may represent an incentive for the service provider to invest in security, so as to limit the charge resulting from the damage sharing policy. We formulate a game-theoretic model, where the interaction between the service provider and the customer includes the damage sharing policy, with the service provider acting on the investment in security, and the customer acting on the amount of personal information released. We find that a single Nash equilibrium is reached for a wide range of cases, and that the quota of loss apportioned to the service provider acts as an incentive both for the service provider to invest and for the customer to release its data. But, if the service provider is charged too high a quota, no Nash equilibrium is reached. Instead, the service provider is largely unaffected by variations in the maximum probability of data breach.

The paper is organized as follows. We describe the behaviour of the service provider and the customer in Section 2, and the damage sharing policy in Section 3. The resulting surplus functions for both stakeholders are derived in Section 4, and are employed in Section 5 to formulate a game between them. The results are analysed in Section 6.

2 Stakeholders and Information Release

The release of personal information by the customer brings both benefits and disadvantages. The service provider rewards the customers by offering discounts or an enhanced service. On the other hand, releasing personal data exposes the customer to data breach risk (and the ensuing money loss). In this section, we provide models for the positive and negative effects of the release of personal data.

We start by considering a simple model for the interaction between the customer and the service provider.

We recall that the service provider sells services at a unit price p ; the customer buys a quantity q of such services, represented, e.g., by minutes of phone traffic, bytes of data traffic volume, digital units, CPU time, bytes of storage capacity. The relationship between p and q is the *demand curve* [5]. For sake of simplicity, we assume here that in our case the relationship is linear. When no personal information is disclosed, those quantities are related by the expression

$$\frac{q}{q^*} + \frac{p}{p^*} = 1 \quad q < q^*, p < p^*, \quad (1)$$

where q^* is the maximum quantity of service that the service provider can provide, and p^* is the maximum unit price that the customer can sustain (its *willingness-to-pay*). When the service is free ($p = 0$), the customer asks for the maximum quantity that the service provider can supply ($q = q^*$). When the price is larger than the willingness-to-pay ($p \geq p^*$), the customer does not buy the service, and the quantity of service sold is $q = 0$. In the following, we treat both the quantity of service q and the unit price p as continuous variables (though their variation is actually discrete), since we assume that their granularity is extremely small with respect to the values at hand.

If the customer is willing to release some personal information, the service provider eases the provision of services, e.g., by providing personalized services or automatic login. In fact, the more the service provider knows about the customer, the better it can shape and direct its offer to achieve a sale. The release of personal data can help reduce the product/service search costs for both parties: the time employed by customers when looking for that product/service, and the effort spent by sellers trying to reach out to their customers. Varian has shown that customers rationally want some of their personal information to be available to sellers [6]. Hence, the customer is incentivized to supply its personal data and increase its consumption. Though the information is actually released in discrete increments (e.g., first the family name, then the birthday, and so on), we assume, for mathematical convenience, that the information is a continuous quantity.

Each release of information by the customer is rewarded by a new offer by the service provider, which at the same time incentivizes the consumption. The demand curve correspondingly changes as in Figure 1, where we can observe how the working point moves onto the new demand curve. For example, in Figure 1, the point (q_1, p_1) on the pre-release demand curve, represented by Eq. (1), moves to the working point (q_2, p_2) on the after-release demand curve.

If we assume the willingness-to-pay to stay unchanged and the demand curve to be linear, the change in the demand curve is equivalent to a translation of the maximum amount of service, as illustrated in Figure 1. When the customer releases the personal information both the marginal demand (i.e., the increase in demand for a decreasing unit change in price) and the maximum consumption increase by the factor $(1 + \alpha)$, where $\alpha > 0$ is the marginal demand factor and is related to the amount of information released. The new demand curve passes through the points $(0, p^*)$ and $(q^*(1 + \alpha), 0)$, so that its equation is now

$$\frac{q}{q^*(1 + \alpha)} + \frac{p}{p^*} = 1. \quad (2)$$

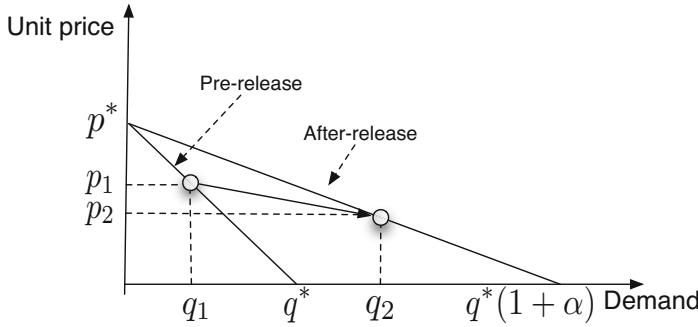


Fig. 1. The demand curve before and after the release of personal data

Since the release of information exposes the customer to the risk of data breach and the resulting money loss, which is an observable quantity, we can relate the marginal demand factor to the money loss. Namely, if we indicate the potential money loss by L , and assume that both the information and the money loss are upper bounded by the quantities α_{\max} and L_{\max} respectively, we can use a power law to describe the relationship between information and money loss

$$\alpha = \alpha_{\max} \left(\frac{L}{L_{\max}} \right)^\nu \quad 0 < \nu < 1. \tag{3}$$

In addition to its well known property of scale invariance and its appearance in a number of contexts (see, e.g., [7][8]), the choice of a power law allows us to describe a variety of behaviours by acting on the single parameter ν . If we make the assumption that the information is released starting with the most potentially damaging, the additional risk associated to further releases is a decreasing function of the information released, and we may postulate a law of diminishing risks, which leads to $\nu < 1$. Within the range $\nu \in [0, 1]$ we can describe different degrees of ability of the service provider to profile its customers. We call ν the privacy parameter. If $\nu \ll 1$ (i.e., the service provider is privacy-friendly), the customer gains a large benefit (a large extension of the maximum quantity of services) even a for small amount of information released (i.e., small potential losses). If $\nu = 1$ we have instead a linear relationship between the information released and the associated economical loss.

The surplus obtained by the customer is the cumulative difference between the price deriving from the demand law and the price \hat{p} set by the service provider, which determines the demand \hat{q} through (2):

$$\hat{S}_c = \int_0^{\hat{q}} (p - \hat{p})dq. \tag{4}$$

By solving the integral (4), we get the final expression for the surplus

$$\hat{S}_c = \frac{(p^* - \hat{p})^2}{2p^*} q^* \left[1 + \alpha_{\max} \left(\frac{L}{L_{\max}} \right)^\nu \right]. \tag{5}$$

3 Ex-ante Regulation of Damage Sharing

A major justification for ex-ante regulation is that the service provider is partially responsible for the overall level of security, and should be held liable for data breaches impacting on the customer. Formulating the regulation policy requires the a priori evaluation of the risk incurred by the customer and its relationship to security investments by the service provider. In this section, we review the risk model associated to data breaches and define the ex-ante regulation policy.

The release of personal information exposes the same customer to the risk of a data breach, quantified through the probability of data breach P_{db} .

We consider that a data breach may take place because of deficiencies on either side of the customer-service provider relationship. The data theft may be due either to an attack on the service provider's information system or to the customer's data repository (e.g., its computer). We assume that the failures on the two sides are independent of each other, and that a data breach takes place as either of the two sides fail. Under these hypotheses, a suitable model for the overall data breach phenomenon is the classical series combination of two systems that we can borrow from the reliability field (see Ch. 3.2 in [9]). The data breach probability P_{db} is then related to the individual data breach probabilities $P_{\text{db}}^{(s)}$ (service provider) and $P_{\text{db}}^{(c)}$ (customer) by the formula

$$P_{\text{db}} = P_{\text{db}}^{(s)} + P_{\text{db}}^{(c)} - P_{\text{db}}^{(s)} \cdot P_{\text{db}}^{(c)}. \quad (6)$$

As to the probability of data breach on the customer's side, we consider it to be a growing function of the amount of personal information that the customer has divulged. We assume a simple power law function to hold, and, by exploiting again the money loss as a proxy for the amount of information released, we obtain the following function:

$$P_{\text{db}}^{(c)} = P_{\text{max}}^{(c)} \left(\frac{L}{L_{\text{max}}} \right)^\theta, \quad (7)$$

where $P_{\text{max}}^{(c)}$ is the probability of breach corresponding to the maximum release of information. The security parameter $\theta \in (0, 1)$ describes the balance between the probability of breach and the quantity of personal information released (for which the economical loss represents a proxy): if $\theta \ll 1$ (reckless customer) the probability of data breach is close to its maximum even for the smallest amount of released information; if $\theta \simeq 1$ (privacy-aware customer), the customer has to release a substantial amount of information before it suffers a significant probability of data breach.

On the other hand, the probability that a data breach occurs on the service provider's side is related to the amount of investments on security spent by the service provider. Namely, we expect that probability to decrease as the investment grows. Again, we assume the following power law to hold

$$P_{\text{db}}^{(s)} = P_{\text{max}}^{(s)} \left[1 - A \left(\frac{I}{I_{\text{max}}} \right)^k \right], \quad (8)$$

where I and I_{\max} are respectively the actual investment and that corresponding to the maximum achievable security, both expressed per customer. On the service provider's side, the probability of data breach ranges then between $P_{\max}^{(s)}(1 - A)$ and $P_{\max}^{(s)}$.

Under the probability P_{db} of data breach, the expected loss for the customer is $P_{\text{db}}L$. The model for data breach risk we have just introduced shows that the service provider may be responsible for that data breach. If it is not held liable for the resulting damage to the customer, it has no incentives to invest in security and reduce the probability of data breach.

Those incentives may be set through a regulation policy. A distinction commonly employed is between ex-ante and ex-post regulation. In ex-ante regulation, the regulator's intervention takes place before the socially undesirable outcome. Instead, in ex-post regulation, the regulator's intervention is spurred by a claim coming from the parties involved (one or both) after the undesirable event has taken place.

In this paper, we consider an ex-ante policy, where the regulator sets the policy beforehand. Since the result of an unsatisfactory security management is a loss of money for the customer, the ex-ante regulation policy consists in the proper apportionment of that damage. A simple damage sharing mechanism consists in attributing a fraction ηL of the money loss to the service provider, while the remaining portion $(1 - \eta)L$ is left to the customer. We call η the damage sharing factor: the larger it is, the more the service provider bears the consequences of careless security management.

4 Surplus Functions

In Section 2, we have evaluated the surplus gained by the customer when buying the service at the price set by the service provider (the customer acts as a price taker). In Section 3 we have evaluated the risk deriving from releasing personal information, and have introduced a damage sharing policy that the regulator can put into place to induce the service provider into investing in security. In this section, we make use of that information to compute the net surplus for the customer and the service provider, which will allow us to define the best strategies for both stakeholders.

4.1 The Customer

When the customer chooses the personal information to release and buys services from the service provider at the price \hat{p} , it gets the surplus expressed by Equation (5). The same release of personal information exposes the customer to the risk of losing money, as described in Section 3. If the regulator adopts the damage sharing policy described in that section, the customer suffers just a fraction of the actual loss, since the rest is charged to the service provider.

The net surplus is the difference between the surplus gained, by purchasing the service at a price lower than the willingness-to-pay, and the fraction of the incurred loss. Its complete expression is

$$S_c = \frac{(p^* - \hat{p})^2}{2p^*} q^* \left[1 + \alpha_{\max} \left(\frac{L}{L_{\max}} \right)^\nu \right] - (1 - \eta)LP_{\text{db}}. \quad (9)$$

4.2 The Service Provider

By setting the unit price \hat{p} and spending the unit cost \hat{c} , the service provider cashes $\hat{p} - \hat{c}$ for each unit of service sold. But that profit is reduced by the security investment I (per customer) and the fraction of the loss suffered by the customer, as set by the damage sharing policy issued by the regulator.

Its net surplus is then

$$S_{\text{sp}} = \frac{p^* - \hat{p}}{p^*} q^* \left[1 + \alpha_{\max} \left(\frac{L}{L_{\max}} \right)^\nu \right] (\hat{p} - \hat{c}) - I - \eta LP_{\text{db}}. \quad (10)$$

5 A Game Formulation for Investments and Risk

In Section [4](#), we have seen that both the service provider and the customer derive a gain, respectively from the sale and from the purchase of services. But they also share the risk associated to information release, through the damage sharing policy enforced by the regulator. In addition, the service provider is induced into investing in security to reduce the loss deriving from the damage sharing policy. The surplus functions of both stakeholders present both positive and negative components. And both stakeholders are called to act on strategic leverages to maximize their profit. Each move by either stakeholder influences the outcome for the other. Their interaction may be modelled as a non-cooperative game. Namely, each player can derive its best response (i.e., the optimal value of its strategic leverage) to the move of its opponent (i.e., to the value the opponent has set for its strategic leverage). In this section, we derive the best response functions for both players.

5.1 Customer's Best Response Function

The customer can act on the amount of personal information that it releases, as a strategic leverage. When releasing more personal information, the customer receives a benefit and a disadvantage at the same time. Its surplus grows because of the movement of the demand curve, due to unit price reductions or demand increase for the same price or both. But the release of information also increases the customer's exposure to the risk of information leak and the subsequent money loss.

We can obtain the best response function, by looking for the value of the amount of information released that maximizes the net surplus. However, rather

than resorting to the information amount, we adopt again the money loss as a proxy. In addition, in order to obtain parametric expressions, we normalize both strategic leverages to their maximum value: we introduce the variables $X = L/L_{\max}$ and $Y = I/I_{\max}$.

By adopting such normalization, the customer’s surplus function (9) can be expressed as follows.

$$S_c = \frac{(p^* - \hat{p})^2}{2p^*} q^* [1 + \alpha_{\max} X^\nu] - (1 - \eta) X L_{\max} P_{db}. \tag{11}$$

Since $\partial S_c / \partial X = L_{\max} \partial S_c / \partial L$, zeroing the derivative $\partial S_c / \partial L$ is tantamount to zeroing $\partial S_c / \partial X$. We obtain the best value of the amount of information released

$$X_{\text{opt}} = X : \partial S_c / \partial X = 0. \tag{12}$$

In deriving Equation (11), it is convenient to obtain the inverse of the customer’s best response function, where the service provider’s strategic leverage (the level of investments) is expressed as a function of the customer’s strategic leverage (the amount of money loss). We obtain

$$Y = \left[\frac{1}{A} - \frac{\Delta X_{\text{opt}}^{\nu-1} - L_{\max}(1 - \eta) \Lambda X_{\text{opt}}^\theta}{\Upsilon(1 - \Lambda X_{\text{opt}}^\theta)} \right]^{1/k}, \tag{13}$$

where we have used the following positions

$$\begin{aligned} \Delta &= \frac{(p^* - \hat{p})^2}{2p^*} q^* \alpha_{\max}^\nu \\ \Lambda &= P_{\max}^{(c)} (1 + \theta) \\ \Upsilon &= P_{\max}^{(s)} A L_{\max} (1 - \eta) \end{aligned} \tag{14}$$

We remark that, when using Equation (13), we should use just those values for which the following condition holds

$$\frac{1}{A} - \frac{\Delta X_{\text{opt}}^{\nu-1} - L_{\max}(1 - \eta) \Lambda X_{\text{opt}}^\theta}{\Upsilon(1 - \Lambda X_{\text{opt}}^\theta)} > 0. \tag{15}$$

5.2 Service Provider’s Best Response Function

The service provider’s strategic leverage is the amount of investments in security. The more it spends on security, the less it has to cover for data breaches through the damage sharing policy.

By adopting the normalization introduced in Section 5.1, the service provider’s surplus function is expressed as

$$S_{\text{sp}} = \frac{p^* - \hat{p}}{p^*} q^* [1 + \alpha_{\max} X^\nu] (\hat{p} - \hat{c}) - Y I_{\max} - \eta X L_{\max} P_{db}. \tag{16}$$

Again, since zeroing the derivative $\partial S_{sp}/\partial I$ is tantamount to zeroing $\partial S_{sp}/\partial Y$, we obtain the optimal value of the amount of investments as

$$Y_{opt} = Y : \partial S_{sp}/\partial Y = 0. \tag{17}$$

The best response function for the service provider is then

$$Y_{opt} = \left[\frac{\Phi X \left(1 - P_{max}^{(c)} X^\theta \right)}{I_{max}} \right]^{\frac{1}{1-k}}, \tag{18}$$

where $\Phi = \eta P_{max}^{(s)} A k L_{max}$.

6 Analysis of Nash Equilibrium

In Section 5, we have seen how each player responds in an optimal way to the decision taken by the other player. In this case, the service provider plays by setting its level of investments, while the customer plays by deciding how much information it releases (by using the risk exposure as a proxy). The two best response functions can be formulated each as a function of the strategic leverage employed by the opponent, so that we have $X_{opt} = f(Y)$ and $Y_{opt} = g(X)$. Equation (18) provides us with the function $g(\cdot)$, while Equation (13) provides us with the inverse function $f^{-1}(\cdot)$. A Nash equilibrium is achieved for any couple of values (X^*, Y^*) for which we have

$$\begin{aligned} X^* &= f(Y^*) \\ Y^* &= g(X^*). \end{aligned} \tag{19}$$

In this section, we examine if the game we have described in the previous sections admits a Nash equilibrium. We solve the system of equations (19) numerically.

We define a reference scenario, by setting the parameters' values on the basis of market and regulatory reports, as reported in Table 1.

For the reference case, we obtain the best response functions shown in Fig. 2. We see that both functions are monotone growing and a single Nash equilibrium point exists. We have examined what happens in a variety of cases, with perturbations around the reference case. In all cases, we have found either a single Nash equilibrium or no equilibrium at all.

We report here the impact of two major parameters on the Nash equilibrium: the damage sharing factor η and the data breach probability.

All the other parameters being equal to the reference case, we have varied η in the (0.5, 0.8) range. At the lower bound, the customer and the service provider share the money loss resulting from the data breach in equal proportions. Instead, when $\eta = 0.8$, the service provider pays most of the toll. In Fig. 3, we see how the equilibrium point moves (we have a single equilibrium point throughout the range). Increasing the burden on the service provider brings it to increase its

Table 1. Parameters' values for the reference case

Parameter	Value
η	0.75
L_{\max}	10000 €
I_{\max}	5 €
p^*	2 €
\hat{p}	1 €
q^*	600
\hat{q}	300
α_{\max}	0.15
$P_{\max}^{(s)}$	10^{-3}
$P_{\max}^{(c)}$	10^{-3}
k	0.5
A	0.9
ν	2/15
θ	2/15

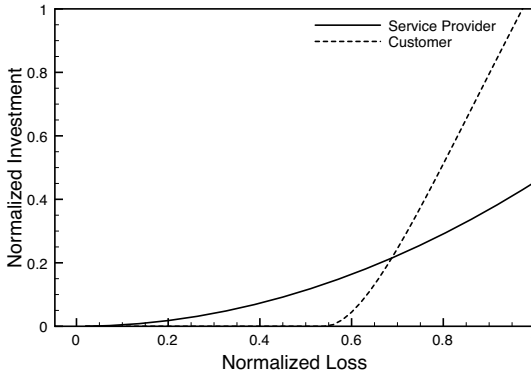


Fig. 2. Best response functions in the reference case

investment in security and the customer to release its personal data more easily. However, when $\eta > 0.8$ there is no Nash equilibrium: the damage sharing policy cannot be stretched too far at the service provider's disadvantage.

Instead, we have examined the effect of data breach probability by increasing the maximum data breach probability on both sides ($P_{\max}^{(s)}$ and $P_{\max}^{(c)}$), from $5 \cdot 10^{-4}$ to $5 \cdot 10^{-3}$. In Fig. 4, we see that the equilibrium is reached when the data breach probability is not too low. The behaviour of the service provider is affected very little by the increase in the data breach probability, while the decade change in both data breach probabilities brings the customer to span all its range of behaviours.

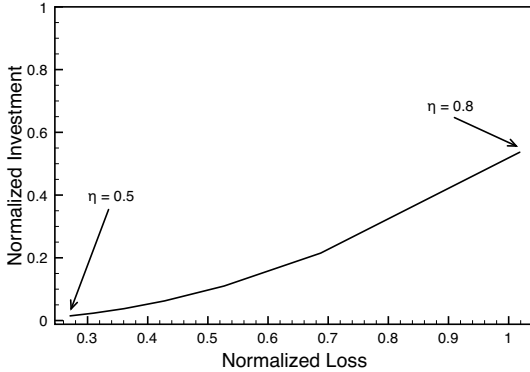


Fig. 3. Impact of damage sharing factor on the equilibrium

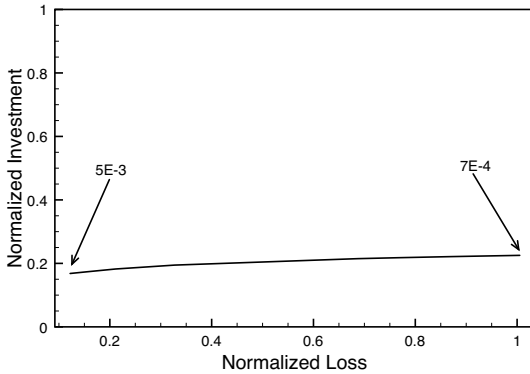


Fig. 4. Impact of data breach probability on the equilibrium ($P_{\max}^{(s)} = P_{\max}^{(c)} \in (7 \cdot 10^{-4}, 5 \cdot 10^{-3})$)

7 Conclusions

We have provided a game-theoretic formulation of the strategic interaction between a customer and its service provider, when both have an interest in the level of security and a damage sharing policy is in place for apportioning the money loss resulting from a data breach. We have provided the analytical expressions of the respective best response functions, where the service provider can choose its level of investment in security, and the customer can choose its level of exposure related to the amount of personal information released. The game's outcome can be used to help formulate the regulatory policy. The presence of Nash equilibrium can be examined numerically. For all the cases examined, the game never exhibits more than a single Nash equilibrium point. Increasing the quota of money loss apportioned to the service provider spurs it to increase its investment in security, and the customer to release more personal data, but no equilibrium is reached when the damage sharing factor grows beyond a threshold. If the

damage sharing factor is kept at not-too-unbalanced values (e.g., lower than 75%), the incentive to invest in security is however quite modest (no more than 30% of the maximum envisaged). Instead, the behaviour of the service provider is relatively unaffected by changes in the data breach probability.

References

1. Gordon, L.A., Loeb, M.P.: The economics of information security investment. *ACM Trans. Inf. Syst. Secur.* 5(4), 438–457 (2002)
2. Lee, Y.J., Kauffman, R.J., Sougstad Profit-maximizing, R.: firm investments in customer information security. *Decision Support Systems* 51(4), 904–920 (2011)
3. Jiang, L., Anantharam, V., Walrand, J.C.: How bad are selfish investments in network security? *IEEE/ACM Trans. Netw.* 19(2), 549–560 (2011)
4. European Network and Information Security Agency (ENISA). *Economics of Security: Facing the Challenge* (2011)
5. Mankiw, N.G.: *Principles of Microeconomics*, 3rd edn. South-Western College Pub. (2003)
6. Varian, H.: Economic aspects of personal privacy. In: Lehr, W.H., Pupillo, L.M. (eds.) *Internet Policy and Economics*, pp. 101–110. Springer (2009)
7. Newman, M.: Power laws, Pareto distributions and Zipf's law. *Contemporary Physics* 46, 323–351 (2005)
8. Roberts, D.C., Turcotte, D.C.: Fractality and self-organized criticality of wars. *Fractals* 6(4), 351–357 (1998)
9. Gnedenko, B., Ushakov, I.: *Probabilistic Reliability Engineering*. John Wiley & Sons (1995)

Optimizing Network Patching Policy Decisions

Yolanta Beres and Jonathan Griffin

HP Labs, Bristol, UK

{yolanta.beres, jonathan.griffin}@hp.com

Abstract. Patch management of networks is essential to mitigate the risks from the exploitation of vulnerabilities through malware and other attacks, but by setting too rigorous a patching policy for network devices the IT security team can also create burdens for IT operations or disruptions to the business. Different patch deployment timelines could be adopted with the aim of reducing this operational cost, but care must be taken not to substantially increase the risk of emergency disruption from potential exploits and attacks. In this paper we explore how the IT security policy choices regarding patching timelines can be made in terms of economically-based decisions, in which the aim is to minimize the expected overall costs to the organization from patching-related activity. We introduce a simple cost function that takes into account costs incurred from disruption caused by planned patching and from expected disruption caused by emergency patching. To explore the outcomes under different patching policies we apply a systems modelling approach and Monte Carlo style simulations. The results from the simulations show disruptions caused for a range of patch deployment timelines. These results together with the cost function are then used to identify the optimal patching timelines under different threat environment conditions and taking into account the organization's risk tolerance.

1 Introduction

Security decisions often involve trade-offs: a security policy choice that optimizes time spent by the security team might create burdens (cost) for IT operations or the business. One of the main tasks faced by the security operations team is vulnerability and patch management. The reasoning behind applying patches to remove system vulnerabilities thereby reducing security risk is well understood. The longer the systems stay unpatched the bigger the risk that a vulnerability may be exploited by malicious attacks or fast spreading malware. However, applying patches usually has many undesirable implications, mainly in terms of business disruptions. These disruptions are particularly high when patching network devices such as routers and switches, and especially so in such highly network reliant environments, as cloud infrastructures. Any time a piece of network equipment is patched, there is a risk of something going wrong: if the patch fails, or results in unexpected interaction with other devices or current configurations, the disruptions caused can have significant impact on the business, which relies on the network infrastructure. For example, patching Cisco devices usually requires the replacement of the complete device operating system; this

is very different from the patching of Windows servers. After patching, it could easily turn out that the existing configuration does not work with the new OS version, or a routing protocol might end up being broken.

When setting their patching strategies, many organizations adopt regular patch update cycles, where the patches are required to be deployed once within a set time period, be it within one, three, or six months. Since patching automation for network devices is currently still technically difficult¹, this time-bound schedule is used to plan and allocate time periods for manually patching all necessary network devices within the schedule. The chosen patching schedule has to optimally address two objectives: minimize business disruptions from planned patches and upgrades, and minimize the time and the number of devices that remain exposed due to being unpatched for a long time.

These two objectives present a conflict, however. To reduce the number of planned disruptions due to patching the operations staff would prefer to adopt patching schedules that span longer periods of time, as more time can be spent on thoroughly testing each patch and also due to the fact that several patches may be released by the vendor when waiting longer and so several patches can be batched together and applied at the same time, reducing the level of disruption from patching. However, from a threat perspective the longer policy would increase the exposure of network devices to potential exploits and attacks due to many devices remaining vulnerable for long periods.

In this paper we explore how IT security policy choices regarding patching schedules for network equipment can be made in terms of economically-based decisions, in which the aim is to minimize the expected overall cost to the organization from patching-related activity. We apply a methodology that combines methods from economics—specifically, cost and utility functions—together with simulations of the executable system model capturing the underlying processes.

First, we introduce a simple cost function that takes into account costs incurred from disruption caused by planned patching and from expected disruption by an emergency when an exploit or malware emerges. We then construct a system model of the patch management processes, which is executed in the context of a stochastic model of the threat environment. To gain insight into the actual network patch management processes, we have worked with security teams and network operations staff across a couple of large organizations. In our model, we capture the main attributes of this process, but include some simplifications to make the model computationally feasible. We also make assumptions about the characteristics of the threat environment specific to network exploits and attacks, mostly based on historical (though sparse) data on network exploits over the past several years since 2004.

Finally we perform the experimental simulations, and show how changes in the patch deployment schedules affect the planned and unplanned disruption levels. These results together with the cost function are then used to suggest the optimal patching schedules for different tolerance levels of disruption and risk. Since the threat environment is ever changing, we perform additional simulations to show how the patching schedules should be adjusted under worsening threat conditions.

¹ Due to high variability of network device OS types, and due to the fact that many patches require upgrades of OS, and thus might require multiple reboots, and possible re-configurations.

Our paper is organized as follows. Section 2 describes the network patching problem and introduces the cost function. In section 3, we present the model, constructed to capture the patch management process in the network environment. Section 3 also describes how we model the external threat environment. In section 4, we describe results and analysis from a set of simulation experiments based on the constructed model and under the assumptions of the introduced cost function. The analysis shows the changing levels of planned patching and emergency disruptions under different patch deployment timelines, and suggests some optimal timelines for the certain emergency tolerance level. Section 5 describes results from another set of experiments with a changing threat environment, and looks at how this affects the optimal timeline choice. In section 6, we discuss the implications of our analysis and some future work. Section 7 reviews related work in this area. Our paper finishes with some final conclusions.

2 Disruption Trade-Offs

In this paper we examine two types of disruptions that security teams dealing with vulnerability management across network devices have to encounter:

1. Planned operational disruptions that are caused by the device downtime due to patching. These can range from relatively short downtime due to device reboot to longer downtime where the patch failed and requires re-application or changes in the system configuration.
2. Unplanned disruptions that are caused by deployment of emergency procedures whenever there is emergence of exploit or malware. Even larger scale disruptions are encountered when the network is actually hit by attack that exploits unpatched vulnerabilities. The cost of emergency procedures is much higher than the planned disruption caused by patching. These emergency procedures could encompass expedited patching, or deployment of emergency workarounds. And so the savings in reduced operational disruption achieved with longer patch deployment timelines have to be weighed against the potential increase these type of disruptions.

2.1 Cost Function

Based on the two forms of business disruption introduced above we define a cost function that is used to determine the acceptable trade-offs when choosing the patch deployment schedules.

We use the following notation:

- c_{patch} is the cost of applying a patch to a single device (or upgrading the OS of a device), which for the purpose of this paper is mainly the disruption caused to the business because the device is offline and not usable;
- $c_{emergency}$ is the cost of applying an expedited fix or a workaround to a single device in case of exploit or actual breach/attack; this again is mainly the disruption caused to the business because the device is not usable;

- $P_{emergency}(t)$ is the probability that an exploit will emerge during some time interval $[t_1, t_2]$, raising the need for an emergency. This probability varies at different points in time t , where t is the time elapsed from the vulnerability disclosure. In section 3.1 we will choose a specific probability density function for our model of the threat environment.

Since an individual organization has hundreds or thousands of devices that might be vulnerable to the same vulnerability and require patching, the overall cost of an individual task of patching is multiplied across the population of all vulnerable devices, denoted as $dev_{patched}$:

$$d_{patch} = c_{patch} dev_{patched}$$

We assume that the cost of applying a patch does not fluctuate significantly across different types of network equipment or from one patch to another. We recognize that in some cases this is a simplification as patch quality may vary, and some network devices have a more critical role and cause more disruption when offline, but to make our analysis and modeling generic rather than organization- or vendor-specific we assume that downtime during patch application is relatively similar across the vulnerable devices.

The cost of an emergency, however, is incurred only if a breach is imminent due to the emergence of an exploit or the detection of an actual attack, and so the emergency disruption is dependent on the number of vulnerable (unpatched) devices at the time of emergency, denoted as $dev_{unpatched}(t)$. The meaning for the cost of the emergency that we use in this paper does not take into account disruptions caused by the actual attacks on the unpatched devices or the implications of more complex attacks that exploit vulnerabilities on the unpatched devices. For the purpose of our analysis we assume that the emergency disruption cost comes from emergency patching of network equipment which happens when an exploit is known about, but before an actual attack takes place (attacks are rare). The resulting disruption is similar in kind to that caused by planned/operational patching, but of much greater severity as the operations and security teams have to rush to deploy patches or emergency workarounds across the remaining unpatched devices causing disruptions outside the agreed allowed downtime periods and more severely impacting organization’s business processes.

The arrival of an emergency event is modeled by the probability $P_{emergency}(t)$. We define the disruption during emergencies, $d_{emergency}$, as the expected value of emergency disruption across the unpatched population of devices:

$$d_{emergency} = c_{emergency} \int dev_{unpatched}(t) P_{emergency}(t) dt$$

Combining the two types of disruption together, the expected overall disruption caused by vulnerability management through patching is the patching cost plus the expected cost of emergency:

$$D = d_{patch} + d_{emergency}$$

The cost of disruption from patching or in case of emergency is obviously organization- and patch-specific, but for our analysis we need to make some simplifications.

We say that the disruption cost per device caused by emergency procedures is α times greater than the disruption caused by applying a patch. This allows us to state that:

$$C_{emergency} = \alpha C_{patch}$$

The actual value of α is organization-specific, and should be selected depending on organization's tolerance level for emergencies, including the procedures for the patch deployment escalations, deployment of workarounds, or redundancies built into the network that might minimize the disruptions caused by an actual attack.

By substituting this into previous equation we have:

$$D = c_{patch} (dev_{patched} + \alpha \int dev_{unpatched}(t) p_{emergency}(t) dt)$$

Since c_{patch} is constant, the overall disruption cost depends mainly on the number of devices requiring a patch and the expected number of devices that remain unpatched at the time of an emergency.

This cost is incurred for each patch or batch of patches released by the vendor, so if, for example, a vendor releases three patches in a year that apply across the same population of devices, the cost D triples. In one year an organization usually has to apply hundreds of patches across its various systems and applications. For network devices, the number of patches released by vendors is considerable smaller, usually in tens rather than in hundreds per year, which is still quite a large number, considering that a typical large organization might have hundreds or thousands of network devices.

2.2 Reducing the Cost of Disruption

The security team can reduce the cost of disruption by planning the patch deployment timelines and using patch batching capabilities to bundle several patches together and apply them in one shot. For example, by bundling two patches together the administrator would cause half as much disruption, as each device has to be touched once instead of twice. The number of patches that can be bundled together is dependent on the vendor's patch release lifecycle. In order to meet the deadlines set in the patch deployment policies the administrators would usually start applying one patch across the first set of devices, and once another patch is released will batch it with the previous patch and apply them together across the next set of devices.

Looking at our cost functions the aim of the batching of patches is to reduce the cost d_{patch} for each patch by reducing the number of devices that the patch has to be applied to individually. By incorporating the patch batching effect for each patch, we can subtract from the total patchable population of devices the population for which the patch can be batched with the next or a superseding patch:

$$d_{patch} = c_{patch} (dev_{patched} - dev_{batch_patched})$$

We are making a simplification here by assuming that the disruption caused when applying the batch of patches is the same as when applying a single patch. If the batch includes many complex patches, this might not be exactly the case. However, it's common within the network context that the next set of network patches is actually a full upgrade that supersedes or includes any previous patches, and so the cost of applying a new upgrade is the same or is increased only by small delta.

By choosing appropriate patch deployment deadlines that correlate with vendor patch release schedules the aim of the security team would be to increase the size of population within $dev_{batch_patched}$, and thus achieve lower overall patch disruption costs. If d_{patch} was the only cost within D the biggest reduction of cost would be waiting for as many patches of possible and applying them together.

The other cost within D , the cost of emergency disruption $d_{emergency}$, however, increases with longer patch deployment timelines, as the population of devices unpatched at the time of an emergency grows. The aim would be to identify the appropriate patch deployment deadline that decreases d_{patch} (the patch has to be applied across minimum number of devices) while not increasing considerably $d_{emergency}$ (minimum number of devices remaining unpatched in case of emergency), and thus *minimizes* the overall disruption D caused by patching.

3 A Systems Model of Network Vulnerability Management

To explore the effect of different patching timelines on the cost of disruption, we use systems modeling and simulations. With the systems modeling the aim is to accurately capture the characteristics and behavior of the vulnerability disclose-exploit-patch lifecycle [1, 12], including the patch testing and patch deployment stages. We use a systems modeling methodology based on process algebra and queuing theory [10] that has been developed for framing and exploring models of complex systems and processes. Within this approach the processes, such as patch deployment, are captured stochastically and events that cause changes in the process, such as vulnerability exploit or patch release, as discrete events whose occurrence is specified with probability distributions or as time-based cycles. The models themselves are developed using specially created Gnosis programming language [25] and are represented visually as an activity type diagrams, with each component encoding certain distinct features of the system.

3.1 Patch Release and Patch Management Processes

We have previously developed a model of the patch management processes to explore the risk exposure window across Wintel environment in an organization [3]. We have adapted this model for network environment by introducing new features such as slower patching rate through patch uptake function and allowing longer patch assessment and preparation time. We have also changed how the threat environment is expressed to reflect the different attacker and exploit developer behavior. Figure 1 shows the visual representation of the created model.

The model diagram should be interpreted as follows: (a) the star and circular shaped components represent the events that occur at regular intervals as defined with an event inter-arrival probability distribution; a circular component has in addition the parameter defining the probability of an event happening; (b) the rectangular shapes correspond to the process steps each with an internal logic, that consists of time duration or some manipulations of internal parameters; (c) the rhombus shapes are if-type decision boxes, that return Boolean values true or false based on a parameter value; and (d) finally, the process dynamics of the model is captured by the arrows connecting events to process steps and decision boxes.

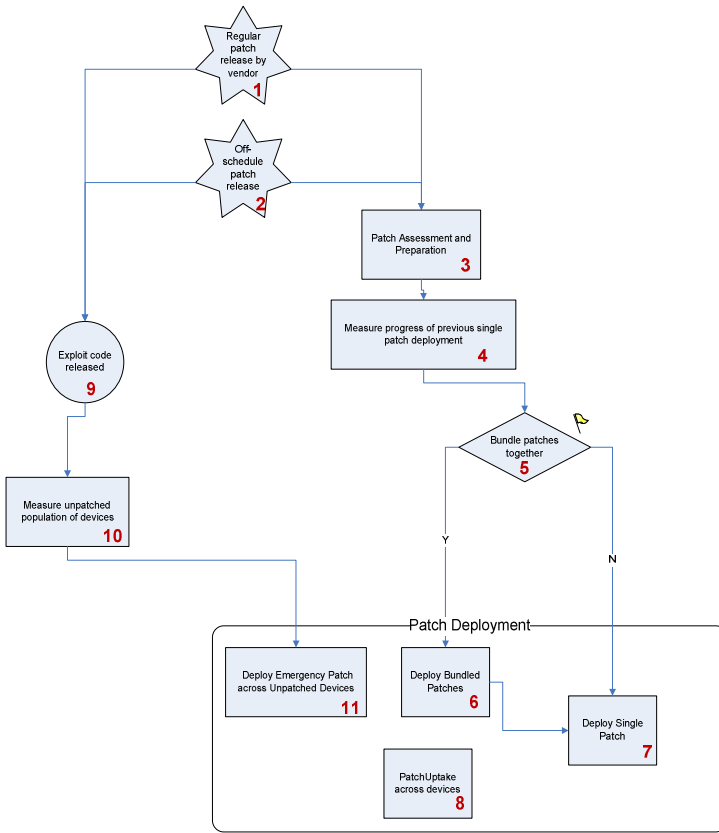


Fig. 1. Visual representation of the developed system model

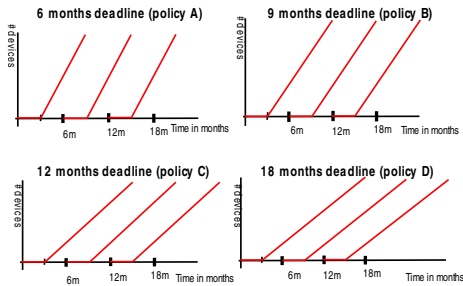


Fig. 2. Linear patch uptake across different patch deployment policies (deadlines)

The main trigger of a vulnerability management processes across networks is usually the release of a patch or a batch of patches by a vendor. This is different from Wintel type environments where such process is usually triggered by the vulnerability announcements, since the threat environment and exploit discovery characteristics are not the same for network platforms. These are discussed in detail in section 3.2.

Some major network equipment vendors have recently adopted regular patch release cycles, and so we decided to include this in our model, but also recognize that off-cycle patches might be released in between. We have examined historical data in Secunia reports [15] regarding the patch release frequency by the three major network device vendors adopted across large enterprises: Cisco, HP ProCurve, and Juniper. Of these three, only Cisco has adopted a regular patch release policy with the main patches being released at 6 months intervals [4], in March and September, and some critical patches in between. Many fewer patches are released by HP ProCurve and Juniper, usually only 1 or 2 per year.

Since Cisco networking equipment is by far the most prevalent across large organizations, we decided to use the six-monthly cycle as the patch release frequency in the component (1) of our model: $f_{patch_regular}(t_{patch})=60$.

We model the arrival of off-cycle patch releases as a Poisson process with an inter-arrival time based on an exponential probability distribution, with the mean time between occurrences set to one year. This is defined in the component (2) in the model: $f_{patch_offcycle}(t_{patch})=365e^{-365t_{patch}}$ ²

The process steps taken internally within an organization to manage patches consist of patch preparation and deployment stages. Based on interviews with the network administrators, we found that large organizations typically allow a fixed time for patch assessment and preparation; in our model this is set to be between 60 and 90 days. This parameter is defined for the component (3) as $f_{patch_prepare}(t_{patch})=U(60,90)$.

Once the patch has been assessed, we measure the progress of the previous patch (component (4)). If the patch has not been deployed across all vulnerable devices (decision component (5)), the patch is bundled with the previous patch and the bundle is deployed across that remaining population (component (6)). A single patch is deployed across all devices that already received the previous patch (if the previous patch has already been deployed across all devices, they would all receive a single patch until the next patch is released and assessed) (component (7)). The components (4)-(7) encode internal measurements, specifically recording the proportion of device population having bundled or single patches installed. The patch deployment is captured within the component (8).

For the patch deployment stage (8) we needed to determine if during a given time period for patch deployment across a number of devices, these devices are patched individually at regular intervals or in groups. By examining the patch deployment practices in several large organizations, we decided to generalize by assuming that in most cases the network devices would be patched individually at regular intervals so as to meet the deadlines set by the organization's policy. This would result in a linear patch uptake during patch deployment over the population of vulnerable systems, as shown in figure 2. We also assumed that the patch uptake has the same characteristics no matter what length the patching policy is set to; e.g. the devices would be patched at equally spaced intervals when the policy deadline is set at 6 months or at 18

² Everything is scaled in days in our model.

months. Based on the lack of automated tools for patching network devices, and the limited number of allowed downtime periods set by the business, we consider that these two assumptions are not far off the actual network device patching practices.

3.2 The Threat Environment

We include the threat environment event in the model that cause an emergency when an exploit appears related to the vulnerability being patched. In choosing how to represent the threat environment in our model, we have examined previously announced cases of network exploits. As we have noted before, exploits on network devices are much less frequent compared to the Wintel environment due to the fact that network devices have many different CPU architectures and multiple ranges of platforms, thus preventing effective automatic exploit development. It is also much harder to reverse engineer the patches when hackers do not have access to the variety of router types or the necessary skills pool.

Up to now, the attacks and exploits that have been publicly announced have targeted specific versions of architecture and platform, making the likelihood of wide-spread attacks very low. Within the years 2002–2009 we have found several, though sketchy, publicly-announced instances of working exploits for Cisco vulnerabilities³. These exploits related to vulnerabilities for which a patch had been released over a year earlier by the vendor. Verizon report on attack vectors observed in 2008 [22] tells similar story, with only a small percentage of hacks observed that targeted routers, switches, or other network devices, and in cases when these exploited vulnerabilities, the patch has been available from a vendor a year or more before. This small number of working exploits together with anecdotal evidence from the hacker community [5, 6, 7, 9, 23] suggests that exploitation of network device vulnerabilities is generally difficult, with working exploits taking significant time to be developed after the patch publication by the vendor.

Based on this analysis, we made the assumption that exploits arrive relatively infrequently and some time after patch publication⁴. Therefore, when representing the threat environment within our system model, we concentrated on two factors, the rate of arrival of exploits, and how long after patch publication the exploit appears—which represents the time it takes for attackers/hackers to develop working exploit code. In the model, for each released patch we perform Bernoulli test on whether an exploit will be developed or not. The rate of arrival of exploits determines the likelihood that an exploit will be developed for any given patch, e.g., if the exploit arrival rate is one per year and the average number of patch releases is three per year, then the probability of an exploit being developed for each patch is 1/3. In the initial

³ March 2004 toolkit exploited vulnerabilities from 2002-2003 [16], Nov 2006 SNMP exploit exploited vulnerability with patch available in 2004-2005 [17], Da Ios rootKit [24] in 2008, Yersinia toolkit released in 2005.

⁴ We recognize that in some cases the vulnerabilities are announced and exploits could be available before the patch is released by the vendor. However, at the moment the evidence suggest that for network vulnerabilities these would be rare cases, especially for an effective exploit to be available much before patch release.

settings of the model we chose the exploit arrival rate as 1 emergency (significantly threatening exploit) every 2 years, as that seems to be closest to the anecdotal evidence available for Cisco vulnerabilities.

As described above, exploits for network equipment generally take some time to be developed. To capture this property we decided to use a Weibull distribution for the exploit development time, with a mean of 1 year and a standard deviation of 0.5 years (resulting in a shape parameter $k=2.10$ and a scale parameter $l=1$).

Thus component (9) in our model has two parameters, namely:

$$p_{\text{emergency(patch)}}=0.16 \text{ and } f_{\text{exploit_delay}(t_{\text{exploit}})}=365 \times 2.10(t_{\text{exploit}})^{1.1} e^{-(t_{\text{exploit}})^{2.1}}$$

These basic settings for parameters regarding the prevailing threat environment are used in what we call the core simulation experiments, the results of which are described in section 4. To reflect potential changes in the threat environment such as increased exploit development rate, or in internal patching processes, we also make various changes in the parameters for additional experiments, which are described in section 5.

After exploit has been released, we measure in our model the proportion of unpatched population (component (10)) that needs emergency patch to be deployed (component (11)).

3.3 Measuring Disruption

During the simulations across different patch deployment schedules, we measure the overall disruption caused by normal patching and emergency procedures. We measure the total disruption caused per year, as this seemed a practical measure that can be used by the security teams in their policy decisions, since many security policies and budgets are determined on yearly basis.

As noted in section 3 the disruption caused by patching mainly depends on the size of the device population that requires a patch to be applied individually. During the simulations, we measure the relative proportion of the population rather than the exact number of devices, as comparisons across different patching policies were done based on the relative increase or decrease in disruption with different patching timelines, rather than the exact number. The same approach was applied for emergency disruption, where we record the proportion of the population remaining unpatched at the time of an exploit.

For example, throughout our set of experiments, the results of which are depicted across the charts in the next couple of sections, the disruption measured ranges from 0 to 10. The disruption of 1 means that full population of devices would be impacted by the operational or emergency patching. In such case, if an organization has 100 network devices, all of these 100 devices will be disrupted, usually by being offline for a certain period of time. The disruption of 3 means that full population of devices would be down three times per year. As was described in section 2, the disruption caused during emergency patching is assumed to be α times greater than the one caused during planned operational patching⁵.

⁵ In the simplest case this would mean that during an emergency disruption a device is down for a longer period compared to the planned operational patch application, as the emergency workaround or patch is more likely to cause failures due to less effort spent on testing.

4 Results from Simulations with Core Model Settings

In the first set of simulation experiments, we look at how disruption changes with increasingly longer patch deployment timelines, with timelines increasing by one month at a time with a maximum of 24 months. The result of these simulations is plotted in figure 3. It shows the mean operational disruption from patching per year (d_{patch}), and the mean emergency disruption ($d_{emergency}$) as longer patch deployment timelines are being adopted by the security operations team.

As can be seen from these plots, with longer timelines the operational disruption decreases quite substantially, while the increase in the emergency disruption is more gradual and smaller. The savings are even bigger after the 6 month timeline. This point corresponds with the 6 month lifecycle when the vendor releases a new patch. For example, when the patch deployment time is set at 8 months the expected operational disruption is half that when patching policy requires patches to be applied immediately, corresponding to the timeline set at 0 months. However, for policies with timelines longer than 15 months the operational disruption improvements are smaller with each longer timeline.

As we recall from section 2, we made an assumption that the disruption cost per device caused by emergency procedures is α times greater than the disruption caused by applying a patch. We stated this as an equation:

$C_{emergency} = \alpha C_{patch}$. When we scale the emergency disruption results by choosing the value $\alpha=10$, we get the third line in the same graph. This is the case when an organization regards disruptions caused by the emergency procedures, be it the patch deployment escalations, deployment of workarounds, or the disruptions caused by an actual attack, as being ten times higher than disruptions caused by the planned patch application. In such cases within the graph we get a crossover point at a timeline of 9 months where $d_{patch} = d_{emergency}$. This represents the patching policy for which the overall disruption is caused by equal measures of operational patching disruption and emergency disruption.

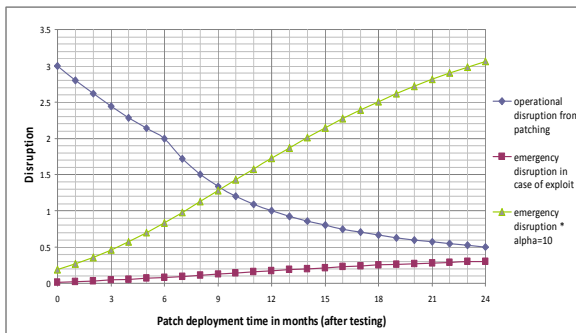


Fig. 3. Operational patching disruption and emergency disruption across changing deployment timeline



Fig. 4. Changes in the overall disruption (operational+emergency) under different values of α

Based on our cost functions from section 2, the optimal patch deployment deadline would be the one where the expected cost of overall disruption, $D = d_{patch} + d_{emergency}$, is minimal for a certain value of α . As the value of α is dependent on a particular organization, its capabilities for dealing with an emergency (such as redundancies across network devices), and its risk appetite, we plotted the overall disruption D under different timelines for several values of α (figure 4).

When $\alpha=10$ the optimal policy is 9 months and this corresponds to the previous crossover point. If α is larger than that, the optimal policy deadline is much shorter, with $\alpha=15$ this being at 3 months and with $\alpha=20$ this being at 2 months. This means that for organizations where emergency disruption is regarded as being more than 10 times worse than the operational disruption caused by normal patch application, the policy should be adopted with patches required to be deployed within a month or two across vulnerable devices.

If, however, emergency disruption is regarded as similar to or just slightly worse than operational disruption⁶, the longer timelines would be more cost effective. For example, when $\alpha=5$, the lowest overall disruption is achieved when the patch deployment deadline is set at 13 months. But even with timelines longer than that, the overall disruption increases only very slightly.

When we run experiments with even longer timelines, with maximum deployment time corresponding to 5 years, the results of which can be seen plotted in figure 5, a point is reached where the amount of emergency disruption exceeds the operational disruption. This is when the timelines are set to longer than 33 months. This suggests that with much longer patch deployment timelines the benefits of reduced disruption become smaller and smaller. The operational disruption cost is reduced only slightly over longer timelines while the emergency disruption cost does not increase, since, based on the assumptions in our model, after 2 years no more exploit is expected for a patch. Also at some point in time batching multiple patches together might not be feasible anymore as there might be physical restrictions that prevent successful installation of new upgrades on old hardware.

⁶ This is quite likely the case within the current threat environment, where past exploits on network equipment have mostly resulted in denial of service (DoS) attacks, rather than attacks that give a complete access to the router or switch. The DoS attacks are usually not regarded as having big impact due to the redundancies that are commonly built in to the network architecture.

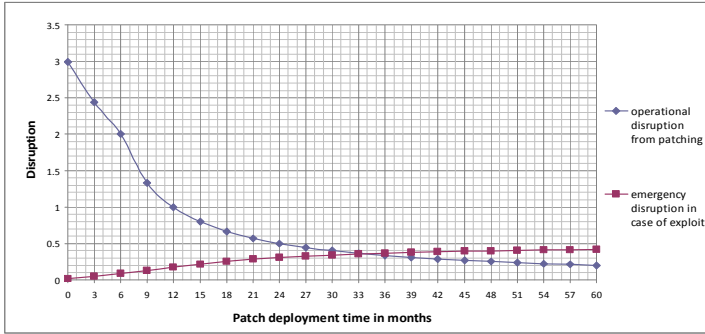


Fig. 5. Operational and emergency disruption with long patch deployment timelines

5 Changes in the Threat Environment

The results described in the previous section were generated from simulations in which the threat environment was as specified in section 4, corresponding to a mean exploit development time of 1 year after patch publication, and an arrival rate of exploits of one every 2 years. With some of the vendors of network equipment aiming to adopt more uniform OS architectures across their range of network devices, developing exploits that impact network devices may become much easier [8], and so the frequency of exploits may increase and the time taken for an exploit to be developed may decrease [13]. The policy which is optimal given current threat conditions may be far from best if the threat level changes. In the next set of simulation experiments we decided to explore how emergency disruption changes under a worsening threat environment, and how the policy deadlines should be adjusted so that to achieve minimal disruption costs.

5.1 Increased Arrival Rate of Exploits

First we increased the arrival rate of exploits, leaving the exploit development time the same as before set at 1 year. The changes in increased emergency disruption for various emergency arrival rates are plotted in figure 6. As can be seen from the chart, the emergency disruption increases considerably as the arrival rate increases, with the highest increase when an exploit appears every 6 months (0.5 year).

When we plot the overall disruption with $\alpha=10$ in figure 7, we can see that with a worsening threat environment the previously optimal patch deployment policy of 9 months no longer applies. Although with the rate of arrival of exploits going up from one per 2 years to one per 1.5 years and one per year, we don't see a substantial increase in the overall disruption, with the rate at one per 0.5 years the increase is much bigger. The best option in such a case would be to patch immediately.

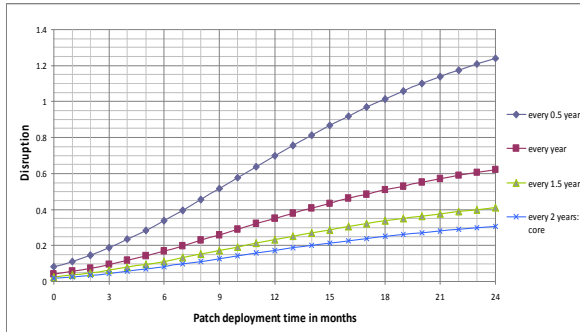


Fig. 6. Emergency disruption for different exploit arrival rates, with exploit development time constant at 1 year

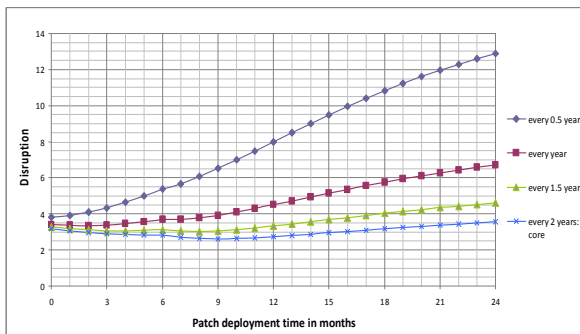


Fig. 7. Overall disruption when $\alpha=10$ and exploit development time is 1 year for different exploit arrival rates

5.2 Faster Exploit Development

When we reduce the mean exploit development time from the initial value of 1 year to 6 months or 3 months, the changes in overall disruption are quite significant, as seen in figure 8.

The results for a mean development time of 6 months are as we might expect, with the earlier exploit arrival times increasing the expected level of emergency disruption, resulting in the optimal policy being to deploy patches as quickly as possible. The results for a mean development time of 3 months seem anomalous at first glance, until we observe that this is the same as the time taken for patch testing, so in this case a large proportion of emergencies arrive before patch deployment has even started. This limits the potential impact of changes in patch deployment timescales and under such threat environment conditions, both the vendors and the organizations would need to re-think the patch release and testing lifecycles and timelines or consider additional mitigation mechanisms.

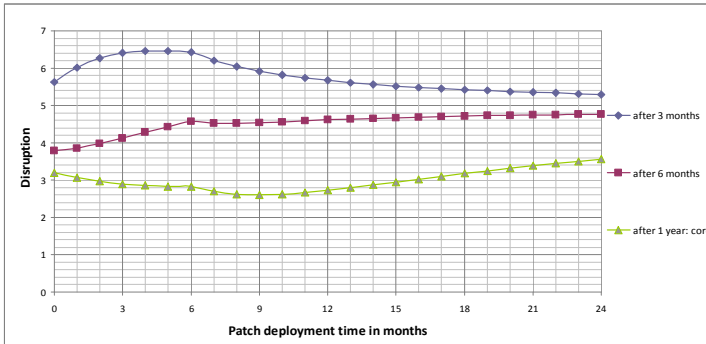


Fig. 8. Overall disruption when $\alpha=10$ and emergency frequency is every 2 years, varying exploit development time.

6 Discussion

Our analysis explored the trade-off between operational and emergency disruption by recognizing that normal patching does introduce disruption, which is not negligible, and this disruption can be reduced with longer patching deadlines, that in turn would potentially increase the risk of a security emergency or breach. This analysis was done under certain assumptions about the parameters in the model. The results are most sensitive to the changing threat environment conditions, and that is why we explored the impact of these changes in detail in the previous section. As the threat environment gets worse, with exploits appearing frequently and soon after patch publication, the trade-off between operational and emergency disruption changes, until eventually the only option is to patch immediately. This would be the case for the Wintel environment, and the disruption from patching in such cases is small compared to emergency disruptions from the constant flow of malware and attacks. The reductions in overall disruption in such cases have to be achieved in a different way, maybe by implementing faster and less disruptive mitigation approaches. Therefore, the analysis in this paper is best suited to the context of vulnerability management for the network environment, or other types of environment where exploits and attacks are less frequent (e.g., some server environments, enterprise applications).

To help determine the appropriate timelines for patch deployment we chose to minimize the cost function that was defined in terms of disruption. In turn, we decided to simplify the definition of disruption so that it was mainly dependent on the size of the population of devices that would be impacted by the specific task: patching or emergency fix. Both of these can be predicted by running the simulations on the system model of the patch management process. To apply the cost function within the context of a specific organization the security team would need to choose only one parameter α , which is an estimation of how much worse the emergency fix is to the operations of the business than the planned patching.

Also our current interpretation of an emergency deals with an expected value that is only dependent on the number of devices unpatched and vulnerable at the time of

the emergency. This might be too simple as the threats to networks become more sophisticated, and impact not only network devices, but critical business applications and transactions. So a more complex definition of emergency disruption might need to be developed, that takes into account the organization's risk appetite and ability to tolerate emergencies. This might also require a more complex system model that captures how an organization reacts to an emergency, including the effect of various processes and security mechanisms that are deployed to deal with the emergency.

7 Related Work

In recent years there has been some work examining the trade-offs involved in different patching policies, but none that specifically address the patching of network devices. Beattie, et al. [2] were the first to explore the factors affecting the best time to apply patches so that organizations minimize disruptions caused by defective patches. Their results indicate that patching during the period of 10 to 30 days after first patch release date is the optimal period for minimizing the disruption caused by defective patches. In our work, rather than looking just at a single patch and adjusting a single point in time to start patch deployment, we analyse the overall patch management process that also takes into account the time taken to apply the patches across all the vulnerable network devices in an organization; this can be considerable for a large organization.

Radianti, et al. [11] explore proactive and reactive patching policies using system dynamics modelling. Although their approach of modelling and simulation is similar to ours, the main difference is in the type of policies that are chosen to explore. Radianti, et al. explore generic patching policies, whereas we aimed specifically at exploring patching of network devices, as these represent a very special case.

Cavusoglu, et al. [18, 19] use a game-theoretic model to study interactions between a vendor releasing patches and an organization deploying the patches across its environment. They examine the cost/benefit consequences of the time-driven release policies adopted by a vendor and similar policies for patch deployment adopted by an organization, and explore situations in which the socially optimal patch management can be achieved. Under the condition that the vendor and patching organization each choose the policies that best suit themselves, they arrive at the conclusion that "vendors are better off releasing patches periodically" and similar regular patch update policies are the optimal strategies for the organization deploying patches. In our paper we assume from the beginning that both the vendor and the organization have adopted regular patch release and deployment cycles, as that has been observed as common practice among vendors and most of the organizations that we have worked with. However Cavusoglu, et al. assume that patch deployment takes an insignificant amount of time compared to the patch release period, whereas this is not the case for patching of network equipment where, as mentioned above, automated patching is not generally available, and patch deployment often takes significantly longer than the vendor's patch release period. Our analysis in this paper is specifically targeted at helping to identify the optimal time period for deploying patches as part of a regular patch cycle.

The work by Zhang, Tan and Dey [14] provides an analytical framework for cost-benefit analysis of different patching policies. They assume that patching lead time (the time taken to apply a patch across the systems) is negligible or very small comparing to the overall patching lifecycle, which we argue is not the case in large organizations with thousands of systems requiring the same patch. The authors also assume that the costs associated with vulnerability exploitations can be estimated with relative ease by an organization, which in reality is very difficult to determine with any accuracy. We believe that our proposed simulation-based approach allows an organization more naturally and flexibly to explore the pragmatic outcomes from different patching policies than a purely analytical framework would allow.

The topic on the cost of administration and cost of system downtime has received some attention [20, 21] and is aimed at identifying the cost to organizations of various administration and operation tasks. Patterson [20] suggests that the cost of downtime should be based upon calculation of two factors: revenue lost and work lost, but can become more complex and include such factors as morale and staff attrition. Couch, et al. [21] explore the actual administration cost incurred in response to various IT support requests. In our paper we take a simplified view of the cost of patching, this basically being the disruption caused to the business because the device is offline and not usable. However, for future work it might be useful to incorporate more complex definitions of cost in our analysis.

8 Conclusions

In this paper we explored how IT security policy choices regarding patching timelines for network equipment can be made in terms of economically-based decisions, in which the aim is to minimize the expected overall costs to the organization from patching-related activity.

We introduced a simple cost function that takes into account costs incurred from disruption caused by planned patching and from expected disruption by an emergency when an exploit or malware emerges. By lengthening the required patch deployment timelines, the IT security policy decision makers can reduce the disruption caused by planned patching as more patches can be batched together, but this would increase the expected emergency disruption as the devices would remain unpatched for longer. We applied a systems modelling and simulation approach to explore the disruptions caused by changing patch deployment timelines within the range of 0 to 24 months, and used the results together with the cost function to identify the optimal patching timeline. When modelling the network vulnerability management process we tried to capture current network patch management practices used across large organizations, and modeled the network threat environment based on historical (though sparse) data on network exploits over the past 4 years. The resulting optimal patch deployment policy of 9 months should be viewed as optimal under these assumptions. By increasing the frequency of exploits in the next set of experiments we saw that this 9 month timeline soon stops being an optimal policy, with the best option being to patch immediately.

We believe that the analysis described in this paper provides guidelines for the IT security policy decision makers in their respective organizations that can be applied when deciding on the network equipment patching policy that is optimal for their organization and their IT environment, and reflects their risk appetite and network emergency tolerance level. It is our hope that this approach may one day form best practice to follow not just in choosing patching policy but in other areas of security decision-making.

This work has been done as part of the Cloud Stewardship Economics Project [26], funded by Technology Strategy Board of UK Government.

References

1. Arbaugh, W.A., Fithem, W.L., McHugh, J.: Windows of Vulnerability: A Case Study Analysis. *IEEE Computer* (2000)
2. Beattie, S., Arnold, S., Cowan, C., Wagle, P., Wright, C., Shostack, A.: Timing the Application of Security Patches for Optimal Uptime. In: *Proc of LISA 2002: 16th System Administration Conference* (2002)
3. Beres, Y., Griffin, J., Heitman, M., Markle, D., Ventura, P.: Analysing the Performance of Security Solutions to Reduce Vulnerability Exposure Windows. In: *Proc. of 2008 ACSAC* (December 2008)
4. Cisco Security Advisories and Notices, The publication schedule for Cisco Internetwork Operating System (IOS) Security Advisories (March 2006)
5. Felix “FX” Lindner, “Cisco Vulnerabilities”, *BlackHat Federal* (2003)
6. Felix “FX” Lindner, “Development in Cisco IOS Forensics”, *Defcon 11* (2003)
7. Technical interview, “Exploiting Cisco with FX” (2005), <http://www.securityfocus.com/columnists/351/2>
8. Felix “FX” Lindner, Cisco IOS-Attack and Defense: State of the Art. In: *25th Chaos Communication Congress* (December 2008)
9. Lynn, M.: The Holy Grail: Cisco IOS shellcode and exploitation techniques, *Black Hat USA* (July 2005)
10. Pym, D., Monahan, B.: A Structural and Stochastic Modelling Philosophy for Systems Integrity. *HP Labs Technical Report* (2006)
11. Radianti, J.: Assessing Risks of Policies to Patch Software Vulnerabilities. In: *Proc. Of International System Dynamics Conference* (July 2006)
12. Schneier, B.: *Managed Security Monitoring: Closing the Window of Exposure*, Counterpane
13. Symantec Global Internet Security Threat Report: Trends for July–December 2007, vol. XII (April 2008)
14. Zhang, G., Tan, Y., Dey, D.: Optimal Policies for Security Patch Management, under review
15. Secunia Advisories by Vendor, <http://secunia.com/advisories/>
16. InfoWorld News, Cisco warns of new hacking toolkit (March 2004)
17. NetworkWorld News, Can anyone stop the Cisco exploit? (November 2006)
18. Cavusoglu, H., Cavusoglu, H., Zhang, J.: Economics of Security Patch Management. In: *Workshop on Economics of Information Security (WEIS)* (June 2006)
19. Cavusoglu, H., Cavusoglu, H., Zhang, J.: Security Patch Management—Share the Burden or Share the Damage? *Management Science* 54(1) (April 2008)

20. Patterson, D.: A simple model of the cost of downtime. In: Proceedings of Large Installation System Administration Conference (LISA 2002), USENIX Assoc. (2002)
21. Couch, A.L., Wu, N., Susanto, H.: Toward a Cost Model for System Administration. In: Proceedings of Large Installation System Administration Conference (LISA 2005). USENIX Assoc. (2005)
22. Verizon 2009 Data Breach Investigations Report (2009)
23. 'FX' Lindner, F.: Cisco IOS Router Exploitation. Blackhat (2009)
24. 'topo' Muñiz, S.: Killing the myth of Cisco IOS rootkits (May 2008), http://www.coresecurity.com/files/attachments/Killing_the_myth_of_Cisco_IOS_rootkits.pdf
25. Collinson, M., Monahan, B., Pym, D.: A Discipline of Mathematical Systems Modelling. HP and College Publications (2012)
26. The Cloud Stewardship Economics Project, IISP, <https://www.instisp.org/SSLPage.aspx?pid=463>

A Risk Assessment Method for Smartphones

Marianthi Theoharidou, Alexios Mylonas, and Dimitris Gritzalis

Information Security and Critical Infrastructure Protection Research Laboratory
Dept. of Informatics, Athens University of Economics and Business (AUEB)
76 Patission Ave., Athens, GR-10434 Greece
{mtheohar, amylonas, dgrit}@aueb.gr

Abstract. Smartphones are multi-purpose ubiquitous devices, which face both, smartphone-specific and typical security threats. This paper describes a method for risk assessment that is tailored for smartphones. The method does not treat this kind of device as a single entity. Instead, it identifies smartphone assets and provides a detailed list of specific applicable threats. For threats that use application permissions as the attack vector, risk triplets are facilitated. The triplets associate assets to threats and permission combinations. Then, risk is assessed as a combination of asset impact and threat likelihood. The method utilizes user input, with respect to impact valuation, coupled with statistics for threat likelihood calculation. Finally, the paper provides a case study, which demonstrates the risk assessment method in the Android platform.

Keywords: Smartphone, Risk Assessment, Android, Security, Threat.

1 Introduction

Smartphones' popularity lies mainly with their pervasiveness, which stems from their small size, advanced processing and connectivity capabilities, reduced cost, and their ability to host multi-purpose third party applications. Smartphones host heterogeneous data such as multimedia, sensor data, communication logs, data created or consumed by applications, etc. A smartphone user carries the device on multiple locations throughout the day, and allows connections to various networks that are often not secure. As the same device may be used for both, work and leisure purposes, smartphones often contain a combination of valuable personal and business data.

The complexity of administrator attempts to secure organisation assets rises, as users continue to bring their own smartphones in corporate premises [17]. Often, organizations are not prepared to manage smartphone heterogeneity, especially when the required resources or expertise is not present (consumerization). Smartphones extend the business perimeter, while existing security and privacy perimeter-oriented mechanisms are inadequate [18]. In this context, the importance of smartphone data, in conjunction with their ability to interact with corporate assets, make them economically attractive to attackers [2]. Hence, traditional risks (e.g. theft, fraud, etc.) may reappear with increased impact. They can pose a security challenge [4] or take place using new attack vectors, e.g., using smartphone location capabilities for

surveillance [16]. In addition, smartphones implement different security models, which make traditional countermeasures ineffective.

Traditional risk assessment methods treat smartphones as an asset of a business information system, similarly to a personal computer or a laptop. They treat the smartphone as a single entity, where threat and vulnerability assessment are performed on the asset as a whole. Although a smartphone can be viewed as a kind of small scale information system, making existing methods applicable, such an assessment is not ideal for risks that target device (sub)assets, e.g. GPS sensor (i.e. surveillance), logs (i.e. call logs disclosure), etc. This is due to the fact that they do not take account smartphone-specific threats, neither the unique vulnerabilities that a smartphone security model introduce. Furthermore, most risk assessment methods are not intended for users, but mainly for businesses. Thus, a targeted risk assessment method is useful, so as to assess user-specific parameters and smartphone-specific threats, in a considerably more fine-grained fashion. We contribute towards this direction by identifying smartphone assets and threats, as well as by proposing a risk assessment method specifically tailored for smartphones.

The paper is organized as follows. Section 2 provides the reader with a smartphone definition and smartphone data taxonomy. In Section 3 the proposed risk assessment method is introduced. The method is applied in Section 4, through the use of a case study. Method limitations and further research ideas are discussed in Section 5.

2 Smartphone: Definition and Assets

The term *smartphone* is frequently used by the industry and research community to refer to state-of-the-art cell phone devices. These devices are considered ‘smart’, and are distinguished from ordinary and technologically constrained cell phones. The latter, which are referred to as *feature phones*, are often restrained by small screen size, limited processing and network capabilities, and execute, in general, a proprietary and not adequately documented operating system. Thus, their security is mainly based on secrecy or as the IT security community refers to, as “security by obscurity”.

In contrast with the term ‘feature phone’, a widely accepted definition for ‘smartphone’ can hardly be found in the literature. Becker et al. [1] define smartphones as devices which: (a) “contain a mobile network operator smartcard with a connection to a mobile network”, i.e. a SIM or USIM card in GSM and UMTS systems, respectively, and, b) “have an operating system that can be extended with third-party software”. However, this definition appears to be rather broad. Also, its properties are valid for feature phones. For instance, the Motorola V3i¹ feature phone would be incorrectly classified as a smartphone, as it contains a mobile carrier SIM card and has a proprietary OS that can be extended by third party applications (specifically with MIDP 2.0 Java applications).

¹ http://www.motorola.com/mdirect/manuals/V3i_9504A480.pdf

The alternative definition of a smartphone, which is adopted here, is the following: *smartphone is a cell phone² with advanced capabilities, which executes an identifiable operating system allowing users to extend its functionality with third party applications that are available from an application repository.* According to this definition, smartphones must include sophisticated hardware with: a) advanced processing capabilities (e.g. modern CPUs, sensors), b) multiple and fast connectivity capabilities (e.g. Wi-Fi, HSDPA), and (optionally) c) adequately limited screen sizes. Furthermore, their OS must be clearly identifiable, e.g. Android, Blackberry, Windows Phone, Apple's iOS, etc. Finally, the OS must allow third party application installation from application repositories ('app markets'), e.g. Android Market, Blackberry App World, App Hub, App Store, etc.

2.1 Smartphone Assets

A smartphone is viewed herein as a small-scale information system, which incorporates various assets. Jeon et al. [12] identify as its assets: (a) *private information* (address book, calling history, location information, etc.), (b) *device* (resources, i.e. CPU, RAM, battery), and (c) *applications*. Another report identifies six assets: (a) *Personal data*, (b) *Corporate intellectual property*, (c) *Classified (governmental) information*, (d) *Financial assets*, (e) *Device and service availability and functionality*, and (f) *Personal and political reputation* [6]. Another taxonomy includes *Communication* (Voice communication, Messaging), *Data access* (E-mail, Web access, Bluetooth/IR), *Applications* (Maps & Navigation, Social networking, etc.), and *Device/Stored data* (Physical device, Offline applications/Utilities, etc.) [14]. An assessment of security in the case of the Android platform [21] analyses: (a) *private/confidential content* stored on the device, (b) *applications and services*, (c) *resources* (battery, communication, RAM, CPU), and (d) *hardware* (device, memory cards, battery, camera).

Our analysis makes use of four asset types: a) *Device*, b) *Connectivity*, c) *Data*, and d) *Applications*. The *Device* asset type includes the physical device and its resources (e.g. battery, RAM, CPU etc.), but *not* the *Data*. The latter appear to be more complex and are analysed in the next section. *Applications* are viewed only as user services.

Smartphones use four *Connectivity* channels, namely: a) *GSM services*, i.e. messaging (SMS, EMS, etc.) and voice calls, b) *PAN interface* (e.g. Bluetooth, IrDA, etc.), a free and ad-hoc short-range data channel, c) *WLANs* (e.g. Wi-Fi, WiMAX, etc.), a fast data channel, and d) *Cellular network*, which provides Internet connectivity at variable speeds, depending on the carrier technology (e.g. GPRS, HSDPA, etc.).

2.2 Data Taxonomy

Data are classified on the basis of two dimensions, i.e., *information type* and *source*. Table 1 associates the two dimensions. These associations will be used later as the basis for the data impact valuation.

² A *cell phone* is a device which: a) *is used primarily by its holder to access mobile network carrier services*, e.g. phone calls, Short Message Services (SMS), etc., and b) *contains a smartcard*, which is controlled by the network carrier (i.e. SIM or USIM card) and *incorporates a billing mechanism* for the used network carrier services.

Table 1. Smartphone data taxonomy

<i>Information</i>	<i>Type</i>	Personal	Business	Government	Financial	Authentic ation	Connecti- on/Service
Messaging		✓	✓	✓	✓	✓	
Device		✓	✓	✓	✓	✓	✓
USIM Card		some	some	some	some	✓	✓
Application		✓	✓	✓	✓	✓	✓
Use history & caching		✓	some	some	some		✓
Sensor		✓	✓	✓			
Input methods		✓	✓	✓	✓	✓	

Smartphone data hold various meanings. Their classification, according to the *information type* they may infer, led us to the following taxonomy:

- *Personal data* are directly related to an identified individual. They are considered private and should not be made public. Examples include the content of a user’s communication, images, videos, etc. Disclosure or unauthorized modification may result in embarrassment, reduction in self-esteem, or legal action.
- *Business data* (or corporate intellectual property) refer to data with commercial and economic corporate significance. These include marketing information, products under design, etc. Unintended disclosure of this data to the public or competitors may lead to strategic advantage loss, copyright breach, loss of goodwill, etc. Such data are usually likely to exist in a ‘personal’ smartphone, if it is (even occasionally) used for business purposes.
- *Government data* affect: (a) public order, (b) international relations, or (c) performance of public service organization(s). They differ from business data, because they hold national or international significance, as opposed to business value.
- *Financial data* refer to records of financial transactions, current financial holdings or position. Unauthorized modification, disclosure, or unavailability may lead to financial loss or contract breach (e.g., due to delays).
- *Authentication data* refer to user credentials, e.g. passwords, PINs, biometrics, etc. Their unauthorized access may lead to impact, such as financial loss, personal information disclosure, legal consequences, etc.
- *Connection/ Service data* refer to data, which are required for network connections. They include connection identifiers, such as Wi-Fi MACs, IMSI, or IMEI, as well as data regarding the connection itself, such as the Wi-Fi joined networks history.

During regular (e.g., daily) use, data are used or stored on various sources. The following taxonomy is based on another dimension, i.e., *data source* [15]:

- *Messaging Data* derive from: (a). mobile carrier messaging services i.e. Short Message Service (SMS), Enhanced Messaging Service (EMS), Multimedia Messaging Service (MMS), or (b). Instant and e-mail messages. They also include messaging logs, e.g. receiver, sender, delivery time and date, attachments, etc.
- *Device Data* are data that (a) are not related to any third party application, or (b) contain device and OS specific information. They may reside in internal (e.g. flash drive, flash memory) and removable (e.g. microSD cards) storage media. Some examples include images, contact list, Wi-Fi MAC address, device serial number, etc.
- *(U)SIM Card Data* reside either in a Universal Subscriber Identity Module (USIM) or Subscriber Identity Module (SIM) card. Typical examples are the International Mobile Subscriber Identity (IMSI)³ and the Mobile Subscriber Identification Number (MSIN)⁴. This source often contains SMS and contact list entries.
- *Application Data* include permanent or temporal data that are necessary for application execution. They may be stored as individual files, or constitute a local database, e.g. SQLite. A typical example is a flat dictionary text file.
- *Usage History Data* are used for logging purposes, such as: (a) call history, which contains incoming or outgoing phone call logs, (b) browsing history, i.e. temporary data created while the user browses local or remote files, (c) network history logs for wireless connections, e.g. Wi-Fi SSIDS, Bluetooth pairing, and (d) event logs, which are created by the OS for system monitoring and debugging.
- *Sensor Data* are created by dedicated hardware. Camera(s) and microphone(s) are two popular sensors. Other sensor hardware include: a) GPS sensor, b) accelerometer, c) gyroscope, d) magnetometer (i.e. digital compass), and e) proximity sensor. These are used to infer the exact device location, its orientation, the way the device is being moved, its heading direction, and the device distance from a surface, respectively. Sensor hardware, such as the light sensor and the temperature or pressure sensor, are present in some smartphones, measuring the device environment surroundings (context). Sensor data are mostly consumed on the fly and are not typically stored for later retrieval. Finally, they may be used as metadata (e.g. in geotagging where GPS data are embedded in photographs and videos).
- *User Input Data* include user gestures, hardware button presses, and keystrokes from a virtual or smartphone keyboard. All involve user interaction with the device. User input data are often consumed on the fly, or stored in a keyboard cache for performance reasons (e.g. improvement of spelling software).

3 Smartphone Risk Assessment

To assess smartphone risk, one should first assess the impact of its assets. Then, assets should be related to smartphone threat scenarios. Impact assessment for each asset is described in the sequel.

³ A unique number that identifies the subscriber to the network.

⁴ The 10-digit phone subscriber number.

3.1 Asset Impact

A key concept is, first, to involve the user with the initial impact valuation process. Then, the risk analyst should perform transparent associations and aggregations to calculate the overall risk.

Device. In typical risk assessment methods, physical assets are valued in terms of replacement or reconstruction costs, in a quantitative way. For a smartphone this refers to replacement or repair device cost, in the case of loss, theft, or damage. However, a smartphone contains, also, various information types, which need to be co-assessed in terms of impact.

Data. For information assets, a loss of confidentiality, integrity, or availability may be valued via several criteria [10], [14], i.e. personal information disclosure, legislation violation, contractual breach, commercial and economic interests, financial loss, public order, international relations, business policy and operations, loss of goodwill/reputation, personal safety, annoyance, etc. Due to the smartphone's multi-purpose nature, these impact types vary from purely personal ones, e.g. user annoyance, to typical information systems ones, e.g. commercial interests. This heterogeneity affects risk assessment. Adequate input from user is, thus, required, as clearly opposed to generic smartphone risk assessment [6], [21], which uses expert opinion.

In a 'personalized' (or 'itemized') risk assessment, the user is asked questions aiming to determine the existing data types (e.g., "*Do you store personal data in your smartphone?*", "*Do you use your smartphone for business purposes? If so, do you work for a governmental institution?*", or "*Do you use your smartphone for financial transactions?*", etc.).

In turn, for each identified data type, the user is invited to assess the impact of the following scenarios: "*Which are the worst consequences if your <data type> are unavailable?*", "*Which are the worst consequences if your <data type> are disclosed to the public?*" and "*Which are the worst consequences if your <data type> are modified or damaged (deliberately/accidentally)?*". The answers lead to impact calculation for each data type, namely the unavailability impact $Impact_{UA}(data_type)$, the disclosure impact $Impact_{DS}(data_type)$, and the modification impact $Impact_{MD}(data_type)$.

Our approach adopts the "worst-case scenario" principle, i.e. the *max* operator is used to calculate the total scenario impact. The answers must follow a qualitative assessment of the impact types mentioned above, evaluated by the user with a 5-item⁵ Likert scale (*very low, low, medium, high, very high*). For each impact criterion, a table needs to be produced, mapping each qualitative assessment to a comprehensive description. For example, for the 'personal information disclosure' criterion, the "*very low*" valuation may refer to "*minor distress to an individual*", as opposed to the "*very high*" one, which may refer to "*significant distress to a group of individuals or legal and regulatory breach*". Again, a map to quantitative values is required, because impact criteria cannot be considered equivalent. For instance, the "*very high*" valuation on 'personal information disclosure' must not be quantitatively valued equally to the "*very high*" valuation on 'personal safety'.

⁵ Any number of levels between 3 and 10 can be used [10].

Connectivity. Likewise, the user assesses the network services impact: “Which are the consequences if you cannot use the SMS service?”, “Which are the consequences if you cannot connect to a Wi-Fi network?”, “Which are the consequences if your Wi-Fi connection is been monitored?”, etc. The assessment should follow the same valuation tables and scales, as the data valuation ones do. The resulting valuations, i.e. $Impact_{UA}(channel)$, $Impact_{DS}(channel)$, $Impact_{MD}(channel)$, are used for risk assessment of threat scenarios which affect connectivity.

Applications. The same procedure can be used for user applications. Although this approach allows for a fine-grained impact valuation, it adds considerable complexity, as the applications may be numerous. It, also, assumes that a user has a clear perception of an application’s significance, e.g. by using the application for some time. A trade-off could be the valuation of applications that the user identifies as more important. The assessment per application should follow the same valuation tables.

Total impact valuation. Based on the above, the user has assessed the impact of various scenarios (loss of availability, confidentiality and integrity) for all four smartphone asset types. These values are combined and used to assess the risk of the threats scenarios. For instance, the data type impact values are inferred to their associated data sources (see Table 1). This means that if a user has identified ‘personal’ and ‘financial’ data types on her smartphone, then the disclosure impact for the data source ‘*Messaging*’ can be calculated, as follows:

$$Impact_{DS}(Messaging) = \max \{Impact_{DS}(personal\ data), Impact_{DS}(financial\ data)\}$$

Likewise, the overall smartphone impact is the *max* impact from all four assets, i.e., the device, data, applications, and connectivity.

3.2 Threat

Threat likelihood is assessed on the basis of: (a) experience and applicable statistics, (b) vulnerabilities, and (c) existing controls and how effectively they may reduce vulnerabilities [10]. In this section a smartphone threat list is presented, together with a discussion on how threat likelihood may be assessed.

Table 2 presents a threat list expanding similar lists that are available in the smartphone literature [1], [5-6], [11-12], [14], [19], [21]. Each threat is grouped in the appropriate attack vector dimension (i.e. an asset utilization may be misused to impair another one (e.g. application access rights may be misused to leak private data)). Each threat is associated with the security attribute that it impairs.

A particular application may be an asset that needs protection and, at the same time, an attack access vector to other assets. For instance, the availability of a social networking application may be considered as a significant asset by a user (*high impact*), while being the attack vector for privacy threats. Even if this application is benign (i.e. not leaking any private data for malicious purposes), its privilege to access private data may be misused by a malicious application performing a deputy attack [3], [7], [9]. Though, it might be the case that a malicious application masquerades as a benign application (e.g. game) luring users into downloading it and, thus, being the attack vector themselves. Thus, such a smartphone privilege is herein considered vulnerability.

The permission acceptance likelihood differs in smartphone platforms. It depends on authorization decisions, as delegated by the platform security model. These decisions differ significantly [16] from allowing users to make security-critical authorization decisions (e.g. Android’s community driven security model), up to placing functionality control barriers in applications that enter application repository (e.g. the ‘walled garden’ approach of Apple’s App Store).

Table 2. Smartphone threats

Dimension	Threat	C	I	A
Network Connectivity	T1 Spoofing	✓	✓	✓
	T2 Scanning	✓		
	T3 Denial of Service, Network congestion			✓
	T4 Spam, Advertisements			✓
	T5 Eavesdropping	✓		
	T6 Jamming			✓
Device	T7 Loss, theft, disposal or damage	✓	✓	✓
	T8 Cloning SIM card	✓	✓	
	T9 Technical failure of device		✓	✓
	T10 Unauthorized device (physical) access	✓	✓	✓
Operating System	T11 Unauthorized Access	✓	✓	✓
	T12 Offline tampering	✓	✓	✓
	T13 Crashing			✓
	T14 Misuse of Phone Identifiers	✓		
	T15 Electronic tracking/surveillance/exposure of physical location	✓		
Applications	T16 Resource abuse			✓
	T17 Sensitive Information Disclosure (SID), Spyware	✓		
	T18 Corrupting or modifying private content		✓	✓
	T19 Disabling applications or the device			✓
	T20 Client Side Injection/ Malware	✓	✓	✓
	T21 Direct billing		✓	
	T22 Phishing	✓	✓	

3.3 Risk

The triplet used for the risk assessment of threats, which are associated with specific permission access rights (i.e. threats T14-T19, T21, T22) is: (*asset, permission combination, threat*).

Asset refers to the asset targeted by the threat. *Permission combination* refers to the permissions for the dangerous functionality required by the *threat*. The *permission combination* is the vulnerability the threat exploits. In turn, *threat likelihood* is valued on the basis of: a) the likelihood of permission combination acceptance in the smartphone platform, b) the threat incident likelihood, i.e. statistics on threat incidents in the platform or previous incidents experienced by the user, and c) the

relevant security control existence (e.g. Use of Mobile Device Management [20]). Given that the user has assessed the asset impact, the impact may be combined with the likelihood of the threat and the permissions acceptance, so as to calculate risk by forming the triplet: (*asset impact, permission likelihood, threat likelihood*) => *Threat Risk*

Risk assessment is calculated on the basis of a risk matrix [10]. Risk can be mapped as *Low* (0-2), *Medium* (3-5), or *High* (6-8). Since each threat is associated with specific security attributes, the relevant asset impacts are taken into account. For instance, when assessing the *Risk* of threat *T17* ‘Sensitive Information Disclosure’ on the data source ‘Messaging’, the disclosure impact value *Impact_{DS}* (*Messaging*) is used as the *asset impact*, because the particular threat affects confidentiality.

Table 3. Risk matrix

Threat likelihood	Low			Medium			High		
Permission likelihood	L	M	H	L	M	H	L	M	H
0	0	1	2	1	2	3	2	3	4
1	1	2	3	2	3	4	3	4	5
Asset impact	2	2	3	4	3	4	5	4	5
3	3	4	5	4	5	6	5	6	7
4	4	5	6	5	6	7	6	7	8

For threats that cannot be associated with specific permissions, i.e. T1-T13, T20, such a triplet cannot be produced. In this case, threats are combined with specific assets, and their risk is calculated on the basis of asset impact and threat likelihood, where the likelihood is calculated based on threat incident statistics or previous incidents experienced by the user. This is done through a simple table (see Table 4).

Table 4. Simplified risk matrix

Threat likelihood	Low	Medium	High
0	L	L	M
Asset	1	L	M
impact	2	L	M
3	M	M	H
4	M	H	H

4 Case Study: Risk Assessment in Android

This section provides a demonstration of the proposed risk assessment method in the case of the Android platform. Android was selected because it is: a) popular smartphone platform holding the 52.5% of the smartphone market sales share in Q3 of 2011[8], b) open source and, hence, its security model details are publicly available, and c) well studied platform and statistics about its threats are available.

For the purpose of this case study a HTC Hero (Android version 2.1) owner is assumed, who holds a ‘high’ managerial position in the Pharmaceutical industry. The user identified two data types, i.e., personal data and business data. She provided data impact valuations as follows: $\text{Impact}_{\text{UA}}(\text{personal})=1$, $\text{Impact}_{\text{DS}}(\text{personal})=2$, $\text{Impact}_{\text{MD}}(\text{personal})=2$, $\text{Impact}_{\text{UA}}(\text{business})=2$, $\text{Impact}_{\text{DS}}(\text{business})=4$ and $\text{Impact}_{\text{MD}}(\text{business})=3$. She chose not to assess network services and applications. She also assessed the replacement cost of the device as *low*, i.e. $\text{Impact}(\text{device})=1$.

In addition, the user provided the following data: 1) the only smartphone security control enabled is the automatic password device lock, 2) she regularly discusses critical business issues over the carrier voice service, 3) she does not consider herself a technology or security savvy user, 4) no past security incident has ever come into her attention, 5) has noticed some delays in the device, 6) travels frequently in technology ‘underdeveloped’ country where fast 3G data connections are not available and, thus, her only way to connect to the internet is either through public free Wi-Fi hotspots, or expensive carrier network if one is not available, 7) has never updated the firmware, 8) she regularly installs applications while she is using public transport (trains, subway, etc). Finally, since Android’s security is based on the user security consciousness (i.e. the user decides permission authorization during installation time in an all or nothing way [16]), the likelihood of a permission combination acceptance is estimated using Android application research and studies [5], [7].

For readability and space-limitations reasons, the case study focuses on risk that is due to threats T5, T10, T11, T13, T14, T17, T18, and T21.

- *T5 Eavesdropping*. The user-identified the use of GSM voice services, in a carrier where UMTS is not supported. As a result, the possibility of abusing the discussion confidentiality is High [1] and the $\text{Impact}_{\text{DS}}(\text{GSM Service})=\max\{\text{Impact}_{\text{DS}}(\text{personal}), \text{Impact}_{\text{DS}}(\text{business})\}=4$. Risk is assessed as High (see Table 4).
- *T10 Unauthorized device (physical) access*. The user has the automatic password device lock enabled, therefore this threat likelihood is Low. As physical access to a device affects all security attributes and may cause significant damage to the hardware itself, the total impact of the asset ‘device’ is the max value of the replacement cost and the relevant impact valuations for the data it holds. Therefore, $\text{Total_Impact}(\text{device}) = \max\{\text{Impact}(\text{device}), \text{Impact}_{\text{DS}}(\text{personal}), \text{Impact}_{\text{DS}}(\text{business}), \text{Impact}_{\text{MD}}(\text{personal}), \text{Impact}_{\text{MD}}(\text{business}), \text{Impact}_{\text{UA}}(\text{personal}), \text{Impact}_{\text{UA}}(\text{business})\} = 4$. Therefore, the threat risk is Medium (see Table 4).
- *T11 Unauthorized Access*. The user is running an Android version that suffers by known security vulnerabilities. The vulnerabilities have been identified and patched by the device vendor without the user applying them. In addition, publicly available and stable (i.e. confirmed) source code exists, which can exploit the vulnerabilities⁶. The source code can be used by an attacker, so as to gain unauthorised access to the device with administrator privileges. Thus, the threat likelihood is High. Since T11 may affect all security attributes but not the hardware itself, the $\text{Impact}_{\text{DS,MD,UA}}(\text{Device})$ is the max impact valuation of the data it holds, i.e. it equals with 4. As a result, the threat risk is High.

⁶ <http://www.exploit-db.com/exploits/15548/>

- *T13 Crashing*. The user is running a buggy version of Android 2.1 that affects the device performance. An official fix (patch) for this vulnerability is available from the device vendor, thus, the threat probability is High. T13 affects the device's availability. The unavailability impact of the asset 'device' is the max value of the relevant impact valuations for the data it holds. Therefore, $\text{Impact}_{\text{UA}}(\text{device}) = \max\{\text{Impact}_{\text{UA}}(\text{personal}), \text{Impact}_{\text{UA}}(\text{business})\} = 2$ and the threat risk is High (Table 4).
- *T14 Misuse of Phone Identifiers*: This threat is associated with the triplet $T1 = \langle \text{USIM Data}, \text{Open Network Sockets} + \text{Access Phone State}, T9 \rangle$. The likelihood of T14 is Low (~20%) [5]. The combination likelihood is, also, Low ($\leq 35\%$) [7]. As a result, the following risk triplet is formed: $\langle \text{Impact}_{\text{DS}}(\text{USIM Data}), \text{Low}, \text{Low} \rangle$. Since the $\text{Impact}_{\text{DS}}(\text{USIM Data}) = 4$ (i.e. max disclosure impact of associated data types - personal and business), the threat risk calculated from the triplet $\langle 4, \text{Low}, \text{Low} \rangle$, and, hence, it is 4 (Medium) (Table 3).
- *T17 Sensitive Information Disclosure (SID), Spyware*. Herein the call logs disclosure threat is examined. This is associated with the triplet: $\langle \text{UsageHistory}, \text{read the user's contacts data} + \text{Open Network Socket}, T17_Call_Logs \rangle$. The permission combination, according to [7], is Low ($\leq 16\%$), while statistics about the threat likelihood are not available. Thus, the risk triplet is: $\langle \text{Impact}_{\text{DS}}(\text{UsageHistory}), \text{Low}, N/A \rangle$, i.e. $\langle 4, \text{Low}, N/A \rangle$. As a result the risk varies from Medium to High.
- *T18 Corrupting or modifying private content*. The removable storage corruption by junk file addition is herein examined. The permission involved in this threat is the 'Write to External Storage'. The threat triplet is: $\langle \text{Device}, \text{Write to External Storage}, T18 \text{ Storage} \rangle$. Relevant studies [7] have revealed the combination likelihood to Medium ($\leq 50\%$), but additional data about the threat likelihood were unavailable at the time of publication. The threat triplet is: $\langle \text{Impact}_{\text{UA}}(\text{Device}), \text{Medium}, N/A \rangle$. Thus, the threat risk is 3-5 (Medium).
- *T20 Client Side Injection/ Malware*. The likelihood of this user downloading malware in her device is considered High, since: a) the user frequently installs applications in the device, b) user is not considered a security savvy one, and c) most smartphone malware are targeting Android (40% of mobile malware that was detected in Q3 of 2011[13]). As in T11, this threat may affect all security attributes but not the hardware itself, the $\text{Impact}_{\text{DS,MD,UA}}(\text{Device})$ is the max impact valuation of the data it holds, i.e. it equals with 4. Thus, the threat risk is High.
- *T21 Direct billing*. Since the user frequently makes use of the carrier data connectivity, in order to connect to the Internet, malicious applications may abuse the Internet permission to incur direct costs to the user. The malicious application needs - apart from the permission to open network socket - access to the networking state, i.e. if the carrier data are being used. Hence, the involved triplet is: $\langle \text{USIMCard}, \text{Use Network Socket} + \text{Access Information about Networks}, T21\text{CarrierData} \rangle$. The permission combination likelihood is High ($\leq 86\%$)⁷ [7], and the threat likelihood is also High. Thus, the threat triplet is: $\langle \text{Impact}_{\text{MD}}(\text{USIMCard}), \text{High}, \text{High} \rangle$. As the $\text{Impact}_{\text{MD}}(\text{USIM Data}) = 3$ (i.e. max

⁷ Access to network state is granted to all Android applications without user intervention.

modification impact of associated data types - personal and business), threat risk is calculated from the triplet $\langle 3, \text{High}, \text{High} \rangle$ and thus it is 7 (High).

Table 5 summarizes the risk assessment of the threats included in the case study.

Table 5. Risk assessment results summary

Threat	Asset Impact	Permission Likelihood	Threat Likelihood	Risk
T5	Impact _{DS} (GSM Service) = 4	N/A	High	High
T10	Total_Impact(Device) = 4	N/A	Low	Medium
T11	Impact _{DS,MD,UA} (Device) = 4	N/A	High	High
T13	Impact _{UA} (Device) = 2	N/A	High	High
T14	Impact _{DS} (USIM Data) = 4	Low	Low	Medium
T17	Impact _{DS} (UsageHistory) = 4	Low	N/A	Medium-High
T18	Impact _{UA} (Device) = 2	Medium	N/A	Medium
T20	Impact _{DS,MD,UA} (Device) = 4	N/A	High	High
T21	Impact _{MD} (USIM Data) = 3	High	High	High

5 Conclusions

We have presented a risk assessment method that is tailored for smartphones. The method is compatible with established guidelines on risk assessment [10]. Nonetheless, contrarily to traditional risk assessment methods, which treat smartphones as a single entity, this method provides a more fine-grained valuation by: (a) dividing the device into various (sub)assets, (b) assessing smartphone-specific threats, and (c) taking into account the characteristics of a smartphone security model.

User input for (sub)asset impact is based on a two-dimensional data taxonomy. The data analysis takes place transparently to the user and it leads to a ‘personalised’ risk assessment, as opposed to other smartphone-oriented methods, which use mainly expert opinions [6], [21]. The level of input detail varies according to user skill [14] - this may indeed affect the quality of results - but our approach requires minimum input, i.e. the data impact valuation. The method could be potentially used in order to extend an information risk assessment method to include smartphone-specific threats, as its theoretical basis is compatible with best practices, i.e. ISO27005 [10].

This is used in combination with a threat list to conduct risk assessment. The list was compiled by extending existing threat lists of smartphone literature. For threat assessment, risk triplets are introduced, which makes the approach novel. They use application permissions as the attack vector, associating assets to threats and permission combinations. Risk is, then, assessed as a combination of asset impact and threat likelihood. Finally, a demonstration of the proposed assessment method is provided, via a case study based on the Android platform.

It should be noted that generic conclusions for specific threats cannot be drawn by ‘high’ risk valuations of a hypothetical, single case. Risk is highly affected by the

valuation of impact, which is a parameter that varies among different users. However, our method can also be applied in other smartphone platforms with permission-based security models (e.g. Symbian, Windows Phone, etc.), as well as in other platforms (e.g. iOS, BlackBerry, etc.) with some minor adjustments. For instance, risk triplets can be created by examining API library combinations of applications that exist in application repositories.

Future research may aim for an extended review of threats and vulnerabilities, along with an analytical dictionary of permission combinations. This will allow a more detailed threat assessment, based on past incidents or statistics, the presence of vulnerabilities or controls, analyzed on a per threat basis. We may also provide explanatory impact valuation tables and relevant questionnaires, which are appropriate for smartphone users.

Acknowledgements. This research has been co-funded by the European Union (ESF) and Greek national funds, through the Operational Program “Education and Lifelong Learning” of the National Strategic Reference Framework (Program HERACLEITUS II: Investing in knowledge society through the European Social Fund).

References

1. Becher, M., Freiling, F., Hoffmann, J., Holz, T., Uellenbeck, S., Wolf, C.: Mobile Security Catching Up? Revealing the Nuts and Bolts of the Security of Mobile Devices. In: Proc. of the 2011 IEEE Symposium on Security and Privacy (SP 2011), pp. 96–111. IEEE Computer Society, USA (2011)
2. Caldwell, T.: Smart security. *Network Security* 2011(4), 5–9 (2011)
3. Dietz, M., Shekhar, S., Pisetsky, Y., Shu, A., Wallach, D.: Quire: lightweight provenance for smart phone operating systems. In: 20th USENIX Security Symposium, USA (2011)
4. Dlamini, M., Eloff, J., Eloff, M.: Information security: The moving target. *Computers & Security* 28(3-4), 189–198 (2009)
5. Enck, W., Ocateau, D., McDaniel, P., Chaudhuri, S.: A study of android application security. In: Proc. of the 20th USENIX Conference on Security (SEC 2011), USA, p. 21 (2011)
6. Hogben, G., Dekker, M.: Smartphones: Information security risks, opportunities and recommendations for users. Technical Report, ENISA (2010)
7. Felt, A.P., Greenwood, K., Wagner, D.: The effectiveness of application permissions. In: 2nd USENIX Conference on Web Application Development (WebApps 2011), pp. 75–86 (2011)
8. Gartner: Market Share: Mobile Communication Devices by Region and Country, 3Q11. Technical Report (2011)
9. Grace, M., Zhou, Y., Wang, Z., Jiang, X.: Systematic Detection of Capability Leaks in Stock Android Smartphones. In: Proc. of the 19th Network and Distributed System Security Symposium, NDSS 2012 (2012)
10. ISO/IEC: Information technology – Security techniques - Information security risk management. ISO/IEC 27005:2008, 1st edn. (2008)
11. Jansen, W., Scarfone, K.: Guidelines on Cell Phone and PDA Security. Recommendations of the National Institute of Standards and Technology, Special Publication 800-124 (2008)

12. Jeon, W., Kim, J., Lee, Y., Won, D.: A Practical Analysis of Smartphone Security. In: Smith, M.J., Salvendy, G. (eds.) HCII 2011, Part I. LNCS, vol. 6771, pp. 311–320. Springer, Heidelberg (2011)
13. Kaspersky Labs: IT Threat Evolution: Q3 (2011), http://www.securelist.com/en/analysis/204792201/IT_Threat_Evolution_Q3_2011
14. Ledermüller, T., Clarke, N.L.: Risk Assessment for Mobile Devices. In: Furnell, S., Lambrinouidakis, C., Pernul, G. (eds.) TrustBus 2011. LNCS, vol. 6863, pp. 210–221. Springer, Heidelberg (2011)
15. Mylonas, A.: Smartphone spying tools. MSc Thesis, Royal Holloway, University of London (2008)
16. Mylonas, A., Dritsas, S., Tsoumas, B., Gritzalis, D.: Smartphone Security Evaluation: The Malware Attack Case. In: Samarati, P., Lopez, J. (eds.) International Conference on Security and Cryptography (SECRYPT 2011), pp. 25–36. SciTePress (2011)
17. Nachenberg, C.: A Window Into Mobile Device Security. Technical Report, Symantec Security Response (2011)
18. Oppliger, R.: Security and Privacy in an Online World. *Computer* 44(9), 21–22 (2011)
19. OWASP: Top 10 Mobile Risks, http://www.owasp.org/index.php/WASP_Mobile_Security_Project
20. Redman, P.: John Girard, L.: Magic quadrant for mobile device management software. Technical Report G00211101, Gartner (2011)
21. Shabtai, A., Fledel, Y., Kanonov, U., Elovici, Y., Dolev, S., Glezer, C.: Google Android: A Comprehensive Security Assessment. *IEEE Security and Privacy* 8(2), 35–44 (2010)

Empirical Benefits of Training to Phishing Susceptibility

Ronald Dodge, Kathryn Coronges, and Ericka Rovira

United States Military Academy, West Point, New York, USA
{ronald.dodge,kate.coronges,ericka.rovira}@usma.edu

Abstract. Social engineering continues to be the most worrisome vulnerability to organizational networks, data, and services. The most successful form of social engineering is the practice of phishing. In the last several years, a multitude of phishing variations have been defined including pharming, spear phishing, and whaling. While each has a specific reason for its success, they all rely on a user failing to exercise due diligence and responsibility. In this paper, we report on a recent phishing experiments where the effects of training were evaluated as well as gathering demographic data to explore the susceptibility of given groups.

1 Introduction

The use of technology has become pervasive in our work and home environments. While security technologies have continued to progress at a pace to parry the onslaught of technical attacks, users continue to expose our networks, data, and services to avoidable security risks. We have achieved little improvement in the awareness of users to detect and avoid one of the most prevalent forms of social engineering – phishing. [1,2,3] Phishing uses a variety of social engineering techniques to entice the user into doing something that they would not do if they understood the ramifications of the action. Regardless of how an organization employs encryption or two factor/token based authentication, a user can subvert all of these controls by simply opening an email.

A not for profit group, the Anti-Phishing Working Group [4], reported in the 2011 phishing report that phishing continues to be a pervasive threat. Of particular note was an increased sophistication by phishers using a technique called spear phishing. In spear phishing, elaborate, targeted emails that use either organizational or personal details are used to entice a user to click a link or open an attachment. The Anti-Phishing Working Group report further that spear phishing was the most concerning threat of 2011. The spear phishing campaigns were used to target organizations where security training might increase the awareness of users. Specific organizations included security companies, defense contractors, and financial institutions. These spear phishing attacks were a key component of the Advance Persistent Threat (APT). In 2007 Jagatic, et al. showed that a user is 4.5 times more likely to fall victim and follow the instructions in a phishing email if it is spoofed to come from someone they recognize. [5]

2 Efforts to Assess the Phishing Threat

Many studies and research projects have sought to explore detection and mitigation options for phishing emails. Some approaches employ advanced semantic processing and keyword scanning to block unwanted emails other techniques use visual clues in the email client to warn users of suspect emails. In most all studies, regardless of the technologies deployed, if the email is tempting enough, a user will fall victim. [6, 7]

The authors have conducted many studies in the effectiveness of training and education in stemming a person’s susceptibility to phishing. In early studies, the focus of the studies focused on the simple effectiveness of phishing exercises. [8] In these early results, the authors constructed an infrastructure to execute a phishing exercise and achieved results indicating an average 40% susceptibility to phishing. These results were validate in a follow-on study [9] and further showed that exercises repeated over a short duration increased awareness and susceptibility reduced to under 5%. The most recent effort analyzed the social network impact on susceptibility [10]. In these results, there was an identifiable clustering of victims by organization; if an organizations leadership was vulnerable – it was likely that so to would the personnel in that organization. In Figure 1, the larger circles indicate supervisors and red circles indicate falling victim to the phishing exercise.

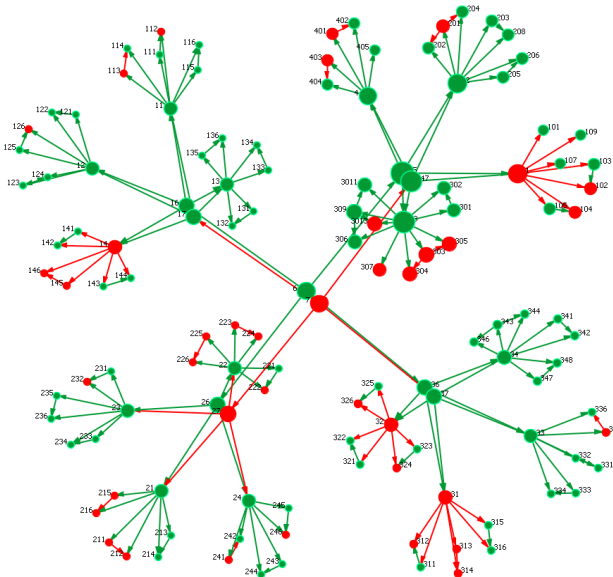


Fig. 1. Supervisorial Network: Phish Victims Sized by Centrality

3 Training Users: Effective or a Waste of Time?

Many large organizations require training to ensure employees are aware of the risks exposed to the organization by users falling victim to a phishing email. This training usually comes at a fiscal and manpower cost to the organization. Phishing exercises have emerged in the past few years to be an effective mechanism to provide a training capability that provides lower investment by the organization. However with the popularity of these exercises – is training still required? The research documented in this paper sought to explore that question and provide empirical evidence to answer.

3.1 Methodology

Eight hundred ninety-two (892) subjects selected at random at the <removed for review> participated in this study. Ages of the subjects ranged from 18-26. The phishing emails sent to all subjects included an embedded URL that when clicked takes users to a web site where they are asked to enter sensitive information (their network credentials). In all cases the email ‘bait’ leveraged knowledge of the users organization (spear phishing), where some sort of free or discounted service appealing to the target population was used. Emails were sent from a third party service provider outside the institution’s boundary. The service selected was phishme.com [11]. Three emails were sent - one in November, another 10 days later in December, and another 6 weeks later in January. Prior to the study, all participants took the required institutional phishing awareness training (in September).

To support the hypothesis of the experiment, the population was broken down into three notification conditions. Notification condition was randomized by organizational group. Each group was made up of three organizational units. There were 287 participants in the group one, 298 in group two, and 307 in group three. The three notification conditions were:

Group 1 (No Notification): participants received the phishing email, however after the user entered data into the website and clicked submit, the page returned a server error and no additional information was provided to the user.

Group 2 (Notification): participants received the phishing email, after the user entered data into the website and clicked submit, the page returned a notice that they fell victim to a phishing attack and provided details as to what the user should have identified in the email.

Group 3 (Training): received the phishing email, after the user entered data into the website and clicked submit, the page returned a notice that they fell victim to a phishing attack and directed the user to take the institutions phishing awareness training.

The emails themselves were constructed to provide several clues designed to alert end users. By default, emails are displayed in plain text mode; users have the ability to choose to view in HTML after the email has been displayed in plain text. Figures 2 and 3 show both presentations of the email.

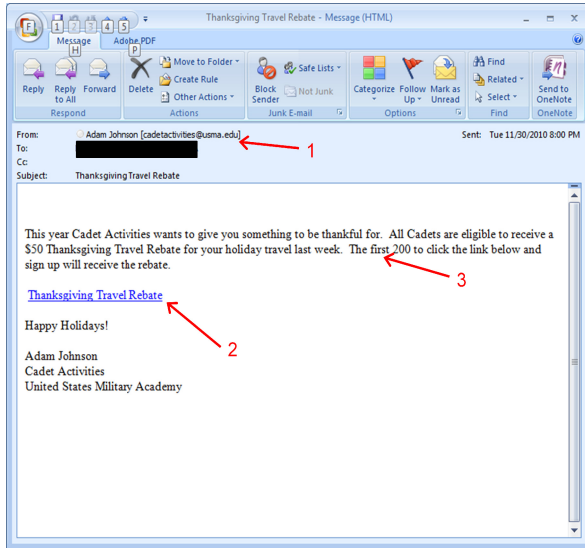


Fig. 2. HTML Email View

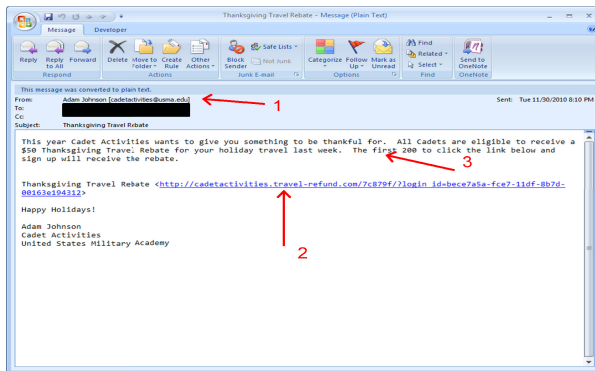


Fig. 3. Plain Text Email View

The following details are items that are part of the annual training our users receive and should have alerted them that the email was likely not legitimate.

1. Source email address (Adam Johnson [user@institution]): The email is from a person; however the source address is an institutional address. Further the user does not exist in our global list (accessible by all authorized users).
2. The URL in the email: In figure 2, the URL is shown in the middle using html formatting for email. This is not how email is delivered by default to our users. Instead the 'presentation text' is listed as well as the actual URL string, which is how it is first displayed to users (as shown in Figure 3).

3. The email urgency: While not always a valid sign of phishing email, when an urgent email is received from a source outside the organization, it is highly suspect.
4. Finally, if they looking in the full mail headers (which is an option if they are concerned about the validity of the email); they would have seen that the originating email server is highly suspect. This is shown below using bolded and underline text.

```
Received: from localhost.localdomain (local-
host.localdomain [127.0.0.1]) by
mail.phishme.managedmachine.com (Postfix) with ESMTP id
5DC6B10A43 for
< user@institution >; Wed, 1 Dec 2010 01:05:14 +0000
(UTC)
Date: Wed, 1 Dec 2010 01:05:11 +0000
From:AdamJohnson < user@institution >
To: < user@institution >
Subject: Thanksgiving Travel Rebate
MIME-Version: 1.0
Content-Type: text/html; charset="utf-8"
X-Priority: 3
X-Pmsid:775892d4-fce0-11df-97b7-0163e4638cc
Message-ID:
<20101201010514.5DC6B10A43@mail.phishme.managedmachine.co
m>
```

Fig. 4. Full Mail Headers

3.2 Task Procedures and Experimental Design

The hypothesis posited is that the completion of mandatory training after falling victim to a phish would have significant impact on a student's future susceptibility. In addition, the group that simply received notification when they fell victim to the attack was expected reduce their susceptibility more so than the group that received no notification at all. Thus, group 3 (those that received training after falling victim to the phishing attack) was expected to improve their phishing vigilance the most, followed by group 2 (the notification group), with the no notification group improving the least.

Phishing "susceptibility" was operationalized as a binary variable indicating whether the participant clicked on a link in the phishing email (indicating a failure to detect the phish) or if they did not click on the embedded link in the phishing email (indicating either successful recognition of the email as a phishing attack, or that the participant never read the email at all).

A single factor between subjects design with three levels (training: none, notification only, training) was implemented to investigate susceptibility to phishing exercises. The first group of 287 participants received the phishing email, however after the user entered data into the website and clicked submit, the page returned a server error and no additional information was provided to the user. The feedback only group consisted of 298 participants that received the phishing email, after the user entered data into the website and clicked submit, the page returned a notice that they fell victim to a phishing attack and provided details as to what the user should have identified in the email. The training group (307 participants) received the phishing email, after the user entered data into the website and clicked submit, the page returned a notice that they fell victim to a phishing attack and directed the user to take the institutions phishing awareness training.

3.3 Results

To Train or Not to Train

The results indicate that over very short periods of time (10 days), there is no significant difference in susceptibility based on training. However, over longer periods of time (63 days), training does contribute significantly to the reduction in susceptibility.

A 3x3 (training x phishing attempts) mixed factorial ANOVA was used to test the effects of training on phishing failures, and how those effects endured over time. Results indicated significant main effects of training ($F(2,874) = 15.78, p < .01$) and phishing attempts ($F(2,1748) = 223.70$). A significant interaction between type of training and phishing attempts was also found, $F(4,1748) = 5.91, p < .01$.

To investigate the interaction, three tests of simple effects examined whether the training had a significant effect on each of the phishing attempts. A one-way ANOVA showed a significant difference between the training groups at phishing attempt one, $F(2,874) = 6.08, p < .01$. Failure rates were 56%, 46%, and 42%, respectively. However, a one-way ANOVA showed no difference between the three groups at phishing attempt two, $F(2,874) = .634, p > .05$. Lastly, a one-way ANOVA showed a significant effect of training at phishing attempt three, $F(2,874) = 18.32, p < .01$. Failure rate was the least for the group that received training (24.5%), but increased for the group that only received feedback (32.08%) and was the highest for the group that received neither feedback nor training group (47.5%). Figure 5 shows that while there was some variability between the groups at phish one, at phish two there was no difference between the groups, and at phish three training reduces phishing susceptibility beyond feedback alone.

Failure rate for the second phish was drastically lower than for both the first and third phish attack. The reduction is most likely due to the timing of the phish: phish two was sent out only two weeks after the initial training. This was due to organizational events and unfortunately skewed the study findings. Phish three was sent out 2 months after phish 2.

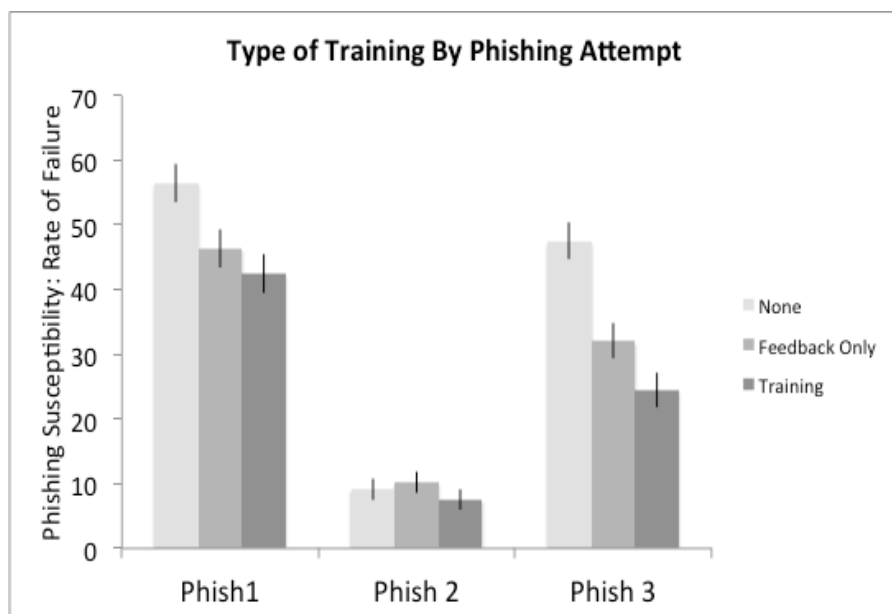


Fig. 5. Phishing Rate of Failure – Training Comparison

4 Conclusions and Recommendations

The outcomes of the experiments documented in this paper indicate several results that can be used to contribute to effective mitigation of phishing susceptibility. The major result shows that while the phishing exercise alone does lead to an increased level of awareness when the participant is notified that they fell victim, phishing training still provides a better return. As with all security program components, the business case for the increased level of awareness must be determined by the organization. Specifically, the organization has to balance the importance of reducing susceptibility of security threats with the increased time and organizational efforts involved with providing security training resources, as well as the additional efforts to make that training mandatory.

There are several types of training that are available that follow different pedagogical approaches. In future work, the authors will further validate the results in this report, while exploring the effectiveness of different training programs in reducing phishing susceptibility. In addition, future research will seek to identify broad demographic characteristics within the user population that may lead to a higher general susceptibility. This knowledge could help organization develop targeted training or increased awareness exercises where they would produce the highest payoff.

References

1. Downs, J., Holbrook, M., Lorrie, C.: Decision Strategies and Susceptibility to Phishing. In: Symposium on Usable Privacy and Security (2006)
2. Hicks, D.: Phishing and Pharming: Helping Consumers Avoid Internet Fraud. *Communities and Banking*, 29–31 (2005)
3. Stajano, F., Wilson, P.: Understanding scam victims: Seven principles for systems security. *Commun. ACM* 54(3), 70–75 (2011)
4. The Anit Phishing Working Group 2011 Annual Report (accessed January 13, 2012), http://www.antiphishing.org/reports/apwg_trends_report_h1_2011.pdf
5. Jagatic, T.N., Johnson, N.A., Jakobsson, M., Menczer, F.: Social phishing. *Commun. ACM* 50(10), 94–100 (2007)
6. Markoff, J.: Larger prey are targets of phishing. *New York Times* (April 16, 2008), <http://www.nytimes.com/2008/04/16/technology/16whale.html>
7. Hong, J.: Why have there been so many security breaches recently? *Blog@CACM* (April 27, 2011), <http://cacm.acm.org/blogs/blog-cacm/107800-why-have-there-been-so-many-security-breachesrecently/fulltext>
8. Dodge, R., Ferguson, A.: Using Phishing for User Email Security Awareness. In: Proceedings of the 21st IFIP International Information Security Conference (May 2006)
9. Dodge, R., Rovira, E., Radwick, Z., Shevchik, J.: Phishing Awareness Exercises. In: Proceedings of the 15th Colloquium for Information Systems Security Education, June 13-15, pp. 120–125 (2011)
10. Coronges, K., Dodge, R., Mukina, C., Rovira, E., Radwick, Z., Shevchik, J.: The Influences of Social Networks on Phishing Vulnerability. In: 2012 45th Hawaii International Conference on System Sciences, January 4-7, pp. 2366–2773 (2012)
11. <http://www.pishme.com> (accessed Decemeber 15, 2011)

Multi-modal Behavioural Biometric Authentication for Mobile Devices

Hataichanok Saevanee¹, Nathan L. Clarke^{1,2}, and Steven M. Furnell^{1,2}

¹ Centre for Security, Communications and Network Research, University of Plymouth,
Plymouth, United Kingdom

² School of Computer and Information Science, Edith Cowan University,
Perth, Western Australia
info@cscan.org

Abstract. The potential advantages of behavioural biometrics are that they can be utilised in a transparent (non-intrusive) and continuous authentication system. However, individual biometric techniques are not suited to all users and scenarios. One way to increase the reliability of transparent and continuous authentication systems is create a multi-modal behavioural biometric authentication system. This research investigated three behavioural biometric techniques based on SMS texting activities and messages, looking to apply these techniques as a multi-modal biometric authentication method for mobile devices. The results showed that behaviour profiling, keystroke dynamics and linguistic profiling can be used to discriminate users with overall error rates 20%, 20% and 22% respectively. To study the feasibility of multi-modal behaviour biometric authentication system, matching-level fusion methods were applied. Two fusion methods were utilised: simple sum and weight average. The results showed clearly that matching-level fusion can improve the classification performance with an overall EER 8%.

Keywords: Behavioural Biometrics, Authentication, Mobile Devices, Behavioural Profiling, Keystroke Dynamics, Linguistic Profiling.

1 Introduction

Mobile devices, such as cellular phones and Personal Digital Assistants (PDAs) are rapidly evolving technologies capable of providing many services through a wide range of applications over multiple networks such as the Internet (e.g. e-mail's and online banking), entertainment (e.g. photos and video games) and the sharing of data (via Bluetooth, laptop/computer). The plethora of functionalities offered by mobile devices enables users to store increasing amounts of wide ranging types of information from business to personal and sensitive data. With this in mind, previous research [1] highlights mobile users concerns of their devices being lost or stolen.

Many authentication mechanisms have been developed for mobile devices with the aim of providing a greater level of security for the end user. Biometric authentication is commonly acknowledged as a reliable solution which provides enhanced authentication over the traditional password ("something you know") and token

(“something you have”) approaches. Biometric characteristics are uniquely individual (“something you are”), non-transferable to others, impossible to forget or lose, difficult to reproduce, usable with or without the knowledge/consent of the individual and difficult to change or hide. However, current approaches are still focused upon point-of-entry authentication (e.g. PIN/passwords, fingerprint), which has a number of weaknesses. In the case that a user chooses not to use authentication in the first place or once the identity of the user has been verified at login, the mobile device is typically accessible to the user until they specifically exit the system. This can lead to a high risk environment in which an imposter targets a post authenticated session.

To increase the level of authentication beyond the standard point-of-entry technique, Clarke and Furnell [2] proposed using a combination of secret based knowledge and behavioural biometric techniques to provide transparent, non-intrusive continuous authentication. To this end, research suggests that no single biometric approach is ideally suited to all scenarios and several studies show that multi-modal biometric approaches are superior to one single biometric approach [3-6]

The popularity of the Short Messaging Service (SMS) is one of the most widely recognised and embraced functionalities of mobile communications with over 6.1 trillion messages sent in 2010; close to 200,000 messages sent every second [7]. This provides a unique opportunity to authenticate and discriminate between users based on their individual linguistic morphology.

This paper investigates three individual behavioural biometric techniques: behavioural profiling, keystroke dynamics and linguistic profiling. The performance of each of the aforementioned techniques is discussed together with the development of a multi-modal behavioural biometric approach in which the three individual techniques are combined.

Section 2 provides an overview of biometric authentication. Section 3 describes the methodology and Section 4 shows the results. Section 5 discusses the implications of the results. Finally, section 6 presents the conclusions and recommendations for future work.

2 An Overview of Biometric Authentication

The International Biometrics Group (IBG) defines biometrics simply as “the automated use of physiological or behavioural characteristics to determine or verify identity” [8]. Physiological biometrics perform authentication based on bodily characteristics such as their fingerprint or their face. By contrast, behavioural biometrics perform authentication based on the way people do things, such as their typing rhythm, their voice or their signature. Physical features are likely to stay more constant over time and under different conditions, and tend to be more unified within a large population [9]. Physiological biometrics therefore tends to be used for identification-based system because they are more trustable approaches. However, some behavioural biometrics have very good accuracy for verification but the identification accuracy of most behavioural biometrics is considerably lower as the number of users in the database becomes larger [10]. This is because users act differently depending on mood, illness, stress, previous events, environment, to name a few. For this reason, behavioural biometrics tends to be only used for authentication-based systems.

Behavioural biometrics provides a number of advantages; they can be collected without the knowledge of the user (non-intrusive) and continuously. Collection of behavioural data often does not require any special hardware and is therefore more cost effective. Based upon a typical mobile device, considering biometric approaches that do not require additional hardware to enable collection, the following biometrics could be utilised; facial recognition, voice verification, keystroke dynamics, behavioural profiling, handwriting recognition and linguistic profiling. Of those that are of interest in this paper: keystroke feature information can be captured through the keyboard interface when users type text messages or mobile phone numbers; linguistic profiling can analyse inputted text messages during SMS compilation; and behaviour profiling can capture users' behaviour continuously during their interaction with the mobile phone. It is hypothesised that each of the three behavioural biometric techniques described can be used to authenticate users. However, more interestingly, it is hypothesised that these three techniques combined together offer the opportunity to improve the underlying performance significantly.

2.1 Behaviour Profiling

Based on mobile devices, behaviour profiling aims to identify patterns of usage based upon characteristics of a user's behaviour. Research in mobile behavioural-based can be divided into two categories: network and host based mechanisms. The former will focus upon user calling and migration behaviour over the service provider network based upon the hypothesis that people have a predictable travelling pattern [11, 12]. A host-based mechanism is founded upon the hypothesis that mobile users utilise their applications differently in different time periods and at different locations. This approach would for example monitor user's calling features (e.g. the day of calling, start time of call, duration of call, dialled telephone number and the location), device usage and Bluetooth scanning [13,14], thereby providing a richer set of potential features than network-based approaches.

2.2 Keystroke Dynamics

Keystroke dynamics is a behavioural biometric which is based on each person's individual typing style on a keyboard. This behavioural biometric is not expected to be unique to each person but it offers sufficient discrimination information to permit identity authentication [15]. Considerable research has been undertaken on the keystroke dynamics and two main characteristics were identified: inter-key latency and hold time. The inter-key time is the duration or interval between two successive keys. Hold-time represents the duration between the press down and releasing of a single key. Several studies have concluded that keystroke dynamics provided valuable discriminative information [16, 17]. Since no additional hardware is required, this has been a favoured technique, with much research on the subject since the 1980's [18].

2.3 Linguistic Profiling

Linguistic profiling is a behavioural biometric that attempts to identify and discriminate between users based on linguistic morphology [19]. Linguistic profiling was used for determination of the language variety or genre of a text, or a classification for

document routing or information retrieval. Linguistic features such as specific aspects of the text often based upon frequency counts of functions words in linguistics and of content words in language engineering are used as a text profile, which can then be compared to average profiles for groups of texts. Considerable research has been undertaken on this technique and many types of linguistic features can be profiled such as lexical patterns, syntax, semantics, information content or item distribution throughout a string of text. Many researchers concluded that structural and stylometric features are valuable tools for author identification and verification [19-21].

The weakness of individual biometric approaches is that no single biometric is ideally suited to all scenarios. For example, linguistic profiling is only practical in scenarios with sufficient word messages. One of the key aims of transparent authentication is to provide a system that enables a variety of biometric techniques to be utilised in order to solve the problem. A multi-modal biometric approach offers the advantage of relaxing the assumption of universality, collectability, acceptability and integrity [22]. Multi-modal biometric approaches require a combination of biometric data. The combination or fusion method can occur effectively at any point within the biometric system: feature-level, matching-level or decision-level [23]. The feature-level method is achieved by combining the variety of feature vectors derived from different biometric techniques. Matching-level fusion takes the output of resulting matching classifications and combines the results (raw score) prior to presenting them to the decision process. At the end of the biometric system, decision-level fusion can occur when each individual biometric system has provided an independent decision. The decision results in a Boolean value which lacks the richness of information for fusion. Amongst the literature, match-level fusion has been shown to be the best performing of the fusion approaches [22, 23].

3 Experiment Procedure

In this paper, three behavioural biometric techniques were investigated: behavioural profiling, keystroke dynamics and linguistic profiling. After studying the performance of each single biometric, the final experiment built upon these findings through the fusion of the three individual techniques. For single modal biometric technique, the general biometric authentication system is illustrated in Fig. 1.



Fig. 1. A generic biometric system

3.1 Behaviour Profiling

The experiment based on behavioural profiling described in [13] has been used. For this study, a total of 30 participant's text messaging activities were recorded from a database provided by the MIT Reality Mining project [24]. As not all participants started or finished the experiment at the same time, each user has a varied number of

logs. This dataset contains 1470 logs and 274 unique texting numbers. The maximum number of logs for a user is 149 and the minimum number is 8 logs. For each text log, the following features: receiver's telephone number and location of texting were extracted to create user behavioural profiling. In the analytical process, neural network (Feed-Forward Multilayer Perception Neural Network) was used in the classification.

3.2 Keystroke Dynamics

The dataset of this experiment was provided by [16]. A total 30 participants were obtained with a total of 900 text messages. In this experiment, two main traditional characteristic features were utilised. To create the hold time dataset, the five letters ('e', 't', 'a', 'o' and 'n') were used in the classification. For the inter-key time dataset, the latency between five pair of letters: 't' – 'g', 'e' – 'p', 'e' – 'm', 'h' – 'd' and 'a' – 'm' were calculated. The database contains 3510 hold-time data, 1080 inter-key time data and outliers were removed (a standard procedure for keystroke analysis studies). Analyses were undertaken using Feed Forward Multilayer Perception Neural Network (FF-MLP) as it had demonstrated better performance in previous studies over other techniques [17].

3.3 Linguistic Profiling

In this experiment, the SMS dataset provided by previous research [25] was utilised. A total of 30 participants were required to send at least 15 messages to each other using a non-predictive text input method. The frequency distribution of abbreviations emotional words were used to create user profiles, including every possible type of feature. For each message, a total of 64 discriminating characteristics were extracted for example, average word length (number of characters), total number of sentences, total number of symbols etc. To create a user profile, t-test ranking measure were apply to rank input features according to its discriminative capability. According to the ranking list, features with p value less than 0.05 ($p < 0.05$) were selected for input vectors to reduce the unnecessary features in classification. Therefore, the number of linguistic features required for discrimination significantly differs between users. To analyse individual user's performance, a number of analyses were undertaken, using the Radial Basis function (RBF) neural network algorithm. Different network configurations were tested, looking for the optimum performance.

For each individual biometric technique, the dataset was divided into two groups: 171 data samples were used for the testing set and the rest were used for training. The pattern classification test was performed with one user acting as the valid user, while all others are acting as impostors. The Equal Error Rate (EER) was calculated to evaluate the system. The EER is the value where False Acceptance Rate (FAR) is crosses the False Rejection Rate (FRR), and is typically used as a comparative measure within the biometric industry [26].

3.4 Fusions

In light of the foregoing exploration, the multi-modal biometric study was conducted using a novel combination of behaviour profiling, keystroke dynamics and linguistic profiling. Of all the fusion approaches, matching-level fusion is the most widely used.

However, invariably the use of different classifiers results in different outputs being produced. The range of output result values might vary. In this study, to solve this problem, score normalisation was applied. The equation provides a mechanism to ensure all outputs are bounded between 0 and 1 is shown below:

$$\text{Score normalization}(X) = \frac{(x_i - \text{Min}(X))}{(\text{Max}(X) - \text{Min}(X))} \quad (1)$$

Where: x_i = the raw score of input i
 X = the set of raw score of individual biometric system
 $\text{Max}(X)$ = the maximum value of raw score vector
 $\text{Min}(X)$ = the minimum value of raw score vector

After applying score normalization into the raw score results, two fusion approaches were utilised: simple sum and weight average. To evaluate the experiment by simple sum technique, the raw scores of each individual biometric system were simply added and rescaled into $[0, 1]$ as below:

$$\text{Simple Sum} = \text{normalization}(\sum_{i=1}^N \sum_{j=1}^M X_{ij}) \quad (2)$$

Where: X_{ij} = the raw score of input i from biometric system j
 N = the total number of multi-modal biometric input score
 M = the total number of biometric system

For the average weight technique, Weights are assigned to the individual matchers based on their EER and the weights are inversely proportional to the corresponding errors; the weights for less EER are higher than those of high EER.

$$\text{Weight average} = \frac{\sum_{i=1}^N (1 - \text{EER}_i)}{\sum_{i=1}^N \text{EER}_i} \quad (3)$$

Where: i = the number of biometric system
 N = the total number of biometric system

4 Results

4.1 Behaviour Profiling

The results of using behaviour profiling to classify user is shown in Table1. The results illustrate that user text messaging application has significant potential to discriminate some users with the overall performance EER 20%. The best case individual user was achieving an EER 1%. Moreover, more than half of participants achieved EER less than 20%. However, the result of worst case individual performance showed fairly high EER 49%. This may be caused by the number of samples assigned to the training of the classification was too small (and a limitation of dataset). Interestingly, only two features: receiver's telephone number and location of texting can achieve the good performance. This is caused by these features having a good level of unique information.

Table 1. Best and worst case results for behavioural profiling

Classifier	EER	EER	EER
	Worst Case	Best Case	Average
SMS texting Profile	49%	1%	20%

4.2 Keystroke Dynamics

The main three biometric measurements were investigated: the hold-time, inter-key time and the combination of the hold time and the inter-key time using different network configurations. Table 2 shows the EER of all biometric measurement.

Table 2. Best and worst case results of individual and combination of keystroke characteristics

Classifier	EER	EER	EER
	Worst Case	Best Case	Average
Inter-Key Time	46%	7%	31%
Hold – Time	49%	5%	20%
Combination	50%	8%	28%

As illustrated in Table 2, considering the two traditionally keystroke characteristics, the results show that the hold-time gave the lowest average EER 20% with the best individual result EER 5%. These findings illustrate that using hold-time as the key to identify users is the most effective measurement. In contrary to the hold-time investigation, the inter-key characteristic provide fairly high EER 31%, there was the best case of user achieving an EER 7%, showing the ability to classify some users. Therefore, these two main traditional keystroke characteristics provide the valuable discriminative information to classify users. This study experience good performance that has been found in previous studies [16, 17]. In order to further assess the performance of keystroke dynamics, the combination of the hold time and the inter-key time was utilised. The results show that using combination can improve the overall EER of inter-key time by 3%. This is because increasing the component of features can increase the uniqueness of users.

4.3 Linguistic Profiling

The findings from this experiment are illustrated in Table 3. Using linguistic profiling to discriminate users showed positive results. The best individual result achieved the lowest EER 0.00 %. The overall EER also showed promising result with EER 22%. The positive results clearly illustrate that linguistic characteristics could be used successfully to discriminate some users. However, some users generated a fairly high EER. This may caused by selection of keywords and effective features process can result in classification performance.

Table 3. Best and worst case results for linguistic profiling

Classifier	EER	EER	EER
	Worst Case	Best Case	Average
Linguistic Profile	49%	0%	22 %

4.4 Fusion

To enhance the overall performance of multi-modal biometric, matching-level fusion of the aforementioned behavioural biometric techniques was investigated. Behavioural profiling results, hold-time results from keystroke dynamics and linguistic profiling were combined. In this experiment, two different fusion methods were utilised: simple sum and weight average. The results show below in Table 4.

Table 4. Best and worst case results of fusion experiments

Classifier	EER	EER	EER
	Worst Case	Best Case	Average
Fusion by sum	40%	0%	10%
Fusion by weight average	37%	0%	8%

As shown in Table 4, the results showed that both fusion methods can reduce the overall error rate thus increasing the overall performance. Fusion by weighted average produced better overall results with an EER of 8%, which improves upon the overall performance when compared against a single biometric 14% (based on the worst EER). Fusion method by sum is also efficient because the overall EER is 10%. In both studies of fusion experiments, the performance was improved for every participant. Therefore, using fusion method can improve the performance with low EER for every participant. Additionally, 90% of participants achieved EER less than 20%.

5 Discussion

The results have shown that behavioural biometric techniques based on user texting activities (behavioural profiling), user typing message rhythm (keystroke dynamics) and word messages (linguistic profiling) has significant potential to authenticate users. However, there are some users that have fairly high error rate for each technique. To improve the performance of classification, multi-modal biometric were investigated. In the fusion experiment, two fusion methods were applied: simple sum and weight average. The results demonstrate the utility of using multimodal biometric systems for achieving better matching performance than single modal system. The user achieved the optimum performance by utilising different fusion methods. This also indicates that the method chosen for fusion has a significant impact on the resulting performance. An additional advantage of fusion at this level is that a common fusion method can be utilised to create the reliable system and existing biometric systems do not need to be modified.

In biometric systems, implementers are forced to make a trade-off between usability and security. However not all techniques are available to fusion. For example, some biometric technique might have insufficient biometric data to classify. Therefore, a dynamic system needs to be developed. The framework requirements to drive the selection of tolerable error rates and in both single modal and multimodal biometric systems.

6 Conclusions

Behavioural biometric authentication tends to be used for authentication-based systems. This is because users act differently depending on mood, illness, stress, previous events, environment etc. The potential advantages of behavioural biometrics are that they can be utilised transparent and continuous authentication system. Additionally, the collection of behavioural data often does not require any special hardware and is so very cost effective. However, individual biometric techniques are not suited to all users. One way to increase the reliability of transparent and continuous authentication system is create a multi-modal behavioural biometric authentication system.

This research investigated three behavioural biometric techniques, behaviour profiling keystroke dynamics and linguistic profiling based on texting SMS activities and messages, looking to apply these techniques as a multi-modal biometric authentication method for mobile devices. The results showed that individual biometric technique can be used to discriminate users with low error rates. Moreover, the overall EER of multi-modal biometric also showed clearly can be successfully used to authenticate user.

The next step in this research is to further implement dynamic authentication system. The proposed framework also should be flexible and scalable in that it can adopt other biometric techniques. Moreover, the system can integrate new techniques or new biometric techniques without having to change the overall system design.

References

1. Edison,
<http://www.mformation.com/mformation-news/press-releases/mformation-sponsored-survey-reveals-mobile-users-worried-about-loss-and-mobile-fraud>
2. Clarke, N., Furnell, S.M.: Advanced user authentication for mobile devices. *Computer and Security* 26, 109–119 (2007)
3. Brunelli, R., Falavigna, D.: Personal Identification using Multiple Cues. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 955–966 (1995)
4. Kittler, J., Matas, J., Jonsson, K., Ramos Sanchez, M.U.: Combining Evidence in Personal Identity Verification Systems. *Pattern Recognition Letters* 18, 845–852 (1997)
5. Poh, N., Korczak, J.J.: Hybrid Biometric Person Authentication Using Face and Voice Features. In: Bigun, J., Smeraldi, F. (eds.) AVBPA 2001. LNCS, vol. 2091, pp. 348–353. Springer, Heidelberg (2001)
6. Ross, A., Jain, A.K., Qian, J.-Z.: Information Fusion in Biometrics. In: Bigun, J., Smeraldi, F. (eds.) AVBPA 2001. LNCS, vol. 2091, pp. 354–359. Springer, Heidelberg (2001)

7. International Telecommunication Union,
<http://www.itu.int/ITU-D/ict/material/FactsFigures2010.pdf>
8. International Biometric Group,
http://www.biometricgroup.com/reports/public/reports/best_biometric.html
9. Woodward, J.D., Orlans, N., Higgins, P.: *Identity Assurance in the Information Age*. McGraw-Hill/Osborne, Berkeley, California (2003)
10. Yamploskiy, R., Govindaraju, V.: Chapter 1 Taxonomy of Behavioural Biometrics. *Behavioural Biometrics for Human Identification: Intelligent Applications* (2010)
11. Gosset, P.: *ASPeCT: Fraud Detection Concepts: Final Report* (1998)
12. Hall, J., Barbeau, M., Kranakis, E.: Anomaly-based intrusion detection using mobility profiles of public transportation users. In: *The Proceeding of IEEE International Conference on Wireless And Mobile Computing, Networking And Communications*, vol. 2, pp. 17–24 (2005)
13. Li, F., Clarke, N., Papadaki, M., Dowland, P.: Behaviour profiling on mobile devices. In: *International Conference on Emerging Security Technologies*, UK, pp. 77–82 (2010)
14. Li, F., Clarke, N., Papadaki, M., Dowland, P.: Behaviour profiling for Transparent Authentication for Mobile Devices. In: *10th European Conference on Information Warfare and Security*, Estonia, pp. 307–314 (2011)
15. Obaidat, M.S., Sadom, B.: Verification of Computer Users Using Keystroke Dynamics. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 27, 261–269 (1997)
16. Clarke, N., Furnell, S.: Authenticating Mobile Phone Users Using Keystroke Analysis. *Information Security*, 1–4 (2006)
17. Karatzouni, S., Clarke, N.: Keystroke Analysis for thumb-based Keyboards on Mobile Devices. In: *22nd IFIP International Information Security Conference*, pp. 253–263. Springer, Heidelberg (2007)
18. Gaines, R., Lisowski, W., Press, S., Shapiro, N.: Authentication by keystroke timing: some preliminary results. *Rand Report R-2560-NSF*, Rand Corporation California (1980)
19. Halteren, H.: Linguistic Profiling for Author Recognition and Verification. In: *42nd Annual Meeting on Association for Computational Linguistics*, NJ, pp. 199–206 (2004)
20. Argamon, S., Saric, M., Stein, S.: Style Mining of Electronic Messages for Multiple Authorship Discrimination: First Results. In: *9th ACM SIGDD International Conference on Knowledge Discovery and Data Mining*, Washington (2003)
21. Goodman, R., Hahn, M., Marella, M., Ojar, C., Westcott, S.: The use of stylometry for email author identification: a feasibility study. In: *Student/Faculty Research Day*, CSIS, Pace University (2007)
22. Poh, N., Bengio, S., Korczak, J.: A multi-sample multi-source model for biometric authentication. In: *IEEE International Workshop on Neural Networks for Signal Processing*, pp.375-384 (2002)
23. Clarke, N.: *Transparent User Authentication*, Springer, p. 229 (2011)
24. Eagle, N., Pentland, A., Lazer, D.: Inferring Social Network Structure using Mobile Phone Data. In: *International Conference on Security & Management*, pp.207-212, Las Vegas (2006)
25. Saevanee, H., Clarke, N., Furnell, S.: SMS Linguistic Profiling Authentication on Mobile devices. In: *5th International Conference on Network and System Security*, pp. 224-229 (2011)
26. Ashbourne, J.: *Biometric, Advanced identity verification. The complete guide*. Springer (2000)

Analysis and Modeling of False Synchronizations in 3G-WLAN Integrated Networks

Christoforos Ntantogian¹, Christos Xenakis¹, and Ioannis Stavrakakis²

¹ Department of Digital Systems, University of Piraeus, Greece
{dadoyan, xenakis}@unipi.gr

² Department of Informatics and Telecommunications, University of Athens, Greece
ioannis@di.uoa.gr

Abstract. Recently, a set of problems have been solved analytically, in order to improve various aspects of the authentication procedure in mobile networks. In these problems, an analytical model is derived, which is used to draw guidelines on the selection of the appropriate value of various system parameters. We observe that all these problems can be solved using a generic modeling procedure that is based on Markov chains assuming exponential distributions. In this paper we apply this generic modeling procedure to the problem of false synchronizations in 3G-WLAN integrated networks. To this end, we first elaborate on false synchronization using a numerical example that facilitates the better understanding of the presented notions. Then, an analytical model based on a four dimensional Markov chain is developed, using the generic modeling procedure, whose accuracy is verified through simulations.

Keywords: Markov chains, analytical modeling, false synchronizations, 3G-WLAN networks, authentication.

1 Introduction

In third generation (3G) mobile networks, authentication is executed between the mobile station (MS) and the home network where MS is subscribed. The MS initiates the authentication procedure by sending an authentication request to the access network. If the latter has authentication credentials, called authentication vectors (AV), available for the specific MS, then the access network authenticates the MS. The authentication procedure of 3G networks includes a security mechanism, which ensures that each AV is used only once (freshness property). This protects both the MS and access network from what is known as replay attacks: that is, an attempt by an adversary to use a compromised previously used AV in order to authenticate itself either as a valid MS or as a valid access network. To detect such attacks, the MS and the access network try to keep track of previously used AVs using either counters or timestamps [1]. This paper considers the first case.

Although this protection mechanism defeats replay attacks, at the same time it creates negative side effects from a performance point of view. More specifically, in some circumstances an MS that changes frequently access networks, may receive AVs that have not been previously used, but the employed mechanism rejects them as

outdated. This phenomenon defined as false synchronization impose signaling overhead that: (i) increases significantly the authentication latency, especially in cases that the MS is located far away from its home network; (ii) increases the call blocking rate in cases that the MS has active real-time sessions (e.g., VoIP, videoconference); and (iii) overcharges the MS in cases that the access network and home network are located in different countries. Therefore, false synchronizations: (a) deteriorate the overall network performance, (b) lower the quality of service offered to MSs, and (c) increase the cost of the network use [4].

Apart from false synchronizations, recently a set of problems have been solved analytically, (such as [5], [6], [7], [8]) in order to improve various aspects of the authentication procedure in mobile networks. In all these problems, an analytical model is derived, which is used to draw guidelines on the selection of the appropriate value of various system parameters. We observe that all these problems can be solved using a generic modeling procedure that is based on Markov chains assuming exponential distributions. Specifically, this modeling procedure includes the following steps: 1) pinpoint the underlying Markov structure that models the system dynamics (i.e., identify the Markov chain dimension, states and its transitions); 2) in case of large or infinite Markov chains, apply a state space truncation to ensure that a steady state analysis can be performed; 3) develop the steady state equations; 4) solve the system of linear equations, using numerical or analytical methods, to derive steady state probabilities. After the last step, the key performance metrics of each problem can be easily derived to provide insights into the specific system.

In order to put the aforementioned modeling procedure into effect, we apply it to the problem of false synchronizations in 3G-WLAN integrated networks [4]. For this purpose, we first elaborate on false synchronization using a numerical example that facilitates the better understanding of the presented notions. Based on the modeling procedure, we develop an analytical solution using a four dimensional Markov chain that captures the system dynamics. We notice that the derived Markov chain cannot be directly used to perform steady state analysis. To this end, we perform a state space truncation of the Markov chain. Finally, we derive the steady state equations and solve a linear system of equations to derive the probability of a false synchronization.

2 Background

2.1 3G-WLAN Network Architecture

As shown in Fig. 1, the 3G-WLAN integrated network architecture consists of four individual parts [2]: (i) the MS, (ii) the UMTS radio access network (UTRAN), (iii) the WLAN, and (iv) the 3G core network. The MS comprises the user's device (e.g., laptop, PDA) and the universal subscriber identity module (USIM), which contains the user's subscriber information. UTRAN consists of Nodes B that provide wireless connections to MS, and radio network controllers (RNCs) that provide radio channel management services. One or more Nodes B connect to an RNC, while one or more RNCs connect to a serving gateway support node (SGSN), which is located in the 3G-WLAN core network. WLAN includes wireless access points that provide Wi-Fi

access and act like authentication, authorization, accounting (AAA) clients, forwarding security related messages to a AAA server. Finally, the 3G core network includes SGSN, the AAA server and an authentication center (AuC). SGSN provides mobility and session management services in UMTS, while the AAA server provides authentication services in WLAN. Both SGSN and the AAA server are connected to AuC, which contains the AVs of MS. In case that a MS wants to gain access to UMTS, it should execute the UMTS authentication and key agreement (UMTS-AKA) protocol [1]. On the other hand, if it wants to have access to WLAN, it should carry out the extensible authentication protocol EAP-AKA [3]. We briefly analyze these two protocols below as well as the replay attack protection mechanism.

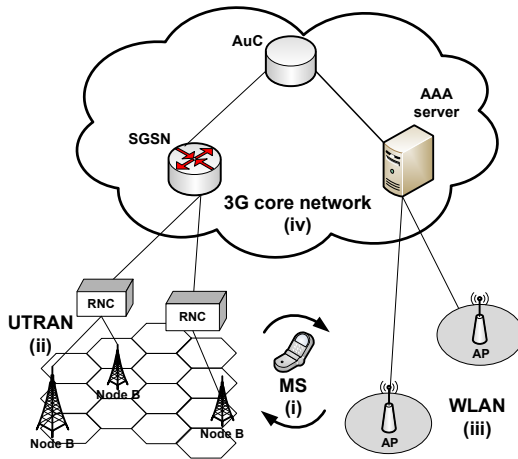


Fig. 1. 3G-WLAN integrated network architecture

2.2 UMTS-AKA

UMTS-AKA provides mutual authentication between MS and the UMTS network, based on the knowledge of a common secret key K , which is stored in USIM and AuC. In the initial step of UMTS-AKA, MS conveys a message that contains its identity to SGSN (see Fig. 2(a) – step a), and the latter conveys this identity to AuC, asking for fresh AVs for the specific MS (i.e., ADR procedure) (see Fig. 2(a) – step b). AuC fetches the permanent key K of MS and generates a batch of L_U ordered AVs [1]. Initially, it increments by one value the counter SQN_{HN} (i.e., $SQN_{HN} = SQN_{HN} + 1$) and sets $SQN_{AV} = SQN_{HN}$. Then, AuC generates a random number (RAND) and calculates the following parameters using the retrieved secret key K and a set of one-way hash functions (i.e., f_1, f_2, f_3, f_4, f_5) as shown in Fig. 3:

1. Message authentication code $MAC = f_1(SQN_{AV} || RAND || AMF)$. (The authentication and key management field (AMF) is reserved to be used whenever the UMTS network wants to convey to a USIM values of parameters that change over time).
2. Expected response $XRES = f_2(RAND)$.

3. Encryption key $CK = f_3(RAND)$.
4. Integrity key $IK = f_4(RAND)$.
5. Anonymity key $AK = f_5(RAND)$.
6. Authentication token $AUTN = SQN_{AV} \oplus AK || AMF || MAC$.

This is repeated L_U times for the generation of the batch of L_U AVs. Next, AuC forwards the L_U AVs in an ordered form (based on SQN_{AV}) to SGSN (see Fig. 2(a) – step c). Upon receiving them, SGSN selects the one with the smallest SQN_{AV} and conveys the pair of RAND and AUTN from the selected AV to MS (see Fig. 2(a) – step d). SGSN stores the remaining $L_U - 1$ AVs to use them in future authentications of MS.

Upon receiving this pair (RAND, AUTN), MS forwards it to USIM. The latter use this pair with the secret key K to compute AK (i.e., $AK = f_5(RAND)$) and SQN_{AV} (i.e., $SQN_{AV} = (SQN_{AV} \oplus AK) \oplus AK$), where $(SQN_{AV} \oplus AK)$ is included in the received AUTN). Then, USIM checks whether $SQN_{AV} > SQN_{MS}$ or $SQN_{MS} - SQN_{AV} \leq \alpha$. The parameter α plays a key role in the replay attack protection mechanism and it is called offset in this paper. If one of the above two conditions is true, USIM accepts the received AV; otherwise, it rejects it as a suspicion of an attack. Thus, the value of offset α is used from USIM to verify whether the received SQN_{AV} is among the last α generated.

In case USIM accepts the received SQN_{AV} and $SQN_{AV} > SQN_{MS}$ then, SQN_{MS} is set equal to the received SQN_{AV} (otherwise, SQN_{MS} remains the same). Afterwards, USIM computes $XMAC = f_1(SQN_{AV} || RAND || AMF)$ and an $AUTN'$ value using the previously generated AK key and XMAC. If the computed $AUTN'$ is equal to the received AUTN, the UMTS network is authenticated to MS and USIM computes $RES = f_2(RAND)$, $CK = f_3(RAND)$ and $IK = f_4(RAND)$. After that, MS forwards the computed RES to SGSN (see Fig. 2(a) – step e), which verifies whether the received RES is equal to XRES, included in the AV. If it is true, MS is authenticated to SGSN. In the last step of UMTS-AKA, USIM and SGSN convey to MS and RNC, respectively, the computed CK and IK (see Fig. 2(a) – step f), which are used to provide confidentiality and integrity services between MS and RNC.

2.3 EAP-AKA

EAP-AKA works similarly to UMTS-AKA, as depicted in Fig. 2(b). First, MS requests for an AV from the AAA server, through the wireless AP. The AAA server executes the ADR procedure to fetch AVs from AuC, which generates and sends back to the AAA server L_W ordered AVs (note that the number of generated AVs in EAP-AKA and UMTS-AKA, (i.e., L_W and L_U respectively) are not necessarily equal). The AAA server selects the AV with the smallest SQN_{AV} , obtains the pair of RAND and AUTN from the selected AV and conveys them to MS (the remaining $L_W - 1$ AVs are stored for future use). Upon receiving this pair (RAND, AUTN), MS forwards it to USIM, which verifies the associated SQN_{AV} and AUTN in order to authenticate the WLAN network, similarly to UMTS-AKA. Afterwards, USIM computes RES, CK and IK, and forwards them to MS. The latter calculates the master session key (MSK) using the CK and IK keys, and conveys RES to the AAA server. Upon receiving RES,

the AAA server authenticates MS (i.e., if $XRES=RES$), calculates MSK (using CK and IK keys), and sends it to the wireless AP. At the end of EAP-AKA, MS and WLAN are authenticated mutually, while MS and the wireless AP share MSK that is used to provide security services in WLAN

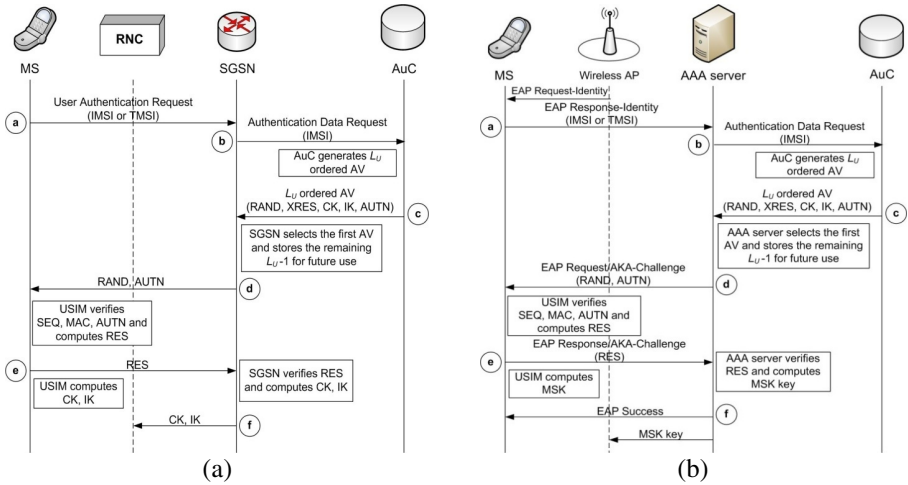


Fig. 2. Message exchange in (a) UMTS-AKA and (b) EAP-AKA

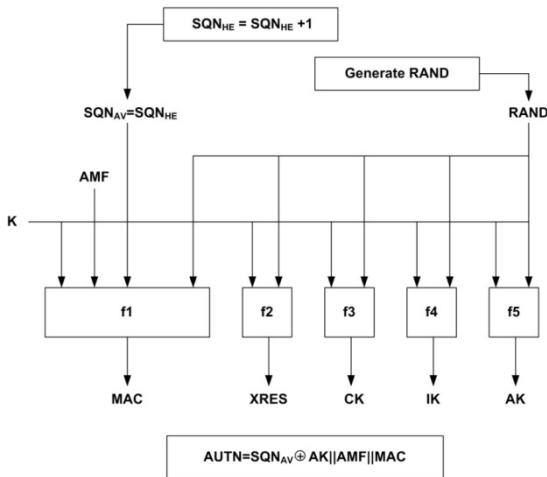


Fig. 3. AVs generation procedure

3 False Synchronizations in 3G-WLAN Integrated Networks

In this section we describe false synchronizations using a numerical example. Assume that: (a) a MS moves between UMTS and WLAN executing the UMTS-AKA and

EAP-AKA protocols, respectively (see Fig. 4); (b) the number of generated AVs in the UMTS and WLAN batch is $L_U = 6$ and $L_W = 4$, respectively; (c) the value of offset $\alpha = 7$. Initially, SGSN and the AAA server do not have any AV stored for the involved MS, and $SQN_{HN} = SQN_{MS} = 0$. Moreover, we use the term ‘handover’ to imply that: (i) a MS has an active connection and moves from UMTS to WLAN and vice versa, performing authentication or (ii) MS has no active connection and moves from UMTS to WLAN and vice versa, establishing a new connection in the newly visited network (executing UMTS-AKA or EAP-AKA).

We consider the time diagram of Fig. 4, assuming that the MS initially resides in UMTS and at the time t_1 executes UMTS-AKA. Since SGSN does not possess any stored AV for the involved MS, it executes the ADR procedure with AuC, which generates a batch of $L_U = 6$ AVs that contain $SQN_{AV} = 1, 2, 3, 4, 5, 6$, respectively and sets $SQN_{HN} = 6$. Next, AuC conveys the $L_U = 6$ newly generated AVs to SGSN, which selects the one with $SQN_{AV} = 1$ and conveys it to the MS (the remaining 5 AVs are stored for future use). The latter accepts the received AV and sets $SQN_{MS} = 1$. After five more successive UMTS-AKA authentications, at the time t_2 , SGSN performs again ADR and thus, AuC generates a new batch of $L_U = 6$ AVs with $SQN_{AV} = 7, 8, 9, 10, 11, 12$ (i.e., $SQN_{HN} = 12$) and sends them to SGSN. At the time t_3 , (after six more UMTS-AKA authentications) the SGSN has consumed all the previously generated AVs and MS has set $SQN_{MS} = 12$.

At the time t_4 , the MS handovers from UMTS to WLAN and executes EAP-AKA. The AAA server performs ADR, since it does not possess any AV for the involved MS. As a result, AuC generates a batch of $L_W = 4$ AVs with $SQN_{AV} = 13, 14, 15, 16$ respectively, sets $SQN_{HN} = 16$, and conveys them to the AAA server. The AAA sever

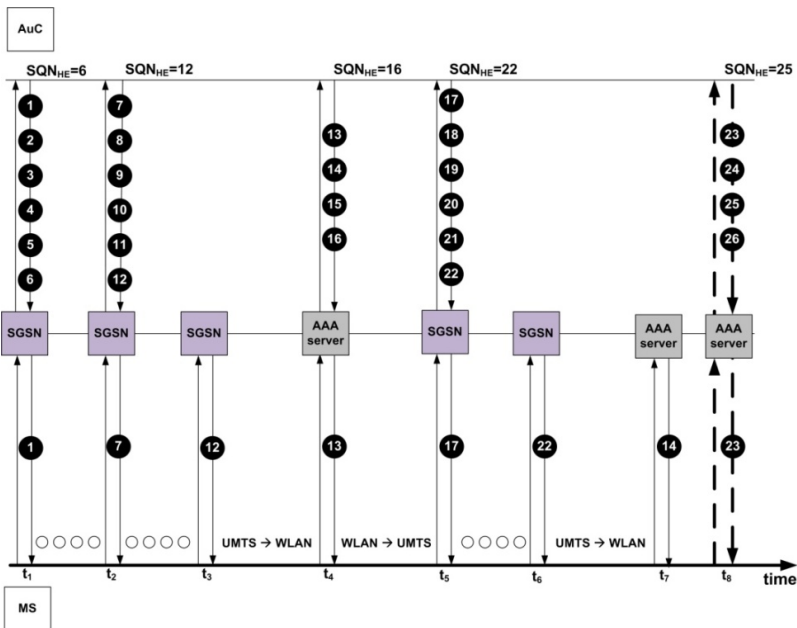


Fig. 4. Time diagram of the numerical example

sends the AV with $SQN_{AV} = 13$ to the MS, which is accepted since $SQN_{MS} = 5 < SQN_{AV} = 13$, and sets $SQN_{MS} = 13$. Then, the MS returns back to UMTS and at the time t_5 executes UMTS-AKA. As SGSN does not have any stored AV, it initiates ADR and thus, AuC generates a batch of AVs with $SQN_{AV} = 17, 18, 19, 20, 21, 22$ ($SQN_{HN} = 22$), which are forwarded to SGSN. The latter conveys the AV with $SQN_{AV} = 17$ to the MS, which accepts it (since $SQN_{MS} = 13 < SQN_{AV} = 17$) and sets $SQN_{MS} = 17$. After that, the MS executes five successive UMTS-AKA authentications, consuming all the above AVs that SGSN possess and sets $SQN_{MS} = 22$. At the time t_7 , the MS returns to WLAN and executes EAP-AKA. The AAA server, which has stored AVs for the MS, conveys the AV with $SQN_{AV} = 14$ to the MS. The latter rejects it ($SQN_{MS} - SQN_{AV} = 8 > 7$), since the employed freshness mechanism, erroneously, indicates that the current AV has been used in the past (i.e., false synchronization). At this point a false synchronization has occurred and a re-synchronization procedure is initiated.

4 Modeling Procedure

In this section an analytical model based on a Markov chain is developed that leads to the derivation of the probability of false synchronizations P_{sync} in 3G-WLAN integrated networks.

Table 1. Analytical model parameters

Notation	Description
N_k	Random variable that denotes if the MS resides in UMTS or WLAN
D_k	Difference between SQN_{UMTS} and SQN_{WLAN}
U_k	Number of AVs stored in SGSN
W_k	Number of AVs stored in the AAA server
SQN_{UMTS}	The last SQN_{AV} that MS has received from UMTS
SQN_{WLAN}	The last SQN_{AV} that MS has received from WLAN
$1/\mu_u$	Mean residence time in UMTS
$1/\mu_w$	Mean residence time in WLAN
λ_u	Authentication rate in UMTS
λ_w	Authentication rate in WLAN
L	Number of AVs that AuC generates in a batch
a	Offset value
P_{sync}	Probability of false synchronization

4.1 Identifying the Markov Chain Structure

To facilitate the system modeling, it is assumed that the residence time of a MS in UMTS and WLAN are exponentially distributed, and the authentication request process is Poisson. The following definitions and notations will be used in the analysis that follows:

- The residence time of a MS in UMTS and WLAN is assumed to follow an exponential distribution with mean $1/\mu_u$ and $1/\mu_w$, respectively.
- The authentication request rate of a MS in UMTS and WLAN is assumed to be Poisson with rate λ_u and λ_w , respectively.
- The number of the generated AVs L_U and L_W in UMTS and WLAN, respectively, is assumed to be equal to L (i.e., $L_U = L_W = L$).
- The last SQN_{AV} that a MS has received from UMTS (i.e., SGSN) and WLAN (i.e., the AAA server), are denoted by SQN_{UMTS} and SQN_{WLAN} , respectively.
- Consider the following quantities or processes embedded at the time instances k at which a network handover or authentication request (referred to as an H-A event) occurs:
 - Let $N_k \in \{0,1\}$ denote whether the MS resides in UMTS ($N_k = 0$) or in WLAN ($N_k = 1$) following the k^{th} H-A request.
 - Let $D_k \in (-\infty, +\infty)$, denote the difference between SQN_{UMTS} and SQN_{WLAN} (i.e., $D = SQN_{UMTS} - SQN_{WLAN}$), following the k^{th} H-A event.
 - Let $U_k \in [0, L)$ and $W_k \in [0, L)$ denote the number of AVs stored in SGSN and the AAA server, respectively, following the k^{th} H-A event. All operations in the set $\{0,1,2,\dots,L-1\}$ defined by the variables U_k and W_k use modular arithmetic with *modulo* L . To simplify notations, the symbol “*modL*” is omitted.

Table 1 summarizes the system and modeling parameters introduced above. In view of the aforementioned system modeling assumptions, it is not hard to show that the four dimensional process $E_k = \{N_k, D_k, U_k, W_k\}$ embedded at time instances k at which a network handover or authentication request occurs, is a discrete-time Markov chain.

It should be noted that the instances k at which an H-A event occurs are generated according to the following dynamics of handover and authentication requests: 1) Given that a MS is in UMTS (WLAN) in the current embedded instant, the next embedded time instant will be generated by an authentication request with probability $p_1 = \frac{\lambda_u}{\mu_u + \lambda_u}$ (with probability $p_2 = \frac{\lambda_w}{\mu_w + \lambda_w}$). 2) Given that MS is in UMTS in the current embedded instant, the next embedded time instant will be generated by a handover from UMTS to WLAN (and will also trigger an authentication in WLAN) with probability $p_3 = \frac{\mu_u}{\mu_u + \lambda_u}$. Similarly, given that a MS is in WLAN in the current embedded instant, the next embedded time instant will be generated by a handover from WLAN to UMTS (and will also trigger an authentication in UMTS) with probability $p_4 = \frac{\mu_w}{\mu_w + \lambda_w}$.

Before proceeding with the steady state analysis of the Markov chain, we elaborate on the conditions that trigger a false synchronization. Assume that the Markov chain is in some state (δ, i, j, z) with $N_k = \delta, D_k = i, U_k = j$ and $W_k = z$. A false synchronization occurs, if the following conditions are satisfied:

1. A handover from UMTS to WLAN takes place, i.e., $\delta: 0 \rightarrow 1$.
2. WLAN (i.e., the AAA server) has at least one AV to provide to the MS, i.e., $W_k = z > 0$.

3. WLAN (i.e., AAA server) provides to the MS an AV that includes a SQN_{AV} , which is at least $a + 1$ values smaller than the last SQN_{AV} that UMTS provided to the MS (i.e., $D_k = i > a$, which means $D_k = a + 1, a + 2, a + 3 \dots$).

The above are defined as false synchronization conditions from UMTS to WLAN. Note that the second condition guarantees that the AAA server has at least one “old” AV to provide to MS. On the contrary, if the second condition is not satisfied (i.e., $W_k = z = 0$), then a false synchronization does not occur when the MS moves to WLAN, since the AAA server performs ADR and fetches a batch of fresh AVs from AuC.

When a false synchronization occurs upon the MS handover from UMTS to WLAN (referred to as false synchronization in WLAN), we argue that the Markov chain jumps from the state $(0, i, j, z)$ to state $(1, -(j + 1), j, L - 1)$ since:

1. The MS is located in WLAN after the handover and thus, $N_k = 1$.
2. The false synchronization initiates ADR in WLAN, and thus, the batch of AVs that resides in WLAN is fresher than this of UMTS, while it was the opposite just before the execution of ADR. Therefore, if the remaining j AVs in UMTS contain $SQN_{AV} = d + 1, d + 2, \dots, d + j$, then the newly generated AVs of WLAN contain $SQN_{AV} = d + j + 1, d + j + 2, \dots, d + j + L$, where $d \in N$. As a result, the difference between the last SQN_{AV} that UMTS has provided (i.e., $SQN_{AV} = d$) and the last SQN_{AV} that the MS receives from WLAN (i.e., $SQN_{AV} = d + j + 1$), is $D_k = -(j + 1)$.
3. The value of U_k is the same, $U_k = j$.
4. From the newly generated AVs the performed authentication consumes one and thus, $W_k = L - 1$.

Summarizing the above we have:

Proposition 1: Upon a handover of MS from UMTS to WLAN, the Markov chain jumps from the state $(0, i, j, z)$ to $(1, -(j + 1), j, L - 1)$, if $i > a$. In addition, if $z > 0$ is satisfied, then the handover from UMTS to WLAN triggers a false synchronization in WLAN.

Similarly, we can infer that a false synchronization occurs upon a MS handover from WLAN to UMTS (referred to as false synchronization in UMTS) when the following conditions (referred to as false synchronization conditions from WLAN to UMTS) are satisfied:

1. A handover from WLAN to UMTS takes place, $\delta: 1 \rightarrow 0$.
2. $U_k = j > 0$.
3. $D_k = i < -a$ (i.e., $D_k = -(a + 1), -(a + 2), -(a + 3) \dots$).

Finally, similarly to proposition 1, we have:

Proposition 2: Upon a handover of MS from WLAN to UMTS, the Markov chain jumps from the state $(1, i, j, z)$ to $(0, z + 1, L - 1, z)$, if $i < -a$. Additionally, if $j > 0$, then the handover from WLAN to UMTS triggers a false synchronization in UMTS.

4.2 State Space Truncation

In order to derive the probability of false synchronizations, the steady state probabilities of the Markov chain E_k should be derived first. We notice that Markov chain E_k has a infinite state space as $D_k \in (-\infty, +\infty)$. Thus, we truncate its state space properly so that its dimensionality becomes finite, without compromising the accuracy of the resulting solution. The following proposition ensures that the original and the truncated Markov chains induce identical false synchronization events.

Proposition 3: Assume that \widetilde{E}_k is a truncated Markov chain of E_k , where $\widetilde{E}_k = \{\widetilde{N}_k, \widetilde{D}_k, \widetilde{U}_k, \widetilde{W}_k: \widetilde{N}_k \in [0,1], \widetilde{D}_k \in [-(a + 1), (a + 1)], \widetilde{U}_k \in [0, L), \widetilde{W}_k \in [0, L)\}$. Then, the evolution of E_k and \widetilde{E}_k is identical in the sub space that determines a false synchronization in UMTS or a false synchronization in WLAN and, thus, the two processes induce the same number of false synchronization events.

Proof: Notice that the values of D_k greater than $a + 1$ (i.e., $D_k = a + 2, a + 3, \dots$) or lower than $-(a + 1)$ (i.e., $D_k = -(a + 2), -(a + 3), \dots$) do not affect the evolution of the Markov chain in the sub space that determines the occurrence of a false synchronization in UMTS or a false synchronization in WLAN. Therefore, D_k parameter can be bounded between $-(a + 1) \leq D_k \leq \alpha + 1$, as elaborated below.

If $-(\alpha + 1) \leq D_k \leq \alpha + 1$, it is evident that the two Markov chains \widetilde{E}_k and E_k are identical, $\widetilde{E}_k \equiv E_k$. In case that both chains are in the state $(0, a + 1, j, z)$ and an authentication in UMTS occurs, then E_k jumps to the state $(0, a + 2, j - 1, z)$, since UMTS consumes one of the stored AVs reducing in this way U_k ($U_k = j - 1$), and increasing D_k ($D_k = a + 2$). On the other hand, \widetilde{E}_k jumps to the state $(0, a + 1, j - 1, z)$. After $n - 1$ successive authentications in UMTS, E_k and \widetilde{E}_k are in states $(0, a + n + 1, j - n, z)$ and $(0, a + 1, j - n, z)$, respectively. We observe that for authentication in UMTS, we have $\widetilde{N}_k = N_k, \widetilde{U}_k = U_k, \widetilde{W}_k = W_k$ and if $D_k > a$, then \widetilde{D}_k remains equal to $\alpha + 1$ while D_k increases (i.e., $D_k = \alpha + 1, \alpha + 2, \alpha + 3, \dots$).

Assume now that a handover from UMTS to WLAN occurs. If $z > 0$, then in both chains this handover triggers a false synchronization in WLAN. More specifically, E_k jumps from the state $(0, a + n + 1, j - n, z)$ to state $(1, -(j + 1), j - n, L - 1)$, since $D_k = a + n + 1 > a$ (see Proposition 1) and \widetilde{E}_k jumps from the state $(0, a + 1, j - n, z)$ to the state $(1, -(j + 1), j - n, L - 1)$, since $\widetilde{D}_k = a + 1 > a$. It is observed that $D_k = \widetilde{D}_k = -(j + 1)$ and thus, the two chains are identical. Therefore, it can be deduced that whenever a false synchronization in WLAN occurs in E_k , the same false synchronization in WLAN also occurs in \widetilde{E}_k . Similarly, we can prove that whenever a false synchronization in UMTS occurs in E_k , the same also happens in \widetilde{E}_k . Thus, we can conclude that the evolution of E_k and \widetilde{E}_k are similar in the sub space that determines a false synchronization in UMTS or a false synchronization in WLAN. This ends the proof.

It is evident that the truncated Markov chain $\widetilde{E}_k = \{\widetilde{N}_k, \widetilde{D}_k, \widetilde{U}_k, \widetilde{W}_k\}$ is ergodic and converges to a steady state. Let $\pi\{N_k, D_k, U_k, W_k\}$ be the steady state probabilities of \widetilde{E}_k .

4.3 Steady State Equations

The truncated Markov chain $\widetilde{E}_k = \{\widetilde{N}_k, \widetilde{D}_k, \widetilde{U}_k, \widetilde{W}_k\}$ is ergodic and converges to a steady state. Let $\pi_{(\delta,i,j,z)}$ be the steady state probability of a generic state (δ, i, j, z) , with $\widetilde{N}_k = \delta, \widetilde{D}_k = i, \widetilde{U}_k = j$ and $\widetilde{W}_k = z$. We divide the steady state equations of \widetilde{E}_k into two main sets, SET I (i.e., $\delta = 0$) and SET II (i.e., $\delta = 1$):

SET I $\delta = 0$: In this case, MS is located in UMTS. Depending on the values of i, j and z the following balance equations are satisfied.

I.1) If $i > 0$: The last SQN_{AV} that MS has received from UMTS is greater than the last one received from WLAN.

CASE I.1.A) $i = a + 1$: The Markov chain arrives at states with $i = a + 1$, only in cases that MS performs an authentication while it is in UMTS. We pinpoint two different sub-cases.

I.1.A.1) If $i - z \equiv -j \pmod{L}$: The Markov chain arrives at the states $(0, a + 1, j, z)$ either from the states $(0, a + 1, j + 1, z)$ or from the states $(0, a, j + 1, z)$ (e.g., in Fig. 5(a) the state B arrives from A or Z). Thus,

$$\pi_{(0,i,j,z)} = \frac{\lambda_u}{\mu_u + \lambda_u} \pi_{(0,i,j+1,z)} + \frac{\lambda_u}{\mu_u + \lambda_u} \pi_{(0,i-1,j+1,z)}. \quad (1)$$

I.1.A.2) Otherwise: The Markov chain arrives at the states $(0, a + 1, j, z)$, only from the states $(0, a + 1, j + 1, z)$, (e.g., in Fig.5(a) the state D can be reached only from C). Thus,

$$\pi_{(0,i,j,z)} = \frac{\lambda_u}{\mu_u + \lambda_u} \pi_{(0,i,j+1,z)} \quad (2)$$

CASE I.1.B) $i = z + 1$ and $j = L - 1$: The Markov chain arrives at states with $i = z + 1$ and $j = L - 1$, only if MS returns to UMTS and executes ADR. In this case, the states $(0, i, j, z)$ can be reached from the states with $(1, -a + 1, (0, 1, 2 \dots L - 1), z)$, (e.g., in Fig. 5(b) the state A can be reached only from B, C, D, ..., Z). Thus,

$$\pi_{(0,i,j,z)} = \frac{\mu_w}{\mu_w + \lambda_w} \pi_{(1, -(a+1), j^*, z)} \quad (3)$$

CASE I.1.C) For all other values of i , with $i > 0$ we recognize two sub-cases:

I.1.C.1) If $z \neq L - 1$: The states $(0, i, j, z)$ can be reached either from the states $(1, i - 1, j + 1, z)$ in MS handover from WLAN to UMTS event, or from the states $(0, i - 1, j + 1, z)$ in MS authentication in UMTS event. Thus,

$$\pi_{(0,i,j,z)} = \frac{\mu_w}{\mu_w + \lambda_w} \pi_{(1,i-1,j+1,z)} + \frac{\lambda_u}{\mu_u + \lambda_u} \pi_{(0,i-1,j+1,z)} \quad (4)$$

I.1.C.2) If $z = L - 1$: The states $(0, i, j, z)$ can be reached from the states $(0, i - 1, j + 1, z)$ in MS authentication in UMTS event. Therefore,

$$\pi_{(0,i,j,z)} = \frac{\lambda_u}{\mu_u + \lambda_u} \pi_{(0,i-1,j+1,z)} \quad (5)$$

I.2) If $i < 0$: The last SQN_{AV} that MS has received from UMTS is smaller than the last SQN_{AV} received from WLAN.

CASE I.2.A) $i = -(a - 1)$ or $j = L - 2$: The states $(0, i, j, z)$ can be reached from the states $(1, i - 1, j + 1, z)$ in MS handover from WLAN to UMTS. Thus,

$$\pi_{(0,i,j,z)} = \frac{\mu_w}{\mu_w + \lambda_w} \pi_{(1,i-1,j+1,z)} \quad (6)$$

CASE I.2.B) For all other values of i , with $i < 0$: It is similar to **I.1.C.1** and thus,

$$\pi_{(0,i,j,z)} = \frac{\mu_w}{\mu_w + \lambda_w} \pi_{(1,i-1,j+1,z)} + \frac{\lambda_u}{\mu_u + \lambda_u} \pi_{(0,i-1,j+1,z)} \quad (7)$$

SET II) $\delta = 1$: The equations of the second set are similar to those of the first and thus, are not analyzed in details.

II.1) $i < 0$,

CASE II.1.A) $i = -(a + 1)$,

II.1.A.1) If $i - j \equiv -z \pmod{L}$,

$$\pi_{(1,i,j,z)} = \frac{\lambda_w}{\mu_w + \lambda_w} \pi_{(1,i,j,z+1)} + \frac{\lambda_w}{\mu_w + \lambda_w} \pi_{(1,i-1,j,z+1)} \quad (8)$$

II.1.A.2) Otherwise,

$$\pi_{(1,i,j,z)} = \frac{\lambda_w}{\mu_w + \lambda_w} \pi_{(1,i,j,z+1)} \quad (9)$$

CASE II.1.B) $i = -(j + 1)$ and $z = L - 1$,

$$\pi_{(1,i,j,z)} = \frac{\mu_u}{\mu_u + \lambda_u} \pi_{(0,a+1,j,z^*)} \quad (10)$$

CASE II.1.C) For all other values of i with $i < 0$:

II.1.C.1) If $j \neq L - 1$,

$$\pi_{(1,i,j,z)} = \frac{\mu_u}{\mu_u + \lambda_u} \pi_{(0,i-1,j,z+1)} + \frac{\lambda_w}{\mu_w + \lambda_w} \pi_{(1,i-1,j,z+1)} \quad (11)$$

II.1.C.2) If $j = L - 1$,

$$\pi_{(1,i,j,z)} = \frac{\lambda_w}{\mu_w + \lambda_w} \pi_{(1,i-1,j,z+1)} \quad (12)$$

II.2) If $i > 0$,

CASE II.2.A) $i = a - 1$ or $z = L - 2$,

$$\pi_{(1,i,j,z)} = \frac{\mu_u}{\mu_u + \lambda_u} \pi_{(0,i-1,j,z+1)} \quad (13)$$

CASE II.2.B) For all other values of i , with $i > 0$:

$$\pi_{(1,i,j,z)} = \frac{\mu_u}{\mu_u + \lambda_u} \pi_{(0,i-1,j,z+1)} + \frac{\lambda_w}{\mu_w + \lambda_w} \pi_{(1,i-1,j,z+1)} \quad (14)$$

Finally, the sum of the steady state probabilities is equal to 1

$$\sum \sum \sum \sum \pi_{(\delta,i,j,z)} = 1 \quad (15)$$

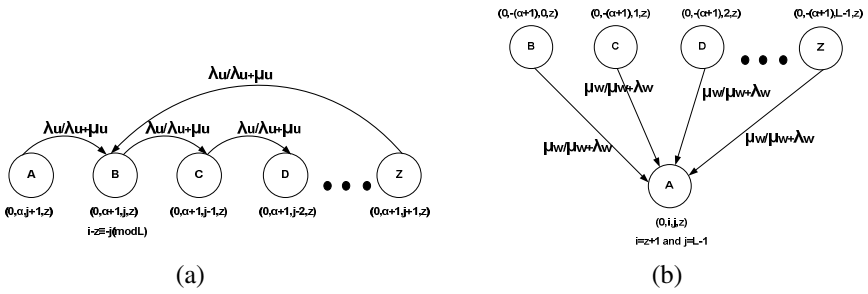


Fig. 5. Transitions examples for: (a) CASE I.1.A and (b) CASE I.1.B

4.4 Solution of Linear System and Derivation of P_{sync}

Equations (1)-(15) comprise a system of linear equations that has a unique solution (i.e., the steady state probabilities), which can be easily calculated using numerical methods. Having obtained the steady state probabilities of the truncated Markov chain, the probability of false synchronization P_{sync} can be calculated as follows. Let $P_{sync,W}$ be the probability of a false synchronization in WLAN, and $P_{sync,U}$ the probability of a false synchronization in UMTS. Considering the false synchronization conditions from UMTS to WLAN (see section 4.1), $P_{sync,W}$ can be derived as, $P_{sync,W} = \Pr [D_k = \alpha + 1 \text{ and the WLAN has at least one AV stored for MS}] \cdot \Pr [\text{MS handover from UMTS to WLAN}]$, which is equivalent to:

$$P_{sync,W} = \sum_{z>0, j=0}^{j=L-1} \pi \{N_k = 0, D_k = \alpha + 1, U_k = j, W_k = z\} \cdot \frac{\mu_u}{\mu_u + \lambda_u}$$

Similarly, $P_{sync,U}$ can be derived as $P_{sync,U} = \Pr [D = -(\alpha + 1) \text{ and UMTS has at least one AV stored for the MS}] \cdot \Pr [\text{MS handover from WLAN to UMTS}]$, which is equivalent to:

$$P_{sync,U} = \sum_{j>0, z=0}^{z=L-1} \pi \{N_k = 1, D_k = -(\alpha + 1), U_k = j, W_k = z\} \cdot \frac{\mu_w}{\mu_w + \lambda_w}$$

Finally, we can derive v as $P_{sync} = P_{sync,U} + P_{sync,W}$

To validate the accuracy of the analytical model, a discrete event-driven simulator written in C/C++ was developed. The statistical results collected from the simulation system, after attained the equilibrium state, were averaged to eliminate the randomness effect. It was observed that the maximum related error between the analytical and simulation results was 1%, which verifies the accuracy of the analytical model. The simulation methodology that we followed is analyzed in [4].

5 Conclusions

In this paper we used a generic modeling procedure to solve analytically the problem of false synchronization in 3G-WLAN integrated networks. First, we elaborated on false synchronizations using a numerical example. Using the generic modeling procedure, we developed an analytical solution based on a four dimensional Markov chain. We truncated the state space of the Markov chain in order to derive the steady state equations. Finally, we solved the linear system of equations to derive the probability of a false synchronization. As a future work, we could apply probabilistic model checking to verify the correctness of the analytical model and improve the modeling effectiveness [9].

References

1. 3GPP TS 33.102 (v11.0.0), 3G security; Security Architecture, Release 11 (2011)
2. 3GPP TS 33.234 (v11.2.0), 3G security; WLAN interworking security, Release 11 (2011)
3. Arkko, J., Haverinen, H.: EAP-AKA Authentication. RFC 4187 (January 2006)
4. Ntantogian, C., Xenakis, C., Stavrakakis, I.: Reducing False Synchronizations in 3G-WLAN Integrated Networks. *IEEE Transactions on Wireless Communications* 10(11), 3765–3773 (2011)
5. Lin, Y.-B., Chen, Y.-K.: Reducing Authentication Signalling Traffic in Third-Generation Mobile Network. *IEEE Transactions on Wireless Communications* 2(3), 493–501 (2003)
6. Zhang, Y., Fujise, M.: An Improvement for Authentication Protocol in Third Generation Wireless Networks. *IEEE Transactions on Wireless Communications* 5(9), 2348–2352 (2006)
7. Liang, W., Wang, W.: On Performance Analysis of Challenge/Response Based Authentication in Wireless Networks. *Computer Networks*, Elsevier Science 48(2), 267–288 (2005)
8. Lin, P., Cheng, S.-M., Liao, W.: Modeling Key Caching for Mobile IP Authentication, Authorization, and Accounting (AAA) Services. *IEEE Transactions on Vehicular Technology* 58(7), 3596–3608 (2009)
9. Alexiou, N., Deshpande, T., Basagiannis, S., Smolka, S., Katsaros, P.: Formal Analysis of the Kaminsky DNS Cache-Poisoning Attack Using Probabilistic Model Checking. In: *IEEE 12th International Symposium on High-Assurance Systems Engineering (HASE)*, San Jose, CA (November 2010)

Password Protected Smart Card and Memory Stick Authentication against Off-Line Dictionary Attacks

Yongge Wang

UNC Charlotte, Charlotte, NC 28223, USA
yonwang@uncc.edu

Abstract. We study the security requirements for remote authentication with password protected smart card. In recent years, several protocols for password-based authenticated key exchange have been proposed. These protocols are used for the protection of password based authentication between a client and a remote server. In this paper, we will focus on the password based authentication between a smart card owner and smart card via an untrusted card reader. In a typical scenario, a smart card owner inserts the smart card into an untrusted card reader and input the password via the card reader in order for the smart card to carry out the process of authentication with a remote server. In this case, we want to guarantee that the card reader will not be able to impersonate the card owner in future without the smart card itself. Furthermore, the smart card could be stolen. If this happens, we want the assurance that an adversary could not use the smart card to impersonate the card owner even though the sample space of passwords may be small enough to be enumerated by an off-line adversary.

1 Introduction

Numerous cryptographic protocols rely on passwords selected by users (people) for strong authentication. Since the users find it inconvenient to remember long passwords, they typically select short easily-rememberable passwords. In these cases, the sample space of passwords may be small enough to be enumerated by an adversary thereby making the protocols vulnerable to a *dictionary* attack. It is desirable then to design password-based protocols that resist off-line dictionary attacks (see, e.g., [16]).

The problem of password-based remote authentication protocols was first studied by Gong, Lomas, Needham, and Saltzer [6] who used public-key encryption to guard against off-line password-guessing attacks. In another very influential work (see, e.g., [16]), Bellare and Merritt introduced Encrypted Key Exchange (EKE), which became the basis for many of the subsequent works in this area. These protocols include SPEKE and SRP (see, e.g., [16]). Other papers addressing the above protocol problem can be found in [1-3]. In models discussed in the above mentioned papers, we can assume that there is a trusted client computer for the user to input her passwords. In a smart card based authentication system, this assumption may no longer be true. The smart card reader could be malicious and may intercept the user input passwords. Furthermore, a smart card could be stolen and the adversary may launch an off-line dictionary attack against the stolen smart card itself. It is the goal for this paper to discuss the security

models for smart card based remote authentication and to design secure protocols within these models.

In a practical deployment of smart card based authentication systems, there may be other system requirements. For example, we may be required to use symmetric cipher based systems only or to use public key based systems. Furthermore, the system may also require that the server store some validation data for each user or the server do not store any validation (this can be considered as identity based systems). Furthermore, there may be other requirements such as user password expiration and changes.

In the following, we use an example to show the challenges in the design of secure smart card based authentication protocols. A traditional way to store or transfer the secret key for each user is to use a symmetric key cipher such as AES to encrypt user's long term secret key with user's password and store the encrypted secret key on the user's smart card or USB memory card. This will not meet our security goals against off-line dictionary attacks. For example, in an RSA based public key cryptographic system, the public key is a pair of integers (n, e) and the private key is an integer d . With the above mentioned traditional approach, the smart card contains the value $AESE_{\alpha}(d)$ in its tamper resistant memory space, where α is the user's password. If such a card is stolen, the adversary could feed a message (or challenge) m to the smart card for a signature. The adversary needs to input a password in order for the smart card to generate a signature. The adversary will just pick one α' from her dictionary and ask the card to sign m . The card will "decrypt" the private key as $d' = AESE_{\alpha'}^{-1}(AESE_{\alpha}(d))$ and return a signature $s' = m^{d'} \bmod n$ on m . Then the adversary only needs to check whether $s'^e \bmod n = m$. If the equation holds, the adversary knows that the guessed password is correct. That is, $\alpha' = \alpha$. Otherwise, the attacker will remove α' from the dictionary. Similar attacks work for Guillou-Quisquater (GQ), Fiat-Shamir, and Schnorr zero-knowledge identification schemes.

This example shows that the "off-line" dictionary attack in the stolen smart card environments is different from the traditional client-server based off-line dictionary attacks. There have been quite a number of papers dealing with smart card based remote authentications (see, e.g., [4,5,7,10,12,13,15]). However, most of these papers present attacks on protocols in previous papers and propose new protocols without proper security justification (or even a security model).

2 Security Models

Halevi and Krawczyk [8, Sections 2.2-2.3] introduced a notion of security for remote authentication with memorable passwords. They provide a list of basic attacks that a password-based client-server protocol needs to guard against. Though these attacks are important for password-based authentication, they are not sufficient for password-protected smart card based remote authentication. In the following, we provide an extended list of attacks that a password-protected smart card based authentication protocol needs to protect against. An ideal password-protected smart card protocol should be secure against these attacks and we will follow these criteria when we discuss the security of password-protected smart card authentication protocols.

- **Eavesdropping.** The attacker may observe the communications channel.
- **Replay.** The attacker records messages (either from the communication channels or from the card readers) she has observed and re-sends them at a later time.
- **Man-in-the-middle.** The attacker intercepts the messages sent between the two parties (between user \mathcal{U} and smart card \mathcal{C} or between smart card \mathcal{C} and servers \mathcal{S}) and replaces these with her own messages. For example, if she sits between the user and the smart card, then she could play the role of smart card in the messages which it displays to the user on the card reader and at the same time plays the role of users to the smart card.
- **Impersonation.** The attacker impersonates the user (using a stolen smart card or a fake smart card) to an actual card reader to authenticate to the remote server, impersonate a card reader to a user who inserts an authentic smart card, impersonate a card reader and a smart card (a stolen card or a fake card but without the actual user), or impersonate the server to get some useful information.
- **Malicious card reader.** The attacker controls the card reader and intercepts the smart card owner's input password. Furthermore, the attacker controls all of the communications between smart card and the card owner via the card reader, and all of the communications between smart card and the remote server. For example, the attacker may launch a man in the middle attack between the smart card and smart card owner.
- **Stolen smart card.** The attacker steals the smart card and impersonates the smart card owner to the remote server via a trusted or a malicious smart card reader. In this case, the attacker could use the stolen card to impersonate the card owner with guessed passwords to the remote server with a limited time of failures since the server may disable the card from the server side after certain number of failures. If the attacker is allowed to use the card with guessed passwords to impersonate the card owner to the remote server for unlimited times of failures, then it will be considered as an on-line dictionary attack which scenario is not considered in this paper. However, the attacker is allowed for three kinds of further attacks that we will discuss in the following. One exception that we need to make in our security model is that we will not allow the attacker to control a malicious card reader to intercept the card owner's password and then to steal the smart card. There are three kinds of attackers based on the stolen smart card scenario:
 - *Smart card is tamper resistant with counter protection.* The attacker cannot read the sensitive information stored in the tamper resistant memory. Furthermore, the attacker may only issue a fixed amount of queries to the smart card to learn useful information. The smart card will be self-destroyed if the query number exceeds certain threshold (e.g., the GSM SIM card V2 or later has this capability).
 - *Smart card is tamper resistant without counter protection.* The attacker cannot read the sensitive information stored in the tamper resistant memory. However, the attacker may issue large amount of queries to the smart card to learn some useful information. For example, the attacker may setup a fake server and uses a malicious card reader to guess the potential password.
 - *Smart card is not tamper resistant.* The attacker (with the card) may be able to break the tamper resistant protection of the smart card and read the sensitive

information stored in the tamper resistant memory. In this case, the smart card will look more like a USB memory stick that stores the user credential with password protection. But still there is a difference here. In order for the user to use USB memory stick based credentials, the user needs the access to a trusted computer to carry out the authentication. However, one may assume that even if the smart card is not tamper resistant, it is not possible for a malicious card reader to read the sensitive information on the card within a short time period (e.g., during the time that the card owner inserts the card into the card reader for an authentication).

- **Password-guessing.** The attacker is assumed to have access to a relatively small dictionary of words that likely includes the secret password α . In an *off-line attack*, the attacker records past communications and searches for a word in the dictionary that is consistent with the recorded communications or carry out interaction with a stolen smart card without frequent server involvement (the attacker may carry out one or two sessions with server involved and all other activities without server involvement). In an *on-line attack*, the attacker repeatedly picks a password from the dictionary and attempts to impersonate U , C , U and C , or S . If the impersonation fails, the attacker removes this password from the dictionary and tries again, using a different password.
- **Partition attack.** The attacker records past communications, then goes over the dictionary and deletes those words that are not consistent with the recorded communications from the dictionary. After several tries, the attacker's dictionary could become very small.

We now informally sketch the definition of security models. We have three kinds of security models.

1. **Type I.** The attacker \mathcal{A} is allowed to watch regular runs of the protocol between a smart card reader \mathcal{R} (could be under the control of \mathcal{A}) and the server \mathcal{S} , can actively communicate with \mathcal{R} and \mathcal{S} in replay, impersonation, and man-in-the-middle attacks, and can also actively control a smart card reader when the card owner inserts the smart card and inputs her password. Furthermore, the attacker may steal the smart card from the user (if this happens, we assume that the attacker has not observed the user password from the previous runs of protocols) and issue a large amount of queries to the smart card using a malicious card reader. However, we assume that the smart card is tamper resistant and the attacker could not read the sensitive data from the smart card. A protocol is said to be *secure* in the presence of such an attacker if (i) whenever the server \mathcal{S} accepts an authentication session with \mathcal{R} , it is the case that the actual user U did indeed insert her smart card into \mathcal{R} and input the correct password in the authentication session; and (ii) whenever a smart card accepts an authentication session with \mathcal{S} , it is the case that \mathcal{S} did indeed participate in the authentication session and the user U did indeed input the correct password.
2. **Type II.** The capability of the attacker is the same as in the Type I model except that when the attacker steals the smart card, it can only issue a fixed number of queries to the smart card using a malicious card reader. If the number of queries exceeds the threshold, the smart card will be self-destroyed.

3. **Type III.** The capability of the attacker is the same as in the Type I model except that when the attacker steals the smart card, it will be able to read all of the sensitive data out from the smart card. But we will also assume that when a card owner inserts the card into a malicious card reader for a session of authentication, the card reader should not be able to read the information stored in the tamper resistant section of the card. In another word, the smart card is not tamper resistant only when the attacker can hold the card for a relatively long period by herself. Another equivalent interpretation of this assumption is that the attacker may not be able to intercept the password via the card reader and read the information stored in the card at the same time.

3 Smart Card Based Secure Authentication and Key Agreement

3.1 Symmetric Key Based Scheme: SSCA

In this symmetric key based smart card authentication scheme SSCA, the server should choose a master secret β and protect it securely. Note that this master secret β could be different for different users (cards). The Setup phase is as follows:

- For each user with identity \mathcal{C} and password α , the card maker (it knows the server's master secret β) sets the card secret key as $K = \mathcal{H}(\beta, \mathcal{C})$ and stores $\mathcal{K} = \mathcal{E}_\alpha(K)$ in the tamper resistant memory of the smart card, where \mathcal{E} is a symmetric encryption algorithm such as AES and \mathcal{H} is a hash algorithm such as SHA-2.

In the SSCA scheme, we assume that the smart card has the capability to generate unpredictable random numbers. There are several ways for smart card to do so. One of the typical approaches is to use hash algorithms and EPROM. In this approach, a random number is stored in the EPROM of the smart card when it is made. Each time, when a new random number is needed, the smart card reads the current random number in the EPROM and hash this random number with a secret key. Then it outputs this keyed hash output as the new random number and replace the random number content in the EPROM with this new value. In order to keep protocol security, it is important for the smart card to erase all session information after each protocol run. This will ensure that, in case the smart card is lost and the information within the tamper resistant memory is recovered by the attacker, the attacker should not be able to recover any of the random numbers used in the previous runs of the protocols. It should be noted that some smart card industry uses symmetric encryption algorithms to generate random numbers. Due to the reversible operation of symmetric ciphers, symmetric key based random number generation is not recommended for smart card implementation.

Each time when the user inserts her smart card into a card reader (which could be malicious), the card reader asks the user to input the password which will be forwarded to the smart card.

1. Using the provided password α , the card decrypts $K = \mathcal{D}_\alpha(\mathcal{K})$. If the password is correct, the value should equal to $\mathcal{H}(\beta, \mathcal{C})$. The card selects a random number R_c , computes $R_A = \mathcal{E}_K(\mathcal{C}, R_c)$, and sends the pair (\mathcal{C}, R_A) to the card reader which will be forwarded to the server.

2. The server recovers the value of (\mathcal{C}, R_c) using the key $K = \mathcal{H}(\beta, \mathcal{C})$ and verifies that the identity \mathcal{C} of the card is correct. If the verification passes, the server selects a random number R_s , computes $R_B = \mathcal{E}_K(\mathcal{C}, R_s)$, and sends (\mathcal{C}, R_B, C_s) to the card reader which forwards it to the card. Here $C_s = \text{HMAC}_{sk}(\mathcal{S}, \mathcal{C}, R_s, R_c)$ is the keyed message authentication tag on $(\mathcal{S}, \mathcal{C}, R_s, R_c)$ under the key $sk = \mathcal{H}(\mathcal{C}, \mathcal{S}, R_c, R_s)$ and \mathcal{S} is the server identity string.
3. The card recovers the value of (\mathcal{C}, R_s) using the key $K = \mathcal{H}(\beta, \mathcal{C})$, computes $sk = \mathcal{H}(\mathcal{C}, \mathcal{S}, R_c, R_s)$, and verifies the HMAC authentication tag C_s . If the verification passes, it computes its own confirmation message as $C_c = \text{HMAC}_{sk}(\mathcal{C}, \mathcal{S}, R_c, R_s)$ and sends C_c to the server. The shared session key will be sk .
4. The server accepts the communication if the HMAC tag C_c passes the verification.

The protocol SSCA message flows are shown in the Figure [1](#).

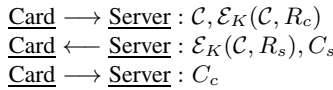


Fig. 1. Message flows in SSCA

In the following, we use heuristics to show that SSCA is secure in the Type I and Type II security models. If the underlying encryption scheme \mathcal{E} and HMAC are secure, then eavesdropping, replay, man-in-the-middle, impersonation, password-guessing, and partition attacks will learn nothing about the password since no information of password is involved in these messages. Furthermore, a malicious card reader can intercept the password, but without the smart card itself, the attacker will not be able to learn information about the secret key $K = \mathcal{D}_\alpha(\mathcal{K})$. Thus the attacker will not be able to impersonate the server or the card owner. When the attacker steals the smart card (but she has not controlled a card reader to intercept the card owner password in the past), she may be able to insert the card into a malicious card reader and let the card to run the protocols with a fake server polynomial many times. In these protocol runs, the attacker could input guessed password α' . The smart card will output $(\mathcal{C}, \mathcal{E}_{K'}(\mathcal{C}, R_c))$ where $K' = \mathcal{D}_{\alpha'}(\mathcal{K})$. Since the attacker has no access to the actual server (this is an off-line attack), the attacker can not verify whether the output $(\mathcal{C}, \mathcal{E}_{K'}(\mathcal{C}, R_c))$ is in correct format. Thus the attacker has no way to verify whether the guessed password α' is correct. In a summary, the protocol is secure in the Type I and Type II security models.

The protocol SSCA is not secure in the Type III security model. Assume that the attacker has observed a previous valid run of the protocol (but did not see the password) before steals the smart card. For each guessed password α' , the attacker computes a potential key $K' = \mathcal{D}_{\alpha'}(\mathcal{K})$. If this key K' is not consistent with the observed confirmation messages in the previous run of the protocol, the attacker could remove α' from the password list. Otherwise, it guessed the correct password.

If we revise the attacker's capability in Type III model by restricting the attacker from observing any valid runs of the protocol before she steals the smart card, we get a new security model which we will call Type III' model. We can show that the protocol SSCA

is secure in the Type III' model. The heuristic is that for an attacker with access to the value $\mathcal{K} = \mathcal{E}_\alpha(K)$, he will not be able to verify whether a guessed password is valid off-line. For example, for each guessed password α' , she can compute $K' = \mathcal{D}_{\alpha'}(\mathcal{K})$. But she has no idea whether K' is the valid secret key without on-line interaction with the server. Thus the protocol is secure in the Type III' security model.

Remarks: Modification of the protocol may be necessary for certain applications. For example, if the card identification string \mathcal{C} itself needs to be protected (e.g., it is the credit card number), then one certainly does not want to transfer the identification string \mathcal{C} along with the message in a clear channel.

3.2 Public Key Based Scheme: PSCAb

In this section, we introduce a public key based smart card authentication scheme with bilinear groups: PSCAb, it is based on the identity based key agreement protocol from IEEE 1363.3 [9,14].

In the following, we first briefly describe the bilinear maps and bilinear map groups.

1. G and G_1 are two (multiplicative) cyclic groups of prime order q .
2. g is a generator of G .
3. $\hat{e} : G \times G \rightarrow G_1$ is a bilinear map.

A bilinear map is a map $\hat{e} : G \times G \rightarrow G_1$ with the following properties:

1. bilinear: for all $g_1, g_2 \in G$, and $x, y \in \mathbb{Z}$, we have $\hat{e}(g_1^x, g_2^y) = \hat{e}(g_1, g_2)^{xy}$.
2. non-degenerate: $\hat{e}(g, g) \neq 1$.

We say that G is a bilinear group if the group action in G can be computed efficiently and there exists a group G_1 and an efficiently computable bilinear map $\hat{e} : G \times G \rightarrow G_1$ as above. For convenience, throughout the paper, we view both G and G_1 as multiplicative groups though the concrete implementation of G could be additive elliptic curve groups.

Let k be the security parameter given to the setup algorithm and \mathcal{IG} be a bilinear group parameter generator. We present the scheme by describing the three algorithms: **Setup**, **Extract**, and **Exchange**.

Setup: For the input $k \in \mathbb{Z}^+$, the algorithm proceeds as follows:

1. Run \mathcal{IG} on k to generate a bilinear group $G_\rho = \{G, G_1, \hat{e}\}$ and the prime order q of the two groups G and G_1 . Choose a random generator $g \in G$.
2. Pick a random master secret $\beta \in \mathbb{Z}_q^*$.
3. Choose cryptographic hash functions $\mathcal{H}_1 : \{0, 1\}^* \rightarrow G$, $\mathcal{H}_2 : \{0, 1\}^* \rightarrow \{0, 1\}^*$, and $\pi : G \times G \rightarrow \mathbb{Z}_q^*$. In the security analysis, we view \mathcal{H}_1 , \mathcal{H}_2 , and π as random oracles.

The system parameter is $\langle q, g, G, G_1, \hat{e}, \mathcal{H}_1, \mathcal{H}_2, \pi \rangle$ and the master secret key is β .

Extract: For a given identification string $\mathcal{C} \in \{0, 1\}^*$, the algorithm computes a generator $g_{\mathcal{C}} = \mathcal{H}_1(\mathcal{C}) \in G$, and sets the private key $d_{\mathcal{C}} = g_{\mathcal{C}}^\beta$ where β is the master secret key. The algorithm will further compute $g_{\mathcal{S}} = \mathcal{H}_1(\mathcal{S}) \in G$ where \mathcal{S} is the server

identity string, and store the value (C, g_S, d'_C) in the tamper resistant smart card where $d'_C = \mathcal{E}_{\mathcal{H}_2(\alpha)}(d_C)$, α is card owner's password. and \mathcal{E} is the encryption function that could be defined in one of the following ways:

1. \mathcal{E} is a standard symmetric cipher such as AES
2. $\mathcal{E}_{\mathcal{H}_2(\alpha)}(d_C) = \text{AES}_{\mathcal{H}_2(\alpha)}(d_C) + i_0$ where $i_0 = \min\{i : \text{AES}_{\mathcal{H}_2(\alpha)}(d_C) + i \in G, i = 0, 1, \dots\}$. For an inputted password α' , d_C is computed as $\text{AES}_{\mathcal{H}_2(\alpha')}^{-1}(d'_C - i_0)$ where $i_0 = \min\{i : \text{AES}_{\mathcal{H}_2(\alpha')}^{-1}(d'_C - i) \in G, i = 0, 1, \dots\}$.
3. $\mathcal{E}_{\mathcal{H}_2(\alpha)}(d_C) = d_C^{\mathcal{H}_2(\alpha)}$

Exchange: The algorithm proceeds as follows.

1. The card selects $x \in_R Z_q^*$, computes $R_A = g_C^x$, and sends it to the Server via the card reader.
2. The Server selects $y \in_R Z_q^*$, computes $R_B = g_S^y$, and sends it to the card.
3. The card computes $s_A = \pi(R_A, R_B)$, $s_B = \pi(R_B, R_A)$, and $d_C = \mathcal{D}_{\mathcal{H}_2(\alpha')}(d'_C)$ where \mathcal{D} is the decryption function and α' is the user inputted password. If d_C is not an element of G , the card chooses the value for sk as a random element of G_1 . Otherwise, the card computes the value $sk = \hat{e}(g_C, g_S)^{(x+s_A)(y+s_B)\beta}$ as

$$\hat{e}\left(d_C^{(x+s_A)}, g_S^{s_B} \cdot R_B\right).$$

4. The card computes $K_1 = \mathcal{H}(sk, R_A, R_B, C, S, 1)$, $K_2 = \mathcal{H}(sk, R_A, R_B, C, S, 2)$, and sends $C_C = \text{HMAC}_{K_1}(C, S, R_A, R_B)$ to the server. K_2 is the shared secret.
5. The server computes $s_A = \pi(R_A, R_B)$, $s_B = \pi(R_B, R_A)$ and sk as

$$\hat{e}(g_C, g_S)^{(x+s_A)(y+s_B)\beta} = \hat{e}\left(g_C^{s_A} \cdot R_A, g_S^{(y+s_B)\beta}\right).$$

6. The server verifies whether C_C is correct. If the verification passes, the server computes $K_1 = \mathcal{H}(sk, R_A, R_B, C, S, 1)$, $K_2 = \mathcal{H}(sk, R_A, R_B, C, S, 2)$ and sends $C_S = \text{HMAC}_{K_1}(S, C, R_B, R_A)$ to the card. K_2 is the shared secret.
7. The card verifies the value of C_S .

The smart card should never export the value of sk to the card reader during the protocol run. However, the smart card may need to export K_2 to the card reader in certain applications. The protocol PSCAb message flows are shown in the Figure 2.

In the following, we use heuristics to show that PSCAb is secure in the Type I, Type II, and Type III security models. It should be noted that if the encryption function is chosen as a standard symmetric cipher such as AES, then PSCAb is only weakly secure

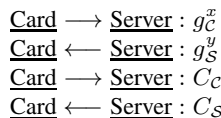


Fig. 2. Message flows in PSCAb

in the Type III security model as follows. When the attacker has access to the value d'_C , she could remove those α' from her dictionary such that $\mathcal{D}_{\mathcal{H}_2(\alpha')}(d'_C)$ is not an element of G . In other words, PSCAb is secure in the type III security model only if the remaining dictionary is still large enough.

The security of the underlying identity based key agreement protocol IDAK [14] (it is called Wang Key Agreement protocol in [9]) is proved in [14]. Furthermore, the eavesdropping, replay, man-in-the-middle, impersonation, password-guessing, and partition attacks will learn nothing about the password since no information of password is involved in these messages. Furthermore, these attackers will learn nothing about the private keys d_C and β based on the proofs in [14]. For an attacker with access to the information d'_C (the attacker may read this information from the stolen smart card), she may impersonate the card owner to interact with the server. Since the attacker could not compute the correct value sk , she will not be able to generate the confirmation message C_C . Thus the server will not send the server confirmation message back to the attacker. In another word, the attacker will get no useful information for an off-line password guessing attack. Furthermore, even if the attacker has observed previous valid protocol runs, it will not help the attacker since the smart card does not contain any information of the session values x of the previous protocols runs.

Remarks: In the protocol PSCAb, it is important to have the card to send the confirmation message to the server first. Otherwise, PSCAb will not be secure in the Type III security model. Assume that the server sends the first confirmation message. After the attacker obtains the value d'_C from the smart card, she could impersonate the user by sending the vale R_A to the server. After receiving the server confirmation message, she will remove α' from her dictionary such that

$$sk' = \hat{e} \left(\mathcal{D}_{\mathcal{H}(\alpha')}(d'_C)^{(x+s_A)}, g_S^{s_B} \cdot R_B \right)$$

is not consistent with the confirmation message C_S .

3.3 Public Key Based Scheme: PSCA

In the previous section, we presented a protocol PSCAb based on the identity based key agreement protocol IDAK. In this section, we briefly discuss a protocol based on the HMQV key agreement protocol [11]. Let g be the generator of the group G_ρ , q be the prime order of g , and h be a constant. In this case, the server and the smart card will both have public keys.

The server private/public key pair is (b, g^b) . The smart card private/public key pair is (a, g^a) . The data stored on the smart card is: $(a \times \mathcal{H}(\alpha), g^b)$. In the following, we use \mathcal{C} and \mathcal{S} to denote the client (smart card) and server identity strings respectively.

1. The card selects $x \in_R [1, q - 1]$, computes $R_A = g^x$, and sends it to the server.
2. Server selects $y \in_R [1, q - 1]$, computes $R_B = g^y$, and sends it to the card.
3. The card decrypts the private key a via the user inputed password, computes $\pi_A = \mathcal{H}(R_A, \mathcal{S})$, $\pi_B = \mathcal{H}(R_B, \mathcal{C})$, $s_A = (x + \pi_A a) \bmod q$, and the shared session key: $K_{\text{HMQV}} = (R_B \cdot (g^b)^{\pi_B})^{s_A h}$.

4. The server computes $\pi_A = \mathcal{H}(R_A, \mathcal{S})$, $\pi_B = \mathcal{H}(R_B, \mathcal{C})$, $s_B = (y + \pi_B b) \bmod q$, and the shared session key: $K_{\text{HMQV}} = (R_A \cdot (g^a)^{\pi_A})^{s_B h}$.

Remarks: Heuristics could be used to show that this protocol is secure in the Type I and Type II security models. However this protocol is not secure in the Type III security model. After the attacker obtains the value $(a \times \mathcal{H}(\alpha), g^b)$, the attacker could recover the password from $a \times \mathcal{H}(\alpha)$ and the smart card public key g^a . However, if g^a is only known to the server, then PSCA should be secure in the Type III model. We conjecture that it may be impossible to design HMQV based protocols that are secure in the Type III model if the public key of the smart card is available to the attackers.

3.4 Public Key Based Scheme with Password Validation Data at Server: PSCAV

In previous sections, we discussed two protocols SSCA and PSCAb that the server does not store any password validation data. In this section, we discuss a protocol where the server needs to store password validation data for each card. One of the disadvantages of this kind of protocols is that if the card owner wants to change her password, the server has to be involved.

It should be noted that the password based remote authentication protocols that have been specified in the IEEE 1363.2 [9] are not secure in our models. The major reason is that the only secure credential that a client owns is the password. If the smart card owner inputs her password on an untrusted card reader, the card reader could just record the password and impersonates the client to the server without the smart card in future.

Before we present our scheme PSCAV, we briefly note that the protocol PSCAb in Section 3.2 can be easily modified to be a password protected smart card authentication scheme that the server stores user password validation data. In Section 3.2, the identity string for each user is computed as $g_C = \mathcal{H}(\mathcal{C}) \in G$. For protocols with password validation data, we can use a different way to compute the identity strings. In particular, assume that the user \mathcal{U} has a password α , then the identity string for the user will be computed as $g_C = \mathcal{H}(\mathcal{C}, \alpha) \in G$ and the private key for the user will be $d_C = g_C^\beta$ where β is the master secret key. The value $(\mathcal{C}, g_S, \mathcal{E}_{\mathcal{H}_2(\alpha)}(d_C))$ will be stored in the tamper resistant smart card, and the value g_C will be stored in the server database for this user. The remaining protocol runs the same as in Section 3.2. We can call the above mentioned protocol as PSCAbV

Now we begin to describe our main protocol PSCAV for this section. Assume that the server has a master secret β (β could be user specific also). For each user with password α , let the user specific generator be $g_C = \mathcal{H}_1(\mathcal{C}, \alpha, \beta)$, the value $g_C^{\mathcal{H}_2(\alpha)}$ is stored on the smart card, where \mathcal{H}_2 is another independent hash function. The value $g_C = \mathcal{H}_1(\mathcal{C}, \alpha, \beta)$ will be stored in the server database for this user. The remaining of protocol runs as follows:

1. The card selects random x and sends $R_A = g_C^x$ to the server.
2. Server selects random y and sends $R_B = g_C^y$ to the card.
3. The card computes $u = \mathcal{H}(\mathcal{C}, \mathcal{S}, R_A, R_B)$ where \mathcal{S} is the server identity string, $sk = g_C^{y(x+u\alpha)}$, and sends $C_c = \mathcal{H}(sk, \mathcal{C}, \mathcal{S}, R_A, R_B, 1)$ to the server
4. After verifying that C_c is correct, server computes $u = \mathcal{H}(\mathcal{C}, \mathcal{S}, R_A, R_B)$, $sk = g_C^{y(x+u\alpha)}$, and sends $C_s = \mathcal{H}(sk, \mathcal{S}, \mathcal{C}, R_B, R_A, 2)$ to the card.

The protocol PSCAV message flows are the same as for the PSCAb protocol message in the Figure 2 (but with different interpretation for the variables in the figure).

In the following, we use heuristics to show that PSCAV is secure in the Type I, Type II, and Type III security models. For the PSCAV protocol, the eavesdropping, replay, man-in-the-middle, card (client) impersonation, password-guessing, and partition attacks will learn nothing about the password due to the hardness of the Diffie-Hellman problem. For the attacker that carries out a server impersonation attack, it will receive the value R_A , and send a random R_B to the card. The attacker will then receive the card confirmation message C_C . The attacker may not launch an off-line dictionary attack on these information since for each guessed password α' , it has no way to generate a session key sk' due to the hardness of the Diffie-Hellman problem. For an attacker with access to the information $g_C^{\mathcal{H}_2(\alpha)}$ (the attacker may read this information from the stolen smart card), she may impersonate the card owner to interact with the server. The attacker may send a random R_A to the server which could be based on $g_C^{\mathcal{H}_2(\alpha)}$, and receives a value R_B from the server. But it cannot compute the correct value for sk based on these information. Thus it could not send the confirmation message C_C to the server. Thus the server will not send the server confirmation message back to the attacker. In other words, the attacker will get no useful information for an off-line password guessing attack. Furthermore, even if the attacker has observed previous valid protocol runs, it will not help the attacker since the smart card does not contain any information of the session values x of the previous protocols runs.

Remarks: The attack described in the Remarks at the end of Section 3.2 could be used to show that it is important to have the card to send the confirmation message to the server first in the protocol PSCAV also.

4 Remote Authentication with Password Protected Portable Memory Sticks

In this section, we investigate the scenario that the user stores her private key on a USB memory stick. Our goal is that if the memory stick is lost, then the adversary will not be able to mount an off-line dictionary attack to impersonate the legitimate user. Since a memory stick will not have its own CPU, the owner has to insert the memory stick into a trusted computer (otherwise, the malicious computer could just intercept the password and copy the content on the memory sticks and impersonates the owner in future). Thus the security model for this kind of protocols are different from the Type I, II, III models that we have discussed in Section 2, but are closely related to the Type III model. Specifically, we will have the following Type IV model for portable memory sticks.

- **Type IV.** The attacker \mathcal{A} is allowed to watch regular runs of the protocol between a client \mathcal{C} (the client will only insert her memory stick and input her password to trusted computers that are not controlled by the attacker) and the server \mathcal{S} , can actively communicate with \mathcal{C} and \mathcal{S} in replay, impersonation, and man-in-the-middle attacks, and can also steal the memory stick from the user and read the content in the memory stick. A protocol is said to be *secure* in the presence of such an attacker

if (i) whenever the server \mathcal{S} accepts an authentication session with \mathcal{C} , it is the case that the actual user \mathcal{U} did indeed insert her memory stick into a computer \mathcal{C} and input the correct password in the authentication session; and (ii) whenever a client \mathcal{C} accepts an authentication session with \mathcal{S} , it is the case that \mathcal{S} did indeed participate in the authentication session and the user \mathcal{U} did indeed input the correct password.

Remark: It is an open question whether the security models Type III and Type IV are equivalent.

References

1. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated Key Exchange Secure against Dictionary Attacks. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, p. 139. Springer, Heidelberg (2000)
2. Boyko, V., MacKenzie, P.D., Patel, S.: Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 156–171. Springer, Heidelberg (2000)
3. Bresson, E., Chevassut, O., Pointcheval, D.: Security proofs for an efficient password-based key exchange. In: ACM Conference on Computer Communications Security, pp. 241–250. ACM Press (2003)
4. Chen, Y., Chou, J., Huang, C.: Comment on four two-party authentication protocols (2010)
5. Das, M.L., Saxena, A., Gulati, V.P.: A dynamic id-based remote user authentication scheme. IEEE Transactions on Consumer Electronics 50, 629–631 (2004)
6. Gong, L., Lomas, T.M.A., Needham, R.M., Saltzer, J.H.: Protecting poorly chosen secrets from guessing attacks. IEEE J. Selected Areas in Communications 11, 648–656 (1993)
7. Goriparthi, T., Das, M.L., Saxena, A.: An improved bilinear pairing based remote user authentication scheme. Computer Standards and Interfaces 31, 181–185 (2009)
8. Halevi, S., Krawczyk, H.: Public-key cryptography and password protocols. ACM Transactions on Information and System Security 2(3), 230–268 (1999)
9. IEEE 1363. Standard specifications for public-key cryptography (2005)
10. Juang, W.S., Chen, S.T., Liaw, H.T.: Robust and efficient password-authenticated key agreement using smart cards. IEEE Trans. Industrial Electronics 55, 2551–2556 (2008)
11. Krawczyk, H.: HMQV: A High-Performance Secure Diffie-Hellman Protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546–566. Springer, Heidelberg (2005)
12. Lee, Y.S., Nam, J., Won, D.H.: Vulnerabilities in a Remote Agent Authentication Scheme Using Smart Cards. In: Nguyen, N.T., Jo, G.-S., Howlett, R.J., Jain, L.C. (eds.) KES-AMSTA 2008. LNCS (LNAI), vol. 4953, pp. 850–857. Springer, Heidelberg (2008)
13. Rhee, H.S., Kwon, J.O., Lee, D.H.: A remote user authentication scheme without using smart cards. Computer Standards and Interfaces 31, 6–13 (2009)
14. Wang, Y.: Efficient identity-based and authenticated key agreement protocol (2005), <http://eprint.iacr.org/2005/108>
15. Xiang, T., Wong, K., Liao, X.: Cryptanalysis of a password authentication scheme over insecure networks. Computer and System Sciences 74, 657–661 (2008)
16. Zhao, Z., Dong, Z., Wang, Y.: Security analysis of a password-based authentication protocol proposed to IEEE 1363. Theoretical Computer Science 352, 280–287 (2006)

Distributed Path Authentication for Dynamic RFID-Enabled Supply Chains*

Shaoying Cai¹, Yingjiu Li¹, and Yunlei Zhao²

¹ Singapore Management University, 80 Stamford Road, Singapore 178902

² Fudan University, No. 825 Zhangheng Road, Shanghai, China 201203
{shaoyingcai.2009,yjli}@smu.edu.sg, ylzhaofudan.edu.cn

Abstract. In this paper, we propose a distributed path authentication solution for dynamic RFID-enabled supply chains to address the counterfeiting problem. Compared to existing general anti-counterfeiting solutions, our solution requires non sharing of item-level RFID information among supply chain parties, thus eliminating the requirement on high network bandwidth and fine-grained access control. Our solution is secure, privacy-preserving, and practical. It leverages on the standard EPCglobal network to share information about paths and parties in path authentication. Our solution can be implemented on standard EPC class 1 generation 2 tags with only 720 bits storage and no computational capability.

1 Introduction

Supply chain is a network involving multiple parties such as suppliers, transporters, storage facilities, distributors, and retailers that participate in the production, delivery, and sale of a product [5]. It is difficult to monitor a supply chain since the involving parties are distributed at multiple locations or even across countries. It is therefore critical to address the counterfeiting problem, where an adversary injects fake goods into a supply chain. The counterfeiting problem has become a major threat to supply chains. According to the 2011 report of International Chamber of Commerce, it is estimated that the counterfeiting accounts for 5-7% of world trade, or about 600 billion U.S. dollars per year [6]. The ratio of counterfeiting is even higher in dynamic supply chains where the involving parties and the paths of processed goods may not be fixed.

Radio Frequency IDentification (RFID) technology has been recently used to facilitate real-time monitoring of supply chains so as to thwart counterfeiting threats. Most existing solutions in industry rely on sharing and processing massive RFID information at item level collected by each supply chain party over the internet. Such solutions inevitably incur high network bandwidth (due to large volume of processed goods) and fine-grained access control (due to security requirements by different parties). It is therefore difficult, even impossible,

* The second author's work is supported in part by the Office of Research at Singapore Management University. The third author is supported in part by an NSFC grant No. 61070248, and a grant from Shanghai Municipal Education Commission.

to implement such solutions in dynamic supply chain environments, where the involving parties may not have pre-existing trust relationship for sharing their RFID information in a secure and efficient manner.

Since dynamic supply chains are prevalent in living supply chains [16], we propose a new distributed path authentication scheme to thwart counterfeiting in such environment. Our scheme verifies whether a tag comes from a reliable source and has been precessed by a series of legitimate entities in a supply chain. Compare to existing solutions, our scheme eliminates the needs of sharing item-level information among supply chain parties; thus, it does not require pre-existing trust relationship nor fine-grained access control. The verification of a tag's path is based on the information carried by the tag and certain auxiliary information obtained from a trusted server in EPCglobal network, which is a standard infrastructure for RFID-enabled supply chains. While our scheme is designed for the most dynamic RFID-enabled supply chains, it is also suitable for a supply chain that has fix partnership or/and static goods processing procedure.

The challenges of designing our scheme come in two aspects. First, the tag storage is restricted to no more than 1088 bits for the most popular low-cost standard C1 G2 tags [4]. In order to prove that a tag has been processed by a series of entities, the tag should carry certain credentials generated by the entities. To avoid complicated key management at item level, a natural way is to use the entities' signatures on a tag's ID as the credentials. However, it is not practical to store all signatures and public keys on a tag as the tag's storage is limited, especially in the case that such information may continually increase as the tag goes through more entities. Our solution keeps the information stored on a tag in constant size, which is less than 1088 bits, satisfying the storage constraint for EPC C1 G2 standard tag. In our solution, an ordered multi-signature scheme [11] is adopted to generate a constant-size signature of the entities on the tag's ID. A path index is used to indicate the series of entities which have processed a tag. The index is stored on the tag instead of the entities' public keys, while the detailed information about the path is stored on a trusted server such as EPCglobal Discovery Server.

The second challenge is to reduce the communication load between an entity and the trusted server when verifying a signature. To verify a signature, an entity queries the trusted server with the index of the path. The server sends the public keys of the entities in the path to the entity. The communication load increases as the path getting longer in a naive solution. We reduce the communication load to a constant level regardless of path length in our solution. Due to the specific constructions of the underlying ordered multi-signature scheme, in our scheme, the server only needs to send an aggregated "path public key" instead of the public keys of the entities. With the "path public key", one can verify whether a signature is generated orderly by the entities in the path or not. The "path public key" even has smaller size than a single public key. In the case where a batch of tags share the same path, a verifier only needs to query the trusted server once for the aggregated "path public key" in practice.

The security of our scheme relies on the unforgeability of the underlying ordered multi-signature scheme. Our scheme is secure in a sense that an adversary cannot forge any valid tag or path. To protect the privacy of each tag, we take advantage of supply chain's batch processing property. In a batch, each tag's information is encrypted with the same key. The key is divided to several shares with a secret sharing scheme. Each tag stores a share of the key together with its encrypted information. Only authorized readers can access the whole batch of tags, recover the key and then decrypt the contents stored on the tags. Our scheme is privacy preserving in a sense that an adversary cannot identify any valid tag, or distinguish whether two valid tags have taken the same path or not. Our scheme leverages on standard EPCglobal network and can be implemented on standard EPC class 1 generation 2 tags with only 720 bits storage and no computational capability.

2 Background

Many protocols such as [23,29] have been designed to provide mutual authentication or tag authentication in RFID systems. These protocols can be used to prevent counterfeiting in supply chains. However, in order to monitor a supply chain, these proposals require the manager to access the databases of all the entities in the supply chain. Therefore, they rely on high quality network connection and fine-grained access control. These requirements are obstacles in deployment of dynamic supply chain management systems. In addition, most of the existing solutions rely on non-standard tags with certain computational abilities, such as hash operation. This will incur high cost for not using standard low-cost tags.

The most related work address the counterfeiting problem in RFID-enabled supply chains based on standard EPC class 1 Gen 2 tags. In Zanetti, Fellmann and Capkun's scheme [30], the entities cooperate together to verify the tags' genuineness without revealing information to each other. This scheme cannot resist tracking attack since any reader can read out each tag's ID. Blass, Elkhyaoui and Molva proposed TRACKER [10] and its extension [9]. In TRACKER system, each tag stores encrypted verifiable ID and path information. The manager decrypts the information and verifies whether the tag has gone through a valid path. TRACKER dose not need the entities in a supply chain to have any connection except the initialization phase. However, it requires the manager to possess the secret keys of all involving entities, the manager can only verify the tags from a set of pre-fixed paths. Therefore, it is difficult to implement TRACKER in dynamic supply chains.

2.1 Dynamic RFID-Enabled Supply Chain

A supply chain consists of multiple entities. We model a dynamic supply chain according to three properties: the affiliation of each entity, the membership management of the supply chain and the logistics flow of the supply chain. In a dynamic supply chain, each entity is independent of each other; any entity can

freely join or leave; the logistics flow is not fixed. An RFID-enabled dynamic supply chain management system should meet the requirements below.

- Each reader is independent with other readers and has an unique ID. Note that some readers may belong to a same entity. Even that, we assume the readers are independent with each other and have no pre-existing connections, such as network connection among themselves to cater for the most flexible deployment condition.
- Each reader does not share its secret (eg. private key) with other readers.
- Each reader in the supply chain is isomorphic, with the same functionalities. Each reader should be able to initialize the tags, identify the tags, verify the tags and update the tags.

2.2 Adversary Model

Figure 1 illustrates our adversary model. The entities in a supply chain provide relative secure environments within their territories, where the tags are beyond the accessible distance of an adversary. During the transportation of tagged goods between entities, a “hit and run” adversary can only approach the goods for a short period of time from a not-so-close distance.

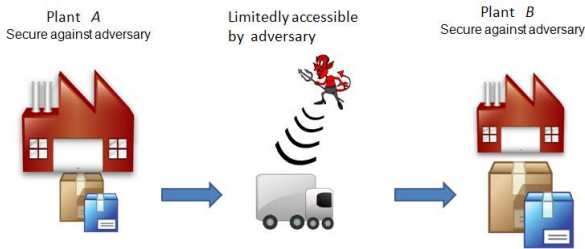


Fig. 1. Adversary Model of Supply Chain

The “hit and run” adversary model was firstly proposed by Ari Juels [18]. Several works [22, 21, 19] have also adopted this model in designing secure RFID systems. The rationals to adopt this model in supply chains are: 1) the tagged goods usually rapid change their physical locations and ownerships so that it is difficult for an adversary to keep in the working distance of the tags (normally no more than ten meters); 2) as both readers and tags work in short range, an adversary bringing a reader into a monitored environment like a shop or warehouse might face difficulties in attempting prolonged intelligence gathering [18].

2.3 Requirements of RFID-Enabled Path Authentication System

In dynamic supply chains, we list the following practice requirements for designing RFID-enabled path authentication:

- Any valid reader can extract a tag’s ID and the exact path that the tag has passed through in the supply chain. However, no readers needs to store the information of all possible paths in its own database in advance.
- An adversary cannot create new tag or modify existing one without being detected. A tag cannot pass the verification process for a path by which it has not passed.
- No efficient adversary can link the state information stored in a tag to the tag’s identity. No efficient adversary can distinguish whether two tags have taken the same path or not.
- Path authentication can be performed on general EPC Class 1 Gen 2 tags, which have at most 1088 bits and no computational capability.

3 Our Construction

Our system contains three components: tags, readers and a trusted server. Each tag stores a tag ID, a path code, and a signature on the tag’s ID generated by the readers in the path. Any valid reader has a pair of public key and private key and a random number used for generating the path code. A valid reader is able to extract each tag’s ID and the path code; while to verify them, it needs to connect with the trusted server which stores the reader’s public information and detailed path information. In designing a secure and privacy-preserving path authentication scheme, we use the following building tools.

3.1 Building Tools

Bilinear map: A bilinear map is a map $e : \mathbb{G} \times \mathbb{G} \rightarrow G_T$, where: (a) \mathbb{G} is a (multiplicative) cyclic groups of prime order p ; (b) $|\mathbb{G}| = |G_T|$; (c) g is a generator of \mathbb{G} . The bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow G_T$ satisfies the following properties: (a) Bilinear: for all $x, y \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, $e(x^a, y^b) = e(x, y)^{ab}$; (b) Non-degenerate: $e(g, g) \neq 1$. We call an algorithm \mathcal{G} that outputs (p, \mathbb{G}, G_T, e) as above a bilinear-group generation algorithm.

Path Encoding Method: Noubir et al. [25] proposes to encode a software’s state machine using polynomials such that the exact sequence of states visited during run-time generates a unique “mark”. We adopt this technique in generating the path code. Suppose there is a path $P_v = \{R_{v_1}, \dots, R_{v_{l_v}}\}$, where l_v is the length of path P_v , v_i represents the reader’s identity of the i th step in path P_v ; we assign each reader R_j with a unique random number $a_j \in \mathbb{F}_q$, where q is a large prime. A path is represented with a polynomial on \mathbb{F}_q . Then the polynomial corresponding to a path $P_v = \overrightarrow{R_{v_1} \dots R_{v_{l_v}}}$ is defined below (all operations are in \mathbb{F}_q):

$$Q_{P(x)} := \sum_{i=1}^{l_v} a_{v_i} x^{l_v-i} \tag{1}$$

Given a generator x_0 of \mathbb{F}_q , we calculate the path code as $\phi(P_v) := Q_{P_v}(x_0)$ and identify a path P_v using its polynomial evaluation $\phi(P_v)$.

Secret Sharing Scheme A (τ, n) -secret sharing scheme is an algorithm that divides data D into n pieces in such a way that: 1) knowledge of any τ or more pieces makes D easily computable; 2) knowledge of any $\tau - 1$ or fewer pieces leaves D completely undetermined (in the sense that all its possible values are equally likely) [28]. We adopt the Tiny Secret Sharing (TSS) [19] proposed by Juels et al. in our construction.

Ordered Multisignature Scheme Ordered multisignature scheme (OMS) allows signers to attest to a common message as well as the order in which they signed. The concept is raised by Boldyreva et al. in [11]. A construction of OMS is provided in [11], and we denote it as BGOY-OMS scheme from now on. We summarize BGOY-OMS system as follows. Each BGOY-OMS system has global information $I = (p, \mathbb{G}, \mathbb{G}_T, e, g, H)$, where $(p, \mathbb{G}, \mathbb{G}_T, e)$ is generated by a bilinear-group generation algorithm \mathcal{G} , g is a random generator of \mathbb{G} , and $H : \{0, 1\}^* \rightarrow \mathbb{G}$ is cryptographic hash function.

- **Key Generation:** On input I , the algorithm chooses random $s, t, u \in \mathbb{Z}_p$ and returns $(S = g^s, T = g^t, U = g^u)$ as pk and (s, t, u) as sk .
- **Signing:** On inputs $sk_i, m, \sigma, L = (pk_1, \dots, pk_{i-1})$, the algorithm first verifies whether Equation 2 defined below holds and if not, outputs \perp . (This step is skipped for the first signer, i.e. if $i = 1$, for whom σ is defined as $(1_G, 1_G)$.) Then it parses σ as (Q, W) and chooses random $w \in \mathbb{Z}_p$ and computes $W' = W \cdot g^w, X = (W')^{t_i + iu_i}, Y = (\prod_{j=1}^{i-1} T_j(U_j)^j)^w$ and $Q' = H(m)^{s_i} \cdot Q \cdot X \cdot Y$. Finally, it returns (Q', W') .
- **Verification:** On inputs $\{(pk_1, \dots, pk_n), m, \sigma\}$, the algorithm first checks that all of pk_1, \dots, pk_n are distinct and outputs 0 if not. Then it parses σ as (Q, W) and checks if

$$e(Q, g) \stackrel{?}{=} e(H(m), \prod_{i=1}^n S_i) \cdot e(\prod_{i=1}^n T_i(U_i)^i, W). \tag{2}$$

If so, it outputs 1. If not, it outputs 0.

3.2 Protocol Details

The tag information, including tag ID, path code, and signature is encrypted. The tags in the same batch share the same encryption key. Then the encryption key is distributed using TSS secret sharing scheme and each tag stores a share of the key together with encrypted data. A valid reader can collect enough shares, recover the key, and decrypts the information of each tag. For each tag, after decryption, a valid reader obtains the tag ID, a path code, and a signature on the tag ID. Querying a trusted server with the path code, the reader gets the aggregated “path public key” of the path which is computed from the readers’ public keys, and uses it to verify the signature. If the tag is valid, then the reader can update the path code and the signature and encrypt them with a new random key. Finally, the reader stores the new tag information on the tag. The tags are

processed by batch in supply chain management. Polynomial signature based path encoding method [25], BGOY-OMS [11], and TSS [19] are incorporated in the design of our system.

System Setup: A BGOY-OMS system is set up by running a bilinear-group generation algorithm \mathcal{G} for output $(p, \mathbb{G}, \mathbb{G}_T, e, g, H)$. Choosing a large prime q , and a random number $x_0 \in \mathbb{Z}_q$, we get $I = (p, q, \mathbb{G}, \mathbb{G}_T, e, g, H, x_0)$, which is the global information for the scheme. Assume that there are m readers in total. Each reader R_j is assigned with public key $pk_j = (S_j, T_j, U_j)$ and secret key $sk_j = (s_j, t_j, u_j)$, where s_j, t_j, u_j are randomly chosen from \mathbb{Z}_p , $1 \leq j \leq m$. Each reader R_j is also assigned with a random number $a_j \in \mathbb{Z}_q$, where a_j will be used in generating a path code.

A trusted server publishes the global information I , each reader’s public key pk_j and random number a_j . The trusted server also stores each path P_v ’s information, including “ $pathcode_v, l_v, P_v = \{R_{v_1}, R_{v_2} \dots, R_{v_{l_v}}\}, ppk_v = (ppk1_v, ppk2_v) = (\prod_{j=1}^{l_v} S_{v_j}, \prod_{j=1}^{l_v} T_{v_j} (U_{v_j})^j)$ ”, where l_v is the number of readers in path P_v , v_j denotes the identity of the j th reader in path P_v , $pathcode_v$ is the path code of P_v generated using Equation (1) in \mathbb{F}_q , and $ppk_v = (ppk1_v, ppk2_v) = (\prod_{j=1}^{l_v} S_{v_j}, \prod_{j=1}^{l_v} T_{v_j} (U_{v_j})^j)$ is each path’s public key stored in a path record. With ppk_v , a valid reader can verify the signature on the tag without knowing any other reader’s individual public key. This will reduce the communication load between a reader and the trust server. In case a reader needs to verify the signature on a tag based on all involving readers’ public keys, it can also get the public keys from the trusted sever. Table 1 shows the contents stored on the trusted server.

Table 1. Contents on Trusted Server

System Parameters	$(p, q, \mathbb{G}, \mathbb{G}_T, e, g, H, x_0)$
Reader Information $R_j,$ for $1 < j < m$ m is the number of readers	$pk_j = (S_j, T_j, U_j), a_j$
Path Information P_v	$pathcode_v, l_v, P_v = \{R_{v_1}, R_{v_2} \dots, R_{v_{l_v}}\},$ $ppk_v = (ppk1_v, ppk2_v) = (\prod_{j=1}^{l_v} S_{v_j}, \prod_{j=1}^{l_v} T_{v_j} (U_{v_j})^j)$

Batch Initialization of the Tags: Suppose that a batch \mathcal{T} of n tags enters in a supply chain, where each tag is denoted as T_i with unique ID id_i , for $i \in \{1, \dots, n\}$. The tags can be initialized by a reader valid $R_x, 1 \leq x \leq m$. In particular, R_x generates a key k and n shares of k using TSS-scheme, where each share is denoted as s_i , for $1 \leq i \leq n$. For each tag T_i , R_x generates a signature $\sigma_i = (Q_i, W_i)$ on the tag ID id_i under its private key using BGOY-MOS scheme. Then R_x sets $pathcode_i$ of each tag T_i to a_x . Finally, R_x encrypts $(id_i, \sigma_i, pathcode_i)$ with the key k , and stores $\{s_i, E_k(id_i, \sigma_i, pathcode_i)\}$ on T_i . After initializing the batch of tags, R_x queries the trust server to check whether the path $P = \{a_x, 1, \{R_x\}, ppk = (S_x, T_x U_x)\}$ already exists; if not, R_x inserts

path P to the database on the trust server. Then R_x releases the batch of tags into the supply chain.

Interactions between Reader and Tag: When a batch \mathcal{T} of tags in the supply chain arrives at reader R_y , where $1 \leq y \leq m$. Each tag T_i in the batch stores a state $st_i = \{s_i, E_k(id_i, \sigma_i = (Q_i, W_i), pathcode_i)\}$. Reader R_y firstly reads all the tags in \mathcal{T} to get st_i for all $1 \leq i \leq n$. Using at least τ shares, R_y recovers a key k , and decrypts the information on each tag $E_k(id_i, \sigma_i, pathcode_i)$ and gets $\{id_i, \sigma_i = (Q_i, R_i), pathcode_i\}$. According to $pathcode_i$, R_y gets the path's information $P = \{pathcode_i, l, \{R_{P_0}, \dots, R_{P_l}\}, ppk = (ppk_1, ppk_2)\}$ from the trusted server. If $e(Q_i, g) = e(H(id_i), ppk_1) \cdot e(ppk_2, W)$, then T_i passes the verification. To update the batch of tags, R_y generates a new key k' and n shares of k' in TSS, where each share is denoted as s'_i . For each tag T_i , R_y shall update both the signature σ_i and the path code $pathcode_i$. To do these, R_y chooses a random number w in \mathbb{Z}_p , computes $W'_i = W_i \cdot g^w$, $Q'_i = W_i^{t_y + (l+1)u_y} \cdot ppk_1^w \cdot Q \cdot H(id_i)^{s_y}$ and $pathcode'_i = pathcode_i \cdot x_0 + a_y$. R_y updates each tag T_i by writing $\{s'_i, E_{k'}(id_i, \sigma'_i = (Q'_i, W'_i), pathcode'_i)\}$ to the tag. Finally, the reader queries the trusted server: if the path $P_{new} = \{pathcode'_i, l + 1, \{R_{P_0}, \dots, R_{P_l}, R_y\}, ppk = (ppk_1 \cdot S_y, ppk_2 \cdot T_y U_y^{l+1})\}$ does not exist in the trusted server, then path P_{new} will be added for future queries.

Implementation Details: TSS scheme can be implemented with Reed-Solomon code [26]. A Reed-Solomon code is specified as $RS(N, K)_S$. A code-word has S bits. A reader chooses a pre-key with $K \cdot S$ bits and encodes the pre-key to N shares with each share S bits. A hash value of the pre-key is used as the encryption key for a batch of N tags. Reed Solomon encoding and decoding have been implemented on an Intel Core 2 CPU 6320 1.86GHz in [2]. Choosing the parameters (N, K, S) as $(32768, 16384, 16)$, the fastest algorithm achieves 6.17 Mbytes throughput per second for encoding, and 2.73 Mbytes throughput per second for decoding. Both encoding procedure and decoding procedure consume less than one second. Suppose there are 20000 tags in a batch, one can firstly choose a pre-key with $16 * 16384$ bits, encode them to 32768 symbols. Each share contains one symbol. Then one can select 20000 shares, randomly store on share on each tag. Any reader that can successfully read more than 16384 tags is able to get the pre-key. Regarding the encryption algorithm, Blowfish [3] is an appropriate choice. According to [3], Blowfish has good performance that achieves 64.386MB per second of encryption throughput on a Pentium 4 2.1 GHz processor under Windows XP SP 1.

4 Analysis

Both the security and privacy of our scheme rely on the security and privacy properties of the underlying secret sharing scheme, encryption scheme and OMS scheme. Due to space limit, we leave detailed proof to an extended version of this paper.

4.1 Security

We assume that in supply chain management, an adversary's goal is to insert counterfeited goods into the supply chain. The security goal of our system is to prevent an adversary from forging a tag's internal state so that the tag is considered from a reliable source and has gone through a valid path that has not actually been taken by the tag in the supply chain.

The security of our solution is based on the unforgeability of the BGOY-OMS scheme. Intuitively, the unforgeability of BGOY-OMS scheme can be described as follows: given an uncorrupted party with key pair (pk, sk) , a forger cannot generate a valid BGOY-OMS signature of the uncorrupted party on any message m' if the forger does not know the party's signature on m' . Note that in the original BGOY-OMS unforgeability game, given a key pair (pk, sk) , the adversary is able to get signature on any tag's ID under the key sk so as to learn useful information. In our system, the adversary cannot get a valid reader's signature on any ID it chooses since the valid reader will verify the genuineness of any tag before generating a signature on its ID. Hence, the adversary in our system is weaker than the adversary in the BGOY-OMS unforgeability game. BGOY-OMS scheme is secure against the forger; hence it is also secure against the adversary in our system. Unless an adversary has corrupted all the readers in a path with path code $pathcode$, it cannot forge a tuple $(ID, \sigma, pathcode)$ such that σ is a valid signature on ID generated by the readers in the path.

While an adversary cannot forge any valid new tuple $(ID, \sigma, pathcode)$ by itself, another way to counterfeit a tag is to use an existing valid tuple in the supply chain. From this aspect, the counterfeiting countermeasure of our system relies on the batch processing property. Only can valid readers read the whole batch of tags and decrypt the contents of the tags. A "hit and run" adversary is not able to read more than $\tau - 1$ tags in a batch, thus cannot recover the decryption key. In EPC Class 1 Gen 2 tags, the user memory bank can be protected by access pin. We use the encryption/decryption key as the access pin for the tags; therefore, an adversary cannot get the complete content stored on a tag and our system is secure against the counterfeiting attack.

4.2 Privacy

The privacy of RFID path authentication can be considered at two levels: tag unlinkability and path unlinkability. Tag unlinkability requires that no efficient adversary can link the state information stored in a tag to the tag's identity. Path unlinkability requires that no efficient adversary can distinguish whether two tags have taken the same path or not. In our scheme, each tag stores a copy of encrypted ID, pathcode and signature together with a single share of the encryption key. The privacy of our system relies on honest behaving of valid readers. Each valid reader uses a new random key to encrypt the updated state for each tag. An hit-and-run adversary who cannot collect enough shares for recovering the encryption keys cannot distinguish between any two ciphertexts of the same tag and any two ciphertexts of different tags. Thus our scheme

preserves tag unlinkability. Similarly, our scheme preserves path unlinkability since an adversary cannot obtain any information about a tag's path from the content of the tag.

4.3 Performance

We analyze the performance of our solution in three aspects: computation requirements, communication requirements and storage requirements.

Computational Requirements: Our scheme does not require tag to perform any computation. All the computation can be performed at RFID reader side. In computing the computational load of a reader, we omit the cheap operations such as Reed Solomon encoding, decoding, encryption and decryption using Blowfish, hash operation, and point addition on elliptic curve. We only count the relative expensive operations such as point multiplication on elliptic curve and paring. A reader needs to perform three paring operations to verify the signature and four point multiplication four updating the signature in each tag.

Due to batch processing in supply chain, we can reduce the computational cost if a batch of tags share the same path. Assuming that a batch of tagged goods is transferred in a supply chain without being mixed with other goods, then a reader can sign the batch of tags with the same random number. That is, each tag shares the same randomization factor W in the signature. To update the signature, the reader firstly computes (W', X, Y) , which requires three point multiplication operations, and uses (W', X, Y) to generate each tag's signature. With (W', X, Y) , the reader's computational load is reduced to one point multiplication in generating each tag's signature. For signature verification, since each tag in the batch stores the same randomization factor W in the signature, a reader can pre-compute $e(ppk_2, W)$ and store the value. The computational requirements for each tag is reduced to two paring operations. Since all the computation can be performed on reader side, our solution is applicable on standard low-cost tags with no computation capability.

Communication Requirement: The verification of each tag's ID and path code requires a valid reader to connect to the trusted server. The reader sends the path code to the server, then the server returns necessary path's information. In the case that a batch of tags passed the same path, the reader needs only one path information from the server. Since batch processing is commonly used in supply chain practice, the communication load required for processing a batch of tags should be almost constant. The communication load can be further reduced if a reader stores path information for frequently used paths in its own database.

Storage Requirement: In each tag T_i , we need to store $\{s_i, E_k(id_i, \sigma_i = (Q_i, W_i), pathcode_i)\}$. s_i is a share of encryption key k . As we implement TSS scheme with Reed-Solomon code, s_i can be a symbol, which we use 16-bit string (a symbol's length depends on the parameters of Reed-Solomon code). Tag ID id_i is an EPC code, which has 96 bits. Tag signature σ_i generated by BGOY-OMS scheme consists of two elements on \mathbb{G} . For 80-bit security level with embedding

degree $k = 2$, an element in \mathbb{G} can be represented in 512 bits. For embedding degree $k = 6$, the length can be reduced to 237 bits [13]. Hence, the storage requirement for σ_i is at least 474 bits. We use 80 bits to represent a path ID, which supports at most 2^{80} different path codes. We adopt Blowfish block cipher [27] for encryption which has a block size of 64-bits. Thus, we need 720 bits in total. Our system can thus be implemented with EPC Class 1 Gen 2 tags with an extensible EPC memory bank (scalable between 16-480 bits), a scalable user memory bank (64-512 bits), a TID bank (32 bits), and a reserved bank (64 bits), which are available on the market [4].

5 Conclusions

In this paper, we proposed a distributed path authentication scheme for dynamic supply chains. Dynamic supply chain is a prevalent supply chain structure in real world that is inherently vulnerable to the injection of counterfeiting goods and at same time challengeable to design an RFID-enabled system which can monitor the whole supply chain. We design a path authentication scheme to enable any valid reader to verify the exact path that a tag has taken without requiring the reader to have any kind of connection with other readers; it is thus suitable for dynamic supply chains where supply chain partners (or readers) may not have pre-existing trust relationship. Our scheme is built upon tiny secret sharing scheme, multisignature scheme, polynomial signature scheme, and encryption scheme. Our system is secure, privacy-preserving and practical. Our scheme leverages on sharing path information on standard EPCglobal network, and can be implemented on standard low-cost RFID tags with no computation capability and limited memory.

References

1. http://www.edn.com/article/458737-Counterfeitcomponentsremains_a_huge_electronics_supply_chain_problem.php
2. http://algo.epfl.ch/~didier/reed_solomon.html
3. http://www1.cse.wustl.edu/~jain/cse567-06/ftp/encryption_perf/index.html
4. <http://www.alientechnology.com/tags/index.php>
5. <http://www.wisegEEK.com/what-is-a-supply-chain.htm>
6. ICC Commercial Crime Services. Counterfeiting intelligence bureau (2011), <http://www.icc-ccs.org/home/cib>
7. Ó hEigeartaigh, C.: Pairing computation on hyperelliptic curves of genus 2. PhD thesis, Dublin City University (2006)
8. Ateniese, G., Camenisch, J., Medeiros, B.D.: Untraceable RFID tags via insubvertible encryption. In: CCS, New York, USA, pp. 92–101 (2005)
9. Blass, E.O., Elkhiyaoui, K., Molva, R.: Tracker: Security and privacy for RFID-based supply chains. Cryptology ePrint Archive, Report 2010/219 (2010)
10. Blass, E.O., Elkhiyaoui, K., Molva, R.: Tracker: security and privacy for RFID-based supply chains. In: NDSS, San Diego, California, USA (2011)

11. Boldyreva, A., Gentry, C., O'Neill, A., Yum, D.H.: Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing. In: CCS, New York, NY, USA, pp. 276–285 (2007)
12. Boneh, D., Gentry, C., Lynn, B., Shacham, H.: A survey of two signature aggregation techniques. *CryptoBytes* 6(2) (2003)
13. Boneh, D., Lynn, B., Shacham, H.: Short Signatures from the Weil Pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001)
14. Cai, S., Li, Y., Li, T., Deng, R.H.: Attacks and improvements to an rfid mutual authentication protocol and its extensions. In: WISEC, Zurich, Switzerland, pp. 51–58 (2009)
15. Dolev, D., Yao, A.C.: On the security of public key protocols. Technical report, Stanford, CA, USA (1981)
16. Gattorna, J.: *Living Supply Chains*. Pearson Education (2006)
17. Golle, P., Jakobsson, M., Juels, A., Syverson, P.: Universal re-encryption for mixnets. In: CT-RSA, San Francisco, California, USA, pp. 163–178 (2004)
18. Juels, A.: Minimalist Minimalist Cryptography for Low-Cost RFID Tags (Extended Abstract). In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 149–164. Springer, Heidelberg (2005)
19. Juels, A., Pappu, R., Parno, B.: Unidirectional Key Distribution Across Time and Space with Applications to RFID Security. In: USENIX, San Jose, California, USA, pp. 75–90 (2008)
20. Krawczyk, H., Bellare, M., Canetti, R.: RFC2104 - HMAC:Keyed-Hashing for Message Authentication. RFC Editor (1997)
21. Langheinrich, M., Marti, R.: Practical Minimalist Cryptography for RFID Privacy. *IEEE Systems Journal*, Special Issue on RFID Technology 1(2), 115–128 (2007)
22. Langheinrich, M., Marti, R.: RFID Privacy Using Spatially Distributed Shared Secrets. In: Ichikawa, H., Cho, W.-D., Satoh, I., Youn, H.Y. (eds.) UCS 2007. LNCS, vol. 4836, pp. 1–16. Springer, Heidelberg (2007)
23. Li, Y., Ding, X.: Protecting rfid communications in supply chains. In: ASIACCS, New York, NY, USA, pp. 234–241 (2007)
24. Molnar, D., Wagner, D.: Privacy and Security in Library RFID: Issues, Practices, and Architectures. In: CCS, Washington, DC, USA, pp. 210–219 (2004)
25. Noubir, G., Vijayan, K., Nussbaumer, H.J.: Signature-based method for run-time fault detection in communication protocols. *Computer Communications* 21, 21–25 (1998)
26. Reed, I.S., Solomon, G.: Polynomial codes Over Certain Finite Fields. *Journal of the Society for Industrial and Applied Mathematics* 8(2), 300–304 (1960)
27. Schneier, B.: Description of a new variable-length key, 64-bit block cipher (blowfish). In: Fast Software Encryption, Cambridge Security Workshop, London, UK, pp. 191–204 (1994)
28. Shamir, A.: How to share a secret. *Commun. ACM* 22(11), 612–613 (1979)
29. Song, B., Mitchell, C.J.: RFID Authentication Protocol for Low-cost Tags. In: WISEC, Alexandria, Virginia, USA, pp. 140–147 (2008)
30. Zanetti, D., Fellmann, L., Capkun, S.: Privacy-preserving Clone Detection for RFID-enabled Supply Chains. In: IEEE RFID, Orlando, Florida, USA, pp. 37–44 (2010)

Enhanced Dictionary Based Rainbow Table

Vrizlynn L.L. Thing and Hwei-Ming Ying

Institute for Infocomm Research, Singapore

{vriz,hmying}@i2r.a-star.edu.sg

Abstract. As users become increasingly aware of the need to adopt strong password, it brings challenges to digital forensics investigators due to the password protection of potential evidentiary data. On the other hand, due to human nature and their tendency to select memorable passwords, which compromises security for convenience, users may select strong passwords by considering a permutation of dictionary words. In this paper, we discuss the existing password recovery methods and briefly present our previous work on the design of a time-memory trade-off pre-computed table (Enhanced Rainbow Table) for efficient random password recovery. We then propose the design of an Enhanced Dictionary Based Rainbow Table to integrate the construction of dictionary based permuted passwords and common passwords within the Enhanced Rainbow Table, to incorporate the two promising password recovery approaches. We then present the analysis of the proposed method.

Keywords: digital forensics, password recovery, rainbow table, cryptanalysis.

1 Introduction

Being the most common authentication method, passwords are widely used to protect valuable data and to ensure a secured access to systems/machines. However, the use of password protection presents a challenge for investigators while conducting digital forensics examinations.

In some cases, compelling a suspect to surrender his password would force him to produce evidence that could be used to incriminate him, thereby violating his right against self-incrimination. Therefore, this presents a problem for the authorities. It is then necessary to have the capability to access a suspects data without expecting his assistance.

While there exist methods to decode hashes to reveal passwords used to protect potential evidence, lengthier passwords with larger characters sets have been encouraged to thwart password recovery. Awareness of the need to use stronger passwords and active adoption have also rendered many existing password recovery tools inefficient or even ineffective.

The more common methods of password recovery techniques are based on brute force, dictionary attack, breaking hashing algorithms and rainbow tables.

In the brute force attack, every possible combination of the password characters in the password space is attempted for a match search. It is an extremely time consuming process. However, due to its exhaustive generation and search, the password will be recovered eventually if sufficient time is given. Cain and Abel (Cain and Abel, 2011), John the Ripper (John The Ripper, 2011) and LCP (LCPSoft, 2011) are popular tools that support brute force attacks.

The dictionary attack method involves loading a file of dictionary words (and performing permutation optionally) into a password recovery tool to search for a match of their hash values with the stored one. If the password is not a dictionary or permuted dictionary word, the recovery would fail.

Research attempting to discover and identify the weaknesses of hashing algorithms have also been useful in passwords or encryption keys recovery. This method is based on the collision of hashes in specific hashing algorithms (Contini, 2006; Fouque, 2007; Sasaki, 2007; Sasaki, 2008). However, they are highly complex and time consuming for use during time-critical forensics investigations. The methods are only applicable to specific hashing algorithms.

The time-memory tradeoff method (Hellman, 1980) is a hybrid of brute force attack and precomputed tables. A large number of passwords are repeatedly hashed and reduced to form password chains. Only the head and tail of these chains are stored. During recovery, the password hash goes through a series of reduction and hashing until a match with one of the stored tails is found. Passwords encrypted with hashing algorithms such as LM or NTLM used for Windows login (Todorov, 2007), MD5 (Rivest, 1992), SHA-2 (NIST, 2002) and RIPEMD-160 (Dobbertin, 1996) are susceptible to this recovery method.

The rainbow table method (Oechslin, 2003; Thing, 2009; Weir 2009; Ying, 2011) is similar to, and falls under the class of time-memory tradeoff method. The difference is that different reduction functions are used at each step of the chain generation, so as to minimise the collision of merging chains. Therefore, the success rate of password recovery can be higher.

In this paper, we first present our time-memory tradeoff rainbow table method, the Enhanced Rainbow Table, which shows promising performance and results (that is, in terms of success rate and recovery speed) during password recovery compared to existing work. We then propose the design of a novel Enhanced Dictionary Based Rainbow Table by including the generation of permuted dictionary words in the algorithm. We then analyse the proposed method.

2 Background

The idea of a general time-memory tradeoff was first proposed by Hellman in 1980 (Hellman, 1980). In the context of password recovery, we describe the Hellman algorithm as follows.

We let X be the plaintext password and Y be the corresponding stored hash value of X . Given Y , we need to find X , which satisfies $h(X) = Y$, where h is a known hash function. However, finding $X = h^{-1}(Y)$ is feasibly impossible since hashes are computed using one-way functions, where the reversal function,

h^{-1} , is unknown. Hellman suggested taking the plaintext values and applying alternate hashing and reducing, to generate a pre-computed table.

For example, the corresponding 128-bit hash value for a 7-character password (composed from a character set of English alphabets), is obtained by performing the password hashing function on the password. With a reduction function such as $H \bmod 26^7$, where H is the hash value converted to its decimal form, the resulting values are distributed in a best-effort uniform manner. For example, if we start with the initial plaintext value of “abcdefg” and upon hashing, we get a binary output of 0000000....000010000000....01, which is 64 ‘0’s and a ‘1’ followed by 62 ‘0’s and a ‘1’. $H = 2^{63} + 1 = 9223372036854775809$. The reduction function will then convert this value to “3665127553”, which corresponds to a plaintext representation “lwmkgij”, computed from $(11(26^6) + 22(26^5) + 12(26^4) + 10(26^3) + 6(26^2) + 8(26^1) + 9(26^0))$.

After a pre-defined number of rounds of hashing and reducing, only the initial and final plaintext values (i.e. “head” and “tail” of the chains) are stored. Using different initial plaintexts, the hashing and reducing operations are repeated, to generate a larger table (of increasing rows/chains). A larger table will theoretically contain more pre-computed values (i.e., disregarding hash collisions), thereby increasing the success rate of password recovery, while taking up more storage space. The pre-defined number of rounds of hashing and reducing will also increase the success rate by increasing the length of the “virtual” chain, while bringing about a higher computational overhead.

To recover a plaintext from a given hash, a reduction operation is performed on the hash and a search for the computed plaintext among the final values in the table is conducted. If a match is not found, the hashing, reducing and searching operations are repeated. The maximum possible rounds of operations is determined by the chain length. If the hash value is found in a particular chain, the values in the chain are then worked out by performing the hashing and reducing functions to arrive at the plaintext giving the specific hash value.

Unfortunately, there is a likelihood that chains with different initial values may merge due to collisions. These merges will reduce the number of distinct hash values in the chains and diminish the recovery success rate. The success rate can be increased by using multiple tables with each table using a different reduction function. If we let $P(t)$ be the success rate of using t tables, then $P(t) = 1 - (1 - P(1))^t$, which is an increasing function of t since $P(1)$ is between 0 and 1. Hence, introducing more tables increase the success rate but also cause an increase in the computational complexity and storage space.

In (Denning, 1982), Rivest suggested a method of using distinguished points as end points for chains. Distinguished points are keys, which satisfy a given criteria, e.g., the first or last q bits are all 0. The chains are not generated with a fixed length but they terminate upon reaching pre-defined distinguished points. This method decreases the number of memory lookups compared to Hellman’s method and is capable of loop detection. If a distinguished point is not obtained after a finite number of operations, the chain is suspected to contain a loop and is discarded. Therefore, the generated chains are free of loops. One

limitation is that the chains will merge if there is a collision within the same table. The variable lengths of the chains will also result in an increase in false alarms. Additional computations are incurred to detect false alarm occurrences.

(Oechslin, 2003) introduced a new table structure to reduce the probability of merging occurrences. The rainbow chains use multiple reduction functions so merges occur only if collisions happen at the same positions in different chains. Oechslin showed that the coverage in a single rainbow table is 78.8% compared to 75.8% in the classical tables of Hellman (with distinguished points).

(Weir, 2009) integrated dictionary attacks with the original rainbow table (Oechslin, 2003) to generate virtual chains of passwords consisting of dictionary words. This method improved the efficiency of dictionary attacks by utilizing the rainbow table. However, the table does not contain randomly generated passwords and can only be used for dictionary password recovery.

3 Enhanced Dictionary Based Rainbow Table

The key objectives in enhancing password recovery is to meet the increasing challenges of strong password-protected evidentiary data. Utilizing the hybrid approach of the brute force technique and precomputed table approach proves to be a cost-efficient way to recover password. Therefore, to improve password recovery performance, further research to increase the success rate and reducing the recovery time by rainbow tables is needed while taking into consideration stronger passwords adopted by common users.

On the other hand, it has been shown that humans are tempted to choose passwords which are easy for them to remember (Google News, 2009; Narayanan, 2005). Such passwords could be based on a combination of common key sequences on the keyboard layout, dictionary words, and a combination of dictionary words. Another common approach to strengthen the passwords while maintain their memorability is to include numbers and special characters in the passwords. Therefore, by taking into consideration the human nature and their tendency in password selections, and incorporating such knowledge into the design of a new password recovery method would improve performance significantly.

In the following sub-sections, we briefly describe our recent work on the enhancement of rainbow tables (Thing, 2009; Ying, 2011). Next, we propose a new approach of integrating memorable passwords with the Enhanced Rainbow Tables by considering the unique features of these new tables. We then present the analysis of this new Enhanced Dictionary Based Rainbow Table method.

3.1 Enhanced Rainbow Table

In (Thing, 2009; Ying, 2011), we proposed an Enhanced Rainbow Table design with a novel sorting algorithm. The first novelty lies in the chains generation technique. Instead of taking a large set of plaintexts as the initial values, we systematically choose a much smaller unique set. We choose a plaintext and compute its corresponding hash value. We let the resulting hash value be H .

Following that, we compute $(H+1) \bmod 2^j$, $(H+2) \bmod 2^j, \dots, (H+k) \bmod 2^j$ for a variable k , where j is the number of bits of the hash output value (e.g. in MD5 hash, $j = 128$). These hash values are the branches of the above chosen initial plaintext. We then apply alternate hashing and reducing operations to all these branches. The resulting extended chain of branches is a block. Only the final values of the plaintexts in each block are stored with one initial plaintext value, instead of storing all the final values with the corresponding initial values in the original rainbow table, resulting in significant storage space conservation (or success rate improvement if the same storage space is provided).

As the “tail” passwords cannot be sorted now, since in doing so, the information of its corresponding initial hash value (which is not stored) will be lost, a novel sorting algorithm (Ying, 2011) was proposed so that the password lookup in the stored tables can be optimized. The use of special characters which are the non-printable ASCII characters, was proposed for the Enhanced Rainbow Table sorting. There are a total of 161 such characters and we assume that these non-printable ASCII characters do not form any of the character set of the passwords since they are not found on the keyboard. We insert a number of these special characters into the stored “tail” passwords. The way in which these special characters are inserted provides information on the original position of the passwords after the table has been sorted. The consequence is that these inserted special characters will incur storage space. We illustrated in (Ying, 2011) that the increase in storage space is minimal and is also significantly lesser than the original rainbow tables storage requirement. The advantage of this sorting algorithm is that the passwords in the table can now be sorted and thus a password lookup can be optimized. The sufficiency of the available special characters for use in sorting, the storage requirements, and the success rate of password recovery were also evaluated in (Ying, 2011). Maintaining the same storage space requirements for both methods, the Enhanced Rainbow Table is able to achieve an improvement of up to 26.13% and 23.60%, for the recovery of alpha-numeric passwords and passwords containing any of the printable ASCII characters, respectively, over the original rainbow table.

Next, we propose the integration of permutated dictionary attack within the Enhanced Rainbow Table to take into consideration the human tendency to select memorable passwords.

3.2 Design of Dictionary Based Enhanced Rainbow Table

In the Enhanced Rainbow Table, passwords were generated based on the hashing and reduction functions. Therefore, the generation is very random and a large percentage of passwords may contain special characters at random places and non-dictionary words. Such passwords have a very low memorability level and users who chose such passwords usually create the passwords using strong password creation tools. Most users also need to note down the passwords and store them separately to prevent them from forgetting their own passwords for subsequent accesses. However, users tend to want to avoid the trouble of choosing a password of such high complexity and worry about forgetting it later. Therefore,

they usually try to choose passwords which are memorable. These passwords are usually dictionary words, common keyboard layout key sequences or information related to themselves (for example, their name, spouse's name or birthdays).

A simple approach is to conduct a dictionary attack first and then the rainbow table password recovery if the dictionary attack fails. However, both the computational overhead and storage requirement will be high. Instead, in this paper, we propose a novel approach to incorporate common passwords into the Enhanced Rainbow Table so that the percentage of common passwords can be higher resulting in a higher success rate even in the scenario whereby the length of the passwords is large and the storage capacity is limited. The aim of the integration of permuted dictionary and the Enhanced Rainbow Table is to construct the table where by it can contain as many permuted dictionary words as possible (in both stored elements and the virtual chains).

The simplest method to create the Enhanced Dictionary Based Rainbow Table is to generate permuted dictionary words as the initial column of passwords. However, this is only applicable for the case of the original rainbow table as the initial column is discarded in the Enhanced Rainbow Table.

Instead, we propose to have the first reduction function generate permuted dictionary words in the first virtual column, which in the Enhanced Rainbow Table, will be recoverable. However, in this case, the possible number of "initially generated" permuted dictionary words is limited by the number of chains in the table. The recovery speed may also suffer for such passwords as they fall under the first virtual column. Therefore, next, we additionally propose constructing some chains where all the entries are common passwords.

Suppose we identify x number of common passwords used. We want to ensure that these x passwords can be generated from a rainbow chain. One way to do this is starting with any password, hash and choose an appropriate reduction function R_1 which reduces to one of the x number of words in the list. Continuing with the chain generation, hash this resultant and choose an appropriate reduction function R_2 which again reduces to another one of the words in the list. Continue doing this until all the words in the list is generated in the chain as required. As long as x is not too large relative to the keyspace, we will always almost certainly be able to choose such R_i and thus, generate such a chain which contains all the words in the list. As for the remaining chains, there is no certainty that dictionary words can be generated accordingly in the same manner since the outputs of hash functions tend to be random. The advantage is that this method will be able to recover both common and random passwords.

For example, suppose we want to consider 7-character passwords (consisting of lower case alphabets, hashed by MD5). Given that "letmein", "abcdefg" and "testing" are three common 7-character passwords, the goal is to include them in the rainbow chains.

Starting with the password "testing", upon hashing, its hashed value is $H_1 = \text{ae2b1fca515949e5d54fb22b8ed95575}$. This value is then converted to its decimal representation and the reduction function is applied where $r_1(H_1) = H_1 + 4938209469 \bmod 26^7$. Converting $r_1(H_1)$ back to its password representation

results in the password “letmein”. The next step is to hash “letmein”. This results in $H_2 = 0d107d09f5bbe40cade3de5c71e9e9b7$. Then, apply a reduction function r_2 to H_2 where $r_2(H_2) = H_2 + 3129034064 \bmod 26^7$. Converting $r_2(H_2)$ back to the password representation will result in the password “abcdefg”.

Hence, this initial rainbow chain consists of the above three passwords.

Proposition 1 : Any given 3 passwords can be recovered regardless of the size of keyspace and the hash applied.

Proof : Let size of keyspace = n . Let the hash function be denoted by h . Then, for simplification, let $=$ to mean $\equiv \bmod n$ for the subsequent parts of the proof. To prove the proposition, we show that there exists at least one arrangement to insert these 3 passwords such their corresponding reduction functions are distinct. Let the 3 passwords be p_1, p_2 and p_3 . Let $h(p_1) = a_1, h(p_2) = a_2$ and $h(p_3) = a_3$. Suppose for all 6 arrangements, each arrangement results in having identical reduction functions. Thus, we obtain, the following set of equations:

- (1) $p_2 - a_1 = p_3 - a_2$
- (2) $p_3 - a_1 = p_2 - a_3$
- (3) $p_1 - a_2 = p_3 - a_1$
- (4) $p_3 - a_2 = p_1 - a_3$
- (5) $p_1 - a_3 = p_2 - a_1$
- (6) $p_2 - a_3 = p_1 - a_2$

Comparing equations 1 and 3, we get $p_1 + p_2 = 2p_3$

Comparing equations 2 and 5, we get $p_1 + p_3 = 2p_2$

Comparing equations 4 and 6, we get $p_3 + p_2 = 2p_1$

Solving these 3 new equations, we obtain $p_1 = p_2 = p_3$. This is a contradiction since p_1, p_2, p_3 are distinct modulo n ; thus proving Proposition 1.

Proposition 2 : Any given 4 passwords can be recovered regardless of the size of keyspace and the hash applied.

Proof : Let size of keyspace = n . Let the hash function be denoted by h . Again, for simplification, let $=$ to mean $\equiv \bmod n$ for the subsequent parts of the proof. Let the 4 passwords be p_1, p_2, p_3 and p_4 and let $h(p_1) = a_1, h(p_2) = a_2, h(p_3) = a_3$ and $h(p_4) = a_4$.

Consider the 3 passwords p_1, p_2, p_3 which are placed in the first 3 entries of the chain. Applying Proposition 1, there exists an arrangement which will result in distinct reduction functions. Without loss of generality, assume that the first 3 passwords in the chain are p_1, p_2, p_3 in that order such that the corresponding reduction functions are distinct. Then, p_4 will be in the 4th entry of the chain. Suppose $p_4 - a_3 \neq p_2 - a_1$ and $p_4 - a_3 \neq p_3 - a_2$. Then $p_1p_2p_3p_4$ is the desired order to place the passwords which ensures distinct reduction functions.

Suppose either $p_4 - a_3 = p_2 - a_1$ or $p_4 - a_3 = p_3 - a_2$.

Case 1 : $p_4 - a_3 = p_2 - a_1$

Consider the arrangement $p_4p_1p_2p_3$. If $p_1 - a_4 \neq p_2 - a_1$ and $p_1 - a_4 \neq p_3 - a_2$, we are done. Otherwise, either $p_1 - a_4 = p_2 - a_1$ or $p_1 - a_4 = p_3 - a_2$.

Case 1(a) : $p_1 - a_4 = p_2 - a_1 = p_4 - a_3$.

Consider the arrangement $p_1p_3p_4p_2$. Suppose not all the reduction functions are distinct, then $p_3 - a_1 = p_2 - a_4$. Next, consider the arrangement $p_4p_1p_3p_2$. If not all the reduction functions are distinct, then $p_3 - a_1 = p_2 - a_3$. This implies $a_4 = a_3$ and thus $p_1 = p_4$ which is a contradiction.

Case 1(b) : $p_1 - a_4 = p_3 - a_2$ and $p_4 - a_3 = p_2 - a_1$

If $p_1 - a_4 = p_1 - a_3$, then $a_3 = a_4$. Thus, $p_4p_3p_1p_2$ will be the desired arrangement. If $a_3 \neq a_4$, consider the arrangements $p_4p_2p_3p_1$, $p_1p_3p_4p_2$, $p_4p_1p_3p_2$, $p_2p_4p_1p_3$ and $p_1p_4p_2p_3$ in the order as stated. Suppose none of these arrangements result in distinct reduction functions.

By considering $p_4p_2p_3p_1$, we get $p_2 - a_4 = p_1 - a_3$.

By considering $p_1p_3p_4p_2$, we get $p_1 - a_3 = p_2 - a_4 = p_3 - a_1$.

By considering $p_4p_1p_3p_2$, we get $p_1 - a_4 = p_2 - a_3$.

By considering $p_2p_4p_1p_3$, we get $p_4 - a_2 = p_3 - a_1$.

By considering $p_1p_4p_2p_3$, we get $p_4 - a_1 = p_3 - a_2$.

From the above arrangement $p_2p_4p_1p_3$, we obtain $p_3 - a_1 = p_4 - a_2$ and from arrangement $p_1p_4p_2p_3$, we obtain $p_4 - a_1 = p_3 - a_2$. Hence, $p_3 = p_4$, which is a contradiction.

Case 2 : $p_4 - a_3 = p_3 - a_2$ and $p_4 - a_3 \neq p_2 - a_1$

Consider the arrangement $p_3p_4p_1p_2$. If $p_4 - a_3 \neq p_1 - a_4$ and $p_2 - a_1 \neq p_1 - a_4$, we are done. Otherwise, either $p_4 - a_3 = p_1 - a_4$ or $p_1 - a_4 = p_2 - a_1$.

Case 2(a) : $p_4 - a_3 = p_3 - a_2 = p_1 - a_4$

Consider the arrangement $p_3p_4p_2p_1$. If this arrangement results in distinct reduction functions, then $p_2 - a_4 = p_1 - a_2$. Next, consider arrangement $p_4p_2p_3p_1$. If this arrangement does not result in distinct reduction functions again, then we must have $p_2 - a_4 = p_1 - a_3$. Hence, $a_2 = a_3$ which implies $p_3 = p_4$, which is a contradiction.

Case 2(b) : $p_1 - a_4 = p_2 - a_1$ and $p_4 - a_3 = p_3 - a_2$

If $a_1 = a_3$, consider the arrangement $p_3p_2p_1p_4$. If this is not the desired arrangement, then $a_2 = a_4$. Hence, $p_2p_1p_3p_4$ will be the desired arrangement.

If $a_1 \neq a_3$, consider the arrangements $p_4p_1p_3p_2$, $p_2p_1p_3p_4$, $p_3p_2p_4p_1$ and $p_2p_3p_1p_4$ in this order. Suppose none of these arrangements result in distinct reduction functions, then we obtain the following set of equations: $p_3 - a_1 = p_2 - a_3 = p_1 - a_2$, $p_4 - a_2 = p_1 - a_4 = p_2 - a_1$ and $p_1 - a_3 = p_4 - a_1$. Simplifying, we obtain $p_1 - p_4 = p_3 - p_2$ and $p_2 - p_1 = p_3 - p_4$. Thus, $p_2 = p_3$, which is a contradiction.

This proves Proposition 2.

3.3 Methods of Constructing Chains

We propose 2 methods of constructing chains such that they include the desired passwords. We then provide an analysis of both methods in terms of feasibility and the expected computational attempts required.

Method 1 : Compute all the possible values of $p_i - a_j$. Then, consider all possible chains that can be formed. For each chain, test if the chain results in distinct reduction functions. If so, we have found the required chain; otherwise continue testing the remaining ones until we find such a chain.

Method 2 : Compute all the possible values of $p_i - a_j$. Take the one which has a lowest occurrence frequency. That link will be the part of the generated chain. For the subsequent links, we choose the ones which are distinct from all previous ones in the chain and occur at a lower frequency. This step is repeated until we have the desired chain or we reach a point where we are unable to add any more links. In the latter case, we backtrack to the previous process and select another link instead, till the desired chain is obtained.

Example

Suppose we want to include 3 passwords p_1 , p_2 and p_3 with the following password (plaintext) and hash values in the chain:

$$p_1 = 10, p_2 = 20, p_3 = 30, h(p_1) = a_1 = 29, h(p_2) = a_2 = 9, h(p_3) = a_3 = 19$$

Then, $p_1 - a_2 = 1$, $p_3 - a_2 = 21$, $p_2 - a_1 = -9$, $p_3 - a_1 = 1$, $p_1 - a_3 = -9$, $p_2 - a_3 = 1$

Hence, only the chains $p_1p_2p_3$ and $p_2p_3p_1$ are the desired ones.

Applying Method 1,

Probability of getting a desired chain in 1 attempt = $1/3$

Probability of getting the chain in 2 attempts = $4/6 \times 2/5 = 4/15$

Probability of getting the chain in 3 attempts = $4/6 \times 3/5 \times 2/4 = 1/5$

Probability of getting the chain in 4 attempts = $4/6 \times 3/5 \times 2/4 \times 2/3 = 2/15$

Probability of getting the chain in 5 attempts = $4/6 \times 3/5 \times 2/4 \times 1/3 = 1/15$

Therefore, expected number of attempts to get a desired chain

$$= 1 \times 1/3 + 2 \times 4/15 + 3 \times 1/5 + 4 \times 2/15 + 5 \times 1/15$$

$$= 7/3$$

Applying Method 2, since $p_3 - a_2$ occurs with the least frequency, we start the construction of the chain with p_2p_3 . We are left with 2 possible links; either $p_1 - a_3$ or $p_2 - a_1$. Since both are distinct from the previous one, we select a link according to its occurrence frequency. In this case, both have the same frequency. Therefore, we can select either; e.g. $p_1 - a_3$. Thus, the generated chain is $p_2p_3p_1$.

Consider a general case of inserting n recoverable passwords. Suppose after computing all $n(n-1)$ values of $p_i - a_j$, we have the following relations :

- 1) $p_1 - a_2 = p_2 - a_3 = p_3 - a_4 = \dots\dots\dots = p_{n-1} - a_n = p_n - a_1$
- 2) the other $n(n-2)$ expressions of $p_i - a_j$ are all mutually distinct.

First, we need to compute the number of chains such that not all its reduction functions are distinct.

Number of chains such that some part of the chain contain $p_{i+2}p_{i+1}p_i$ or $p_2p_1p_n$ or $p_1p_np_{n-1}$
 $= n(n-2)!$

Number of chains such that some part of the chain contain $p_{i+1}p_i$ and $p_{j+1}p_j$
 $= (n-2)! \binom{n}{2} - n]$
 $= (n-2)! \frac{n(n-3)}{2}$

Total number of chains such that not all its reduction functions are distinct
 $= n(n-2)! + (n-2)! \frac{n(n-3)}{2}$
 $= \frac{n!}{2}$

Then, total number of chains such that all of its reduction functions are distinct

$= n! - \frac{n!}{2}$
 $= \frac{n!}{2}$

Probability of getting a valid chain after k attempts

= Probability of getting the invalid chains for the first $k-1$ attempts and getting a valid chain on the k^{th} attempt

$= \frac{n!/2}{n!} \times \frac{n!/2-1}{n!-1} \times \frac{n!/2-3}{n!-3} \times \dots\dots\dots \times \frac{n!/2-(k-2)}{n!-(k-2)} \times \frac{n!/2}{n!-(k-1)}$
 $= [\frac{n!-(k-1)}{n!/2-(k-1)} / \frac{n!}{n!/2}] \times \frac{n!/2}{n!-(k-1)}$

Expected number of attempts required

$= [\frac{n!}{2} / \binom{n!}{n!/2}] \sum_{k=1}^{n!/2+1} \binom{n!-(k-1)}{n!/2} \frac{k}{n!-(k-1)}$
 $= \sum_{k=1}^{n!/2+1} \binom{n!-k}{n!/2-1} k / \binom{n!}{n!/2}$

We consider the asymptotic value of n . Let $z = \frac{n!}{2}$

Then, the expected number of attempts required can be rewritten as

$= \sum_{k=1}^{z+1} \binom{2z-k}{z-1} k / \binom{2z}{z}$
 $= \frac{z!}{2(2z-1)!} \sum_{k=1}^{z+1} k(2z-k)(2z-k-1)\dots\dots(z-k+2)$

As n increases, z increases. $\frac{z!}{2(2z-1)!} \sum_{k=1}^{z+1} k(2z-k)(2z-k-1)\dots\dots(z-k+2)$ approaches

$\sum_{k=1}^{\infty} \frac{k}{2^k}$. However, this is expected of the geometric distribution with parameter $\frac{1}{2}$. Hence, $\sum_{k=1}^{\infty} \frac{k}{2^k} = 2$ and $\frac{z!}{2(2z-1)!} \sum_{k=1}^{z+1} k(2z-k)(2z-k-1)\dots\dots(z-k+2)$ approaches

2 as z gets large. This in turn implies that $\sum_{k=1}^{n!/2+1} \binom{n!-k}{n!/2-1} k / \binom{n!}{n!/2}$ approaches

2 as n gets large. Hence, for Method 1, we can deduce that for large n , the expected number of attempts required is close to 2.

For Method 2, we first select a link that occurs with the least frequency, e.g. $p_2 - a_1$. Then, we build the chain from this initial link and we get $p_1p_2p_3$ and so on until we arrive at $p_1p_2p_3\dots\dots\dots p_n$, which is one of the desired chains.

4 Conclusion

In this paper, we presented the novel design of an Enhanced Dictionary Based Rainbow Table. We then proposed two new methods of chains construction. We analysed and proved the feasibility of the proposed methods. We also analysed the probability of generating the desired chains in specific scenarios of different password space sizes and in the generic case of n password space, and expected computational attempts required using each method. The analysis results showed that the proposed Enhanced Dictionary Based Rainbow Table method is a promising new approach to efficiently recover passwords by taking into consideration both the use of common passwords (human memorable) and randomly generated passwords at the same time.

References


1. Cain and Abel: Password recovery tool (2011), <http://www.oxid.it> (retrieved December 2011)
2. Contini, S., Yin, Y.L.: Forgery and Partial Key-Recovery Attacks on HMAC and NMAC Using Hash Collisions. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 37–53. Springer, Heidelberg (2006)
3. Denning, D.E.R.: Cryptography and Data Security. Addison-Wesley Publication (1982)
4. Dobbertin, H., Bosselaers, A., Preneel, B.: Ripemd-160: A Strengthened version of RIPEMD. In: Gollmann, D. (ed.) FSE 1996. LNCS, vol. 1039, pp. 71–82. Springer, Heidelberg (1996)
5. Fouque, P.-A., Leurent, G., Nguyen, P.Q.: Full Key-Recovery Attacks on HMAC/NMAC-MD4 and NMAC-MD5. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 13–30. Springer, Heidelberg (2007)
6. Google News (2009), Favorite passwords: '1234' and 'password', <http://www.google.com/hostednews/afp/article/ALeqM5jeUc6Bb1nd0M19WVQWvjs6D2puvw> (retrieved December, 2011)
7. Hellman, M.E.: A cryptanalytic time-memory trade-off. IEEE Transactions on Information Theory IT-26(4), 401–406 (1980)
8. John The Ripper, Password cracker (2011), <http://www.openwall.com> (retrieved December 2011)
9. LCPSoft, Lcpsoft programs (2011), <http://www.lcpsoft.com> (retrieved December 2011)
10. Narayanan, A., Shmatikov, V.: Fast dictionary attacks on passwords using time-space tradeoff. In: ACM Conference on Computer and Communications Security, pp. 364–372 (2005)

11. National Institute of Standards and Technology, NIST (2002), Secure hash standard. Federal Information Processing Standards Publication 180(2)
12. Oechslin, P.: Making a Faster Cryptanalytic Time-Memory Trade-Off. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 617–630. Springer, Heidelberg (2003)
13. Rivest, R.: The MD5 message-digest algorithm. IETF RFC 1321 (1992)
14. Sasaki, Y., Yamamoto, G., Aoki, K.: Practical password recovery on an MD5 challenge and response. Cryptology ePrint Archive, Report 2007/101 (2008)
15. Sasaki, Y., Wang, L., Ohta, K., Kunihiro, N.: Security of MD5 Challenge and Response: Extension of APOP Password Recovery Attack. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 1–18. Springer, Heidelberg (2008)
16. Smyth, S.M.: Searches of computers and computer data at the United States border: The need for a new framework following United States V. Arnold. *Journal of Law, Technology and Policy* (1), 69–105 (2009)
17. Thing, V.L.L., Ying, H.M.: A novel time-memory trade-off method for password recovery. *Digital Investigation, International Journal of Digital Forensics and Incident Response* 6(suppl.), S114–S120 (2009)
18. Todorov, D.: *Mechanics of user identification and authentication: Fundamentals of identity management*. Auerbach Publications, Taylor and Francis Group (2007)
19. Ying, H.M., Thing, V.L.L.: A novel rainbow table sorting method. In: *International Conference on Technical and Legal Aspects of the e-Society (CYBERLAWS)* (2011)
20. Weir, M.: *Enough with the Insanity: Dictionary Based Rainbow Tables*. ShmooCon (2009)

Authorization Policies for Materialized Views

Sarah Nait-Bahloul, Emmanuel Coquery, and Mohand-Saïd Hacid

Université de Lyon
Université Claude Bernard Lyon 1 LIRIS CNRS UMR 5205
43, bd du 11 novembre 1918
69622 Villeurbanne cedex, France
{sarah.nait-bahloul,emmanuel.coquery,mshacid}@liris.cnrs.fr

Abstract. In this paper, we propose a novel approach to facilitate the administration of access control policies to ensure the confidentiality of data at the level of materialized views. A materialized view stores both the definition of the view and the rows resulting from the execution of the view. Several techniques and models have been proposed to control access to databases, but to our knowledge the problem of automatically generating from access control policies defined over base relations the access control policies that are needed to control materialized views is not investigated so far. We are dealing with this problem by resorting to an adaptation of query rewriting techniques. We choose to express fine-grained access control through authorization views .

Keywords: Database Security, Access Control, Authorization Views, Materialized Views, Datalog.

1 Introduction

In the area of data management, the problem of access control was one of the most sensitive issues. Several techniques and models have been proposed to improve data security and to ensure data confidentiality (see, among others, [\[10\]](#) [\[7\]](#)).

With the use of large systems like Data Warehouses [\[15\]](#) or Distributed Database Systems [\[3\]](#), new security issues arise [\[9\]](#) [\[13\]](#). In our work, we focus on the problem of securing materialized views. A lot of organizations use relational DBMS to store and manage their information and very often they resort to materialized views. A materialized view records the results returned by the query into a physical table. In many settings, the views are materialized in order to optimize access. For instance, in Data Warehouses, they can be used to precompute and store complex aggregations. Thus, the user can use the materialized view as any other base relation. In this context, ensuring security at the materialized view level is as important as ensuring security at the level of tables. The question is then *how to ensure data security at the level of a materialized view?* So far,

¹ This work is partially supported by the Rhône-Alpes Region, Cluster ISLE (Informatique, Signal, Logiciel Embarqué).

access control rules on materialized views are defined manually by an administrator by trying to comply with the basic ones. In a system containing tens or hundreds of tables controlled by tens or hundreds of rules, it becomes impossible for administrators to deal with such large sets of rules and consider all the relevant ones.

Based on our previous work [6], we propose a novel approach that compute new access rules based on existing access rules over base relations. In our approach, we consider fine-grained authorization policies that are defined and enforced in the database through authorization views [10]. Figure 1 summarizes our approach. The idea is the following: Given a set of base relations (with the corresponding authorizations) and a set of materialized view definitions, synthesize a set of authorization views that will be attached to the materialized views, such that querying the materialized views through those authorization views does not deliver more information than querying the database through the original ones. In this paper, authorization and materialized views are restricted to conjunctive queries.

The rest of the paper is organized as follows: Section 2 discusses authorization views. Section 3 presents our approach. In section 4, we present the related work. We conclude in section 5.

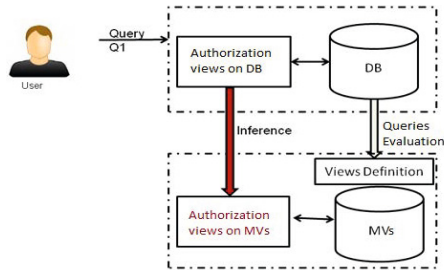


Fig. 1. Security policies for materialized views: System architecture

2 Datalog for Authorization

Among the several proposed techniques and models to ensure data confidentiality, we are particularly interested by the "Authorization views" [10]. They are a well known database technique that provides content-based and fine-grained access control. Authorization views are logical tables that specify exactly the accessible data, either drawn from a single table or from multiple tables.

The goal of our work is to automatically determine the set of "authorization views" that will be attached to materialized views. Doing so, the user can query her/his appropriate authorization views or (s)he can query the materialized views and the system will rewrite the query using the authorization views. Thus, in our proposal, we are independent of the way the materialized views are accessed. Nevertheless, to facilitate the understanding of our approach and without loss of generality, we assume that the user can query only the authorization views.

As we restrict our self to conjunctive queries, we use non recursive Datalog without negation [11] as a formal framework for expressing access control rules. For example, the following rule defines an authorization view on the Doctor table:

$$av_1(IdD, Dname, Dfname, Dspecialty) \leftarrow \\ doctor (IdD, Dname, Dfname, Dadr, Dphone, Dspecialty, Dsalary).$$

The user has right to view the last name, the first name and the speciality of doctors in the hospital, but not personal nor salary information.

3 Contribution

In this section we present our algorithm *ACMV* (Access Control to Materialized Views) that generates a set of authorization views \mathcal{AVMV} that should be attached to and defined on materialized views. The algorithm takes as input two sets: (1) A set of views (\mathcal{AV}), representing the authorization views defined on the base tables and (2) a set of views (\mathcal{MV}), representing the definitions of materialized views.

The generated views should be *secure*. Secure means that the generated views should not give access to information that are no allowed by the basic authorization views. We have to guarantee that for each query on \mathcal{AVMV} , there exists an equivalent query on \mathcal{AV} .

In order to automatically generate the relevant set of authorization views \mathcal{AVMV} , we propose a novel approach based on query rewriting techniques. To ensure the security property, we propose an algorithm that performs a double rewriting, using the two sets of views (\mathcal{AV} and \mathcal{MV}).

We first describe the core of our algorithm which is based on query rewriting technique, namely MiniCon [8]. We propose an adaptation of this algorithm to the security context.

S-MiniCon: An Adaptation of the MiniCon Algorithm to the Security Context

The MiniCon algorithm [8] was initially proposed as an efficient method for answering queries using views. It takes as input a query Q and a set of views V and calculate all possible rewritings of Q using views in V , such that, for each rewriting rw , we have $rw \subseteq Q$.

Let us take a simple example to show why we have to adapt the MiniCon algorithm to the security context. Assume the query (1) shown below, which makes a copy of the table *patient*(*IdP*, *Name*, *Disease*). The authorization view (2) specifies an authorization access to the tuples (*IdP*, *Name*) of the *patient* relation.

$$q(IdP, Name, Disease) \leftarrow patient(IdP, Name, Disease). \tag{1}$$

$$av(IdP, Name) \leftarrow patient(IdP, Name, Disease). \tag{2}$$

We propose to rewrite the query q using the authorization view av in order to determine which set of tuples in q is accessible given av . If we apply the original MiniCon algorithm, the authorization view av will be considered as irrelevant. The condition regarding the head variables is not satisfied [8]. But, if we do not take the authorization view as relevant, no rewriting will be generated, i.e. no tuple in q is accessible. It is too restrictive, since one can have access to the tuples $(IdP, Name)$ by projecting them out. Therefore, in our framework, we propose to adapt the MiniCon algorithm by relaxing the condition on the head variables.

ACMV Algorithm: Effectively and Efficiently Apply the S-MiniCon Algorithm

In this section, we present our proposal that exploits a double rewriting. The algorithm takes as input a set of views \mathcal{Q} to rewrite and the two sets of views \mathcal{AV} and \mathcal{MV} . For the first iteration, we define the set of views \mathcal{Q} that specify a full access to \mathcal{MV} . The algorithm starts by rewriting each q_i of \mathcal{Q} using \mathcal{AV} , the result is a set of rewritings \mathcal{RW}_{q_i} . Let $\mathcal{RW}_{q_i} = \{rw_1, \dots, rw_n\}$. This first step determines which set of tuples of q_i is accessible from \mathcal{AV} . The second step consists in checking if this set is also accessible from \mathcal{MV} . For this, the algorithm rewrites each rw_j of \mathcal{RW}_{q_i} using \mathcal{MV} . We note \mathcal{RW}_{rw_j} , the rewritings generated with this second rewriting.

Algorithm 1. Double rewriting

Input: q the view to rewrite
 \mathcal{AV} : Set of authorization views
 \mathcal{MV} : Set of materialized views

Output: \mathcal{RW} : Set of Rewritings

$q^{exp} = expansion(q)$
 $\mathcal{RW}_q = S\text{-MiniCon}(q^{exp}, \mathcal{AV})$
foreach rewriting rw_j of \mathcal{RW}_q **do**
 $rw_j^{exp} = expansion(rw_j)$
 $\mathcal{RW}_{rw_j} = S\text{-MiniCon}(rw_j^{exp}, \mathcal{MV})$
 add \mathcal{RW}_{rw_j} to \mathcal{RW}
end
return \mathcal{RW}

In order to ensure the equivalence of the security property, the algorithm must check whether the application of the double rewriting using \mathcal{AV} and \mathcal{MV} has filtered tuples of \mathcal{Q} . For this, we propose to check, for each q_i , if the union of the generated rewritings contains q_i . We rely on subsumption [2] algorithm for this check. We recall here that the application of the S-Minicon algorithm will generate rewritings that do not necessarily have the same schema (we have relaxed the condition on the head variables). So, to verify the containment, the algorithm selects only the comparable rewritings (\mathcal{CRW}) and verify if $q_i \not\subseteq \bigcup_{j=1}^n \mathcal{CRW}_{rw_j}$. A comparable rewriting [14] is a rewriting that has the same

schema as the query. The subsumption test verifies that no tuple has been filtered by the double rewriting. In other words, any tuple in q_i can be accessed by both \mathcal{AV} and \mathcal{MV} . The algorithm then terminates and returns q_i as a new authorization view on \mathcal{MV} . Otherwise, $q_i \not\subseteq \bigcup_{j=1}^n \mathcal{CRW}_{rw_j}$, which means that one of the two rewriting steps has filtered some tuples. In this case the double rewriting algorithm is applied again by considering \mathcal{RW}_{rw_j} (for j from 1 to n) as the set of queries to be rewritten.

Algorithm 2. *ACMVAlgorithm*

```

Input:  $\mathcal{Q}$ : Set of views which give a full access on  $\mathcal{MV}$ 
 $\mathcal{AV}$ : Set of authorization views on basic relations
 $\mathcal{MV}$ : Set of materialized views
Output:  $\mathcal{AVMV}$ : Set of authorization views on  $\mathcal{MV}$ 

while  $\mathcal{Q}$  is not empty do
    pick  $q_i$  in  $\mathcal{Q}$ 
     $\mathcal{RW} = \text{DoubleRewriting}(q_i, \mathcal{AV}, \mathcal{MV})$ 
    if  $\bigcup \mathcal{CRW}$  subsumes  $q_i$  then
        | Add  $q_i$  to  $\mathcal{AVMV}$ 
    else
        | Add  $\mathcal{RW}$  to  $\mathcal{Q}$ 
return  $\mathcal{AVMV}$ ;

```

4 Related Work

Rosenthal and Sciore [11] have considered the problem of how to automatically coordinate the access rights of the warehouse with those of sources. The framework proposed by the authors determine only if a user has right to access a derived table (based on explicit permission) but our proposal goes further by determining which part the user has right to access in the derived table. Also, the authors stated the inference rules at a high level. The properties of the underlying inference system and the efficiency of the proposed algorithm were not investigated and remain an open research issue.

In [4], the authors have built on [5] to provide a way to select access control rules to be attached to materialized view definitions based on access control rules over base relations. They resort to the basic form of the bucket algorithm which does not allow to derive all relevant access control rules. Another limitation of this work is that since they only deal with selection of rules, the framework remains strongly dependent of the base relations. That is, the body of the derived rules involves base relations only. In our work, we synthesize new rules from existing rules where the body of the new rules makes reference to materialized views.

5 Conclusion

In the case of large organizations, the management of thousands of datasets is very common. Ensuring data confidentiality in the presence of materialized

views is also important. In this work, we presented a novel approach for an automated method to derive authorization views to be attached to materialized views. We also presented S-MiniCon algorithm, an adaptation of a query rewriting algorithm to the security context. As mentioned above, we have discussed only conjunctive queries. In large systems, (e.g., data warehouses), materialized views can be used to precompute and store aggregated data (e.g., sum of sales). This framework should be extended to accommodate materialized views with aggregations. For this purpose, we will consider algorithms for rewriting aggregate queries using views [12].

References

1. Abiteboul, S., Hull, R., Vianu, V.: *Foundations of Databases*. Addison-Wesley (1995)
2. Chakravarthy, U.S., Grant, J., Minker, J.: Logic-based approach to semantic query optimization. *ACM Trans. Database Syst.* 15(2), 162–207 (1990)
3. Chaves, L.W.F., Buchmann, E., Hueske, F., Böhm, K.: Towards materialized view selection for distributed databases. In: *EDBT*, pp. 1088–1099 (2009)
4. Cuzzocrea, A., Hacid, M.-S., Grillo, N.: Effectively and efficiently selecting access control rules on materialized views over relational databases. In: *IDEAS*, pp. 225–235 (2010)
5. Nait-Bahloul, S.: *Inference of security policies on materialized views*. rapport de master 2 recherche (2009), <http://liris.cnrs.fr/~snaitbah/wiki>
6. Nait-Bahloul, S., Coquery, E., Hacid, M.-S.: Access control to materialized views: an inference-based approach. In: *EDBT/ICDT Ph.D. Workshop*, pp. 19–24 (2011)
7. Olson, L.E., Gunter, C.A., Madhusudan, P.: A formal framework for reflective database access control policies. In: *ACM Conference on Computer and Communications Security*, pp. 289–298 (2008)
8. Pottinger, R., Levy, A.Y.: A scalable algorithm for answering queries using views. In: *VLDB*, pp. 484–495 (2000)
9. Priebe, T., Pernul, G.: A Pragmatic Approach to Conceptual Modeling of OLAP Security. In: Kunii, H.S., Jajodia, S., Sølvberg, A. (eds.) *ER 2001*. LNCS, vol. 2224, pp. 311–324. Springer, Heidelberg (2001)
10. Rizvi, S., Mendelzon, A.O., Sudarshan, S., Roy, P.: Extending query rewriting techniques for fine-grained access control. In: *SIGMOD Conference*, pp. 551–562 (2004)
11. Rosenthal, A., Sciore, E.: Administering permissions for distributed data: Factoring and automated inference. In: *Proc. of IFIP WG11.3 Conf.* (2001)
12. Srivastava, D., Dar, S., Jagadish, H.V., Levy, A.Y.: Answering queries with aggregation using views. In: *VLDB*, pp. 318–329 (1996)
13. Steger, J., Günzel, H., Bauer, A.: Identifying security holes in olap applications. In: Thuraisingham, B.M., van de Riet, R.P., Dittrich, K.R., Tari, Z. (eds.) *DBSec. IFIP Conference Proceedings*, vol. 201, pp. 283–294. Kluwer (2000)
14. Wang, J., Maher, M., Topor, R.: Rewriting Unions of General Conjunctive Queries Using Views. In: Jensen, C.S., Jeffery, K., Pokorný, J., Šaltenis, S., Bertino, E., Böhm, K., Jarke, M. (eds.) *EDBT 2002*. LNCS, vol. 2287, pp. 52–69. Springer, Heidelberg (2002)
15. Yang, J., Karlapalem, K., Li, Q.: Algorithms for materialized view design in data warehousing environment. In: *VLDB*, pp. 136–145 (1997)

Enhancing the Security of On-line Transactions with CAPTCHA Keyboard

Yongdong Wu and Zhigang Zhao

Institute for Infocomm Research, Singapore
{wydong,zzhao}@i2r.a-star.edu.sg

Abstract. In an on-line transaction, a client usually have to present some authenticators (password, user certificate or both) to the server. However, those authenticators are exposed to client-side malware such that the malware is able to obtain the server-client messages, or impersonate the user to build another “secure” channel with the server.

The present paper aims to patch this client-side security flaw with a novel password-input method. Specifically, it enables a user to input a password by clicking an on-screen CAPTCHA keyboard, rather than a keyboard typing. The CAPTCHA keyboard is designed to greatly increase the difficulty of password eavesdropping and phishing in a malicious environment given that the malware can not monitor the browser secret memory space. Our implementation shows that Firwfox browser incorporated with CAPTCHA Keyboard and smartcard is viable and transparent over HTTPS protocol.

1 Introduction

Presently, most of the on-line applications such as e-banking and e-government are built on or assisted by HTTPS protocol. These applications carry on the server-client procedure as follows. When a user initiates a transaction with a web browser (*e.g.*, Internet Explorer or IE for short, Firefox), she will send a request to the web server with its URL (Universal Resource Locator). On a request, the server will ask the user to input her personal information. Once the server authenticates the browser with the user’s input, the web server sends a confidential page within the browser window which will be shown to the user. In order to carry on the on-line transaction securely, the international standard SSL/TLS [1] allows both server and client to authenticate each other and negotiate a cryptographic suite before transmitting and receiving the first byte of data. In the protocol, mutual authentication between client and server is the most critical component.

All the smartcard-based SSL/TLS schemes assume that the channel between the browser and smartcard is secure. However, this assumption does not always hold when the computer is compromised. *An attacking Trojan horse program that places itself between the signing software and the signature smart card can observe the communication between the two parties* [2]. In a compromised computer, the malware is able to intercept the password input and hence it can access the

smartcard at will such that the above smartcard-based authentication schemes fail.

This paper presents a CAPTCHA keyboard to deter password eavesdropping attack and a watermark mechanism to discourage password phishing. Specifically, in the process certificate-based TLS channel set-up, a browser will generate an on-screen CAPTCHA keyboard windows for inputting password. The CAPTCHA window must have the same visible watermark (or background) as that of the browser main window. When a user clicks the CAPTCHA keyboard, the click points are mapped to a password by the browser internally. Furthermore, all the communication messages between the smartcard and the browser are protected. The implementation on Firefox shows that the scheme is viable.

2 Secure On-line Transaction Protocol

2.1 Security Model

There are 5 parties in an on-line transaction workflow: user \mathcal{U} , remote server \mathcal{S} , Internet browser \mathcal{B} , user smartcard \mathcal{C} , and adversary \mathcal{A} . User \mathcal{U} communicates with server \mathcal{S} via the browser \mathcal{B} . In order to build a secure channel with SSL/TLS protocol, the server has a certificate issued by a CA (Certificate Authority). Browser \mathcal{B} holds the public key of the CA so as to verify the server's certificate. The user holds smartcard \mathcal{C} which stores a certificate issued by server \mathcal{S} or CA and the corresponding private key. In order to prove the ownership of the smartcard, user \mathcal{U} shares a password p with smartcard \mathcal{C} . Besides the user-smartcard password p , user may share a password P with server. These two passwords may or may not be the same. The adversary \mathcal{A} has the capability to control all the communication channels. More precisely,

- Adversary \mathcal{A} controls the network between browser \mathcal{B} and server \mathcal{S} . He is able to disrupt the communication or alter data at will. For example, an adversary \mathcal{A} can have this capability if he is in a privileged “gateway” of the network.
- Adversary \mathcal{A} is able to install malware in the user's machine so as to take partial control of the user's machine and perform some actions such as monitoring USB communication, Network communication, keystroke monitoring, screen dumping etc.
- Adversary \mathcal{A} is not able to monitor/modify the browser secret space where secrets are stored and manipulated due to software protection technologies such as code obfuscation. Hence, it is beyond our scope to defeat the attack where \mathcal{A} exploits the secrets (e.g., session key, random source), and CA's public key stored inside browser \mathcal{B} , etc.
- Adversary \mathcal{A} is unable to corrupt either server \mathcal{S} or smartcard \mathcal{C} .
- CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart, [3]) problem can not be solved by computer technology, e.g., OCR recognition technologies.

2.2 Security-Enhanced TLS Protocol

Given that a user shares a password p with a smartcard which stores a certificate C_u /private key pru , the present browser-smartcard protocol proves that the user owns the smartcard. Furthermore, assume the server has an authorized certificate C_s , it builds a secure TLS channel with browser/smartcard. Thus, the user and the server are able to set-up a secure channel even in the presence of malware. As shown in Fig. 1, the security-enhanced TLS protocol is as follows.

- (1) When a browser is activated, it selects a one-time secret watermark \mathbf{W} . Whenever a TLS channel is setting up, the watermark \mathbf{W} is used as the background of the page.
- (2) To ensure the authenticity of the smartcard ownership, Browser \mathcal{B} and smartcard carries on a verification protocol as follows:
 - Browser asks the user to input a password p with CAPTCHA keyboard (Subsection 2.3). Then it sends to Smartcard \mathcal{S} a request message **Login** so as to negotiate a session secret k with smartcard. For instance, we can employ Encrypted Key Exchange (EKE) [4] to share an authentic session key k between browser and smartcard such that any original plaintext m between them is encrypted with an authentic symmetric cipher $\mathcal{E}_k(\cdot)$.
 - Browser sends to smartcard a **CertificateVerify** request including the authenticated encryption $\mathcal{E}_k(h)$, where h is the hash value of all the transcripts in the previous steps between browser and server.
 - Smartcard \mathcal{C} recovers h by decrypting the received ciphertext with the session key k , and then creates the signature $\sigma = \text{SIGN}_{pru}(h)$ with user's private key pru stored in the smartcard. Afterwards, smartcard sends the signature σ to Browser.
 - Browser sends to Server \mathcal{S} a **CertificateVerify** message including σ . Then Server \mathcal{S} verifies the signature using the user's public key pk_u extracted from user's certificate.
- (3) Continue the rest of TLS protocol.

2.3 On-screen CAPTCHA Keyboard

In the on-line transaction, the smartcard will perform all the operations related to the private key, e.g., decrypting and signing. In order to ensure that only the authorized party can perform these security-related operations, a password should be used, otherwise, a malware can obtain the decrypted data, or genuine signatures for bogus data. Hence, a password is usually required for smartcard applications. However, when a user types the password with a normal keyboard, the password may be easily intercepted by a malware. Although an on-screen keyboard is able to deter the malware, it can be invalidated by an advanced OCR-enabled malware.

We design a new user interface as Figure 2 (left) to input password to browser by adapting the CAPTCHA keyboard [5]. Specifically, when a browser asks for inputting a password, it outputs a CAPTCHA keyboard image whose background is the watermark \mathbf{W} . Before clicking the CAPTCHA keyboard, the user

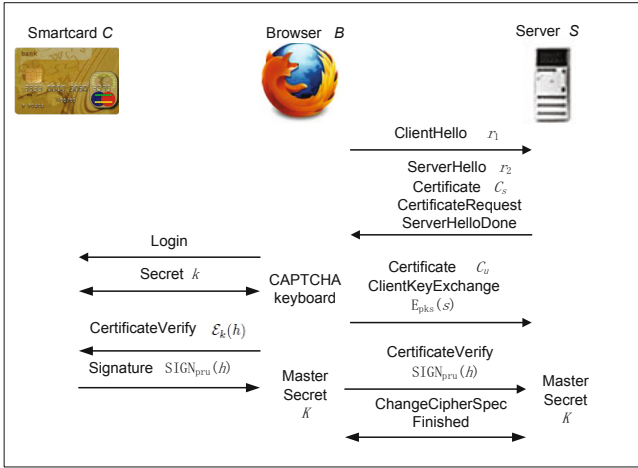


Fig. 1. Malware-resistant TLS handshaking protocol in on-line transactions

shall check whether the background (or watermark **W**) of the CAPTCHA window matches that of the browser main window. If not, the user will terminate the transactions. Otherwise, the user clicks the CAPTCHA keyboard, the character in the position is regarded as the input of the password. As the browser creates the image and knows the mapping between the character and position, it can obtain the password. Hence, the on-screen CAPTCHA keyboard has the merits of both text-based CAPTCHA and object-based CAPTCHA: good for password input and secure against keyboard/mouse eavesdropping.

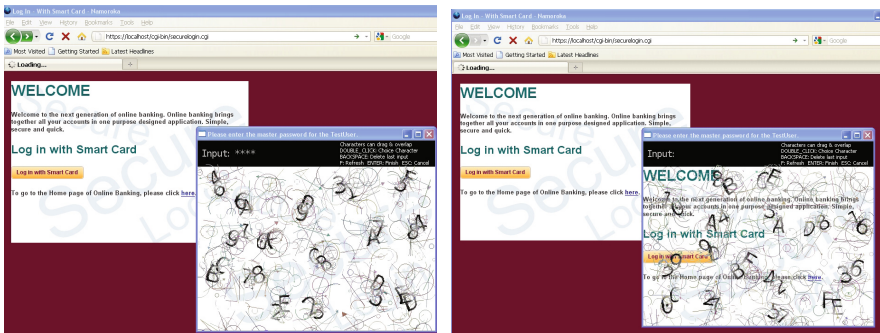


Fig. 2. CAPTCHA keyboard. Left: genuine, Right: Bogus.

3 Discussion

3.1 Security

Protocol Security: In the process of SSL/TLS handshake, the smartcard is used as a trusted computing platform for client-side authentication. Hence, the setup process of client-server communication channel is secure if the browser environment is trusted. Meanwhile, browser and smartcard run the standard password authentication protocol for login process, hence, the login process is secure too. Hence, an attacker may turn to exploit the browser environment so as to get the secrets of the user, i.e., the private key and/or the password. As the private key is stored in the smartcard, and never disclosed outside the smartcard, the adversary has no means to obtain the private key. But if the adversary is able to get the password, it has the capability to access the private oracle such as signing at will. Thus, an adversary will attempt to eavesdrop the password so as to penetrate the on-line transaction scheme.

Password Phishing: An adversary may forge a CAPTCHA keyboard and allude the user to disclose her password. However, this phishing attack can be detected by the user because this bogus window will have noticeable artifacts. According to our protocol, a malware does not have the original watermark **W**, it has to dump the screen and super-impose a CAPTCHA keyboard to generate a fake password input window. Therefore, the content in the browser window will exist in the phishing window as Figure 2 (right).

Password Eavesdropping: As described in Subsection 2.3, a password is input when a user clicks on the CAPTCHA image. In this process, a malware is able to obtain the clicking points by hooking the mouse input routine. However, the malware can not obtain the password because it can not solve CAPTCHA from the points and the CAPTCHA image. The argument is as follows. If the adversary can solve CAPTCHA with the intercepted coordinates, the adversary is capable of recognizing the CAPTCHA at a very high probability when the distorted characters are close to each other. However, this capability is in conflict with CAPTCHA assumption. In addition, the present 2-dimensional CAPTCHA does not require that the characters align in a (rough) line. Indeed, as each character can be randomly distributed over a 2-D image background, and overlapped with each other, the present CAPTCHA mechanism is at least as secure as the conventional text-based CAPTCHA. Hence, the password input with CAPTCHA keyboard is secure even if the malware is able to get the CAPTCHA keyboard and click points.

Human-Assisted Eavesdropping: The present scheme is not a panacea because an adversary may help a malware to get the password. Specifically, a malware may eavesdrop the CAPTCHA screen and click points, then send them to a remote human. Based on the assumption of CAPTCHA, the human is able

to solve the CAPTCHA problem. However, this “successful” attack incurs extra human effort, and high risk due to the trace (e.g., IP address) leakage of the human such that the attacker may not launch this attack.

3.2 Performance Comparison

With the similar configuration as other browsers, the present scheme provides overall better performance. Table 1 shows the comparison result with the competitors. The second column indicates the requirements for user certificate. The present scheme is applicable to absence/presence of client certificate. In the third column, scheme [7] has additional requirement for the web pages due to the proxy mechanism. The fourth and fifth columns show that only the present scheme is secure in the malicious host, i.e., only the present scheme can provide a secure on-line transaction in a compromised and legacy environment. All of the schemes have the same processing time at both the smartcard and server.

Table 1. Performance comparison

	User's Certificate	Standard compatibility	Keystroke Anti-eavesdropping	Password Anti-Phishing	Smartcard security
Microsoft IE	Optional	Yes	No	No	Low
TLS+SESAME [6]	Yes	Yes	No	No	Low
Urien [7]	Yes	No	No	No	Low
OTP	No	Yes	No	No	Nil
Present	Optional	Yes	Yes	Yes	High

References

1. Dierks, T., Rescorla, E.: The TLS Protocol, Version 1.1, IETF Draft, RFC 2246 (2005)
2. Spalko, A., Cremers, A.B., Langweg, H.: The fairy tale of What You See Is What You Sign - Trojan Horse Attacks on Software for Digital Signatures. In: IFIP Working Conf. on Security and Control of IT in Society-II (2001)
3. Mori, G., Malik, J.: Recognizing objects in adversarial clutter: Breaking a visual CAPTCHA. In: CVPR, vol. 1, pp. 134–141 (2003)
4. Sheffer, Y., Zorn, G., Tschofenig, H., Fluhrer, S.: An EAP Authentication Method Based on the Encrypted Key Exchange (EKE) Protocol, IETF RFC 6124 (2011)
5. Szydłowski, M., Kruegel, C., Kirda, E.: Secure input for Web applications. In: AC-SAC, pp. 375–384 (2007)
6. Vandenwauver, M., Ashley, P., Claessens, J., Looi, M., Moreau, W.: Using Smart Cards to Integrate SSL/TLS and SESAME. In: IFIP TC6/TC11 International Conference on Communications and Multimedia Security, vol. 152, pp. 303–317 (1999)
7. Urien, P.: Collaboration of SSL smart cards within the WEB2 landscape. In: Int'l Symposium on Collaborative Technologies and Systems, pp. 187–194 (2009)

Fighting Pollution Attack in Peer-to-Peer Streaming Networks: A Trust Management Approach

Xin Kang and Yongdong Wu

Institute for Infocomm Research, 1 Fusionopolis Way, #21-01 Connexis, Singapore 138632
{xkang, wydong}@i2r.a-star.edu.sg

Abstract. Nowadays, peer-to-peer (P2P) streaming systems have become the most popular way to deliver multimedia content over the internet due to their low bandwidth requirement and high video streaming quality. However, P2P streaming systems are vulnerable to various attacks, especially pollution attacks, due to their distributed and dynamically changing infrastructure. In this paper, by exploring the unique features of various pollution attacks, we propose a trust management system tailored for P2P streaming systems. Both direct trust and indirect trust are taken into consideration in the design of the system. A new approach to model the direct trust is proposed, and a dynamic confidence factor that can dynamically adjust the weight of direct and indirect trust is also proposed. It is shown that the proposed trust management system is effective in identifying polluters and preventing them from further sharing of polluted data chunks.

1 Introduction

The past decade has witnessed the rising of large-scale P2P multimedia streaming networks, over which millions of users interact with each other and exchange media contents in a distributed way. In these P2P streaming networks, peers are assumed to be well behaved and non-malicious. However, due to their distributed and dynamically changing infrastructure, P2P streaming systems are vulnerable to various attacks, especially pollution attacks. Malicious peers may intentionally forge data chunks or alter received data chunks, and make these polluted data chunks available to other peers. Without the ability to differentiate between malicious peers and good peers, peers are highly likely to request and forward polluted data chunks, consequently degrading the performance of the whole system. Therefore, effective pollution-resistant mechanisms are badly needed for P2P streaming systems.

As a matter of fact, a great deal of scholarly work has already published on the design of pollution-resistant mechanisms for P2P streaming systems. In [1], by measuring the PPLive streaming system, the authors showed that without any pollution-resistant mechanisms, the polluted content could spread through much of the P2P network. In [2], blacklisting and reputation mechanisms were proposed to avoid polluted content dissemination and isolate malicious peers. While in [3], the authors proposed a trust management system to defend strategic polluters who could upload polluted and clean chunks alternatively to avoid being detected. Trust management mechanisms for P2P applications have also been extensively studied in literatures [4-7]. However, trust is in nature a complex psychological concept involving a lot of complex properties,

the methodology used to model the trust has a significant influence on the performance of the trust management system. Trust models should be designed to meet the specific requirements of different P2P applications.

In this paper, by exploiting the unique features of pollution attacks, we design a trust management system to defend against various kinds of pollution attacks for P2P streaming systems. The main contributions of this paper are as follows.

- A theoretic framework on the modeling of the trust management system for P2P streaming systems to fight against pollution attack is proposed and investigated.
- A dynamic confidence factor is proposed to dynamically adjust the weight of direct and indirect trust in computing the trust, which is shown to be pretty effective in reducing the negative effects of the bad-mouthing attack and the collusion attack. Guidelines on how to design such a dynamic confidence factor are given, and two specific designs of the dynamic confidence factor are proposed and investigated.
- A novel approach to model the direct trust is proposed based on the unique features of pollution attacks. It is shown that the proposed trust model is effective in defending against the on-off pollution attack.

2 Trust Management in P2P Streaming Networks

In our trust management system, we use $T_{i,j}(t)$ to denote the trust that user i has on user j at time t . The value of $T_{i,j}$ is within the range $[0, 1]$, with "0" denoting distrust and "1" denoting fully trust.

Let $D_{i,j}(t)$ and $I_{i,j}(t)$ denote the direct trust and indirect trust that user i has on user j at time t , $T_{i,j}(t)$ can then be computed as follows

$$T_{i,j}(t) = \alpha_{i,j}D_{i,j}(t) + (1 - \alpha_{i,j})I_{i,j}(t), \quad (1)$$

where $0 \leq \alpha_{i,j} \leq 1$ is a parameter reflecting user i 's confidence of its direct trust over user j . A larger value of $\alpha_{i,j}$ indicates that user i is more confident of its own judgement of user j , while a smaller value of $\alpha_{i,j}$ indicates that user i relies more on other peers' recommendation on user j . For notation convenience, we drop t in the following discussion.

2.1 Confidence Factor

Different from the existing literatures (such as [3]) that use a constant to adjust the weight between the direct trust and the indirect trust, in this paper, we define a dynamic *confidence factor* $\alpha_{i,j}$, which is given as

$$\alpha_{i,j} = f(N_{i,j}^T), \quad (2)$$

where $f(\cdot)$ is a function, and $N_{i,j}^T$ denotes the number of direct transactions that has been made between user i and j at time t .

Basically, $f(\cdot)$ should have the following properties:

- $\forall N_{i,j}^T \in [0, +\infty)$, $f(N_{i,j}^T) \in [0, 1]$.
- $f(0) = 0$, and $\lim_{N_{i,j}^T \rightarrow \infty} f(N_{i,j}^T) = 1$.
- $f(N_{i,j}^T)$ is a monotonic increasing function of $N_{i,j}^T$.

Remark: (a). The first property guarantees that the value of the trust defined in Equation (11) falls within the range $[0, 1]$. (b). The second property captures the fact that when there is no direct transaction between user i and user j , user i can only rely on the indirect trust values gathered from other peers to determine its trust of user j . When the number of direct transactions between user i and user j is sufficiently large, user i can ignore the indirect trust. (c). The third property captures the fact that the confidence of user i on its own judgement of the trustworthiness of user j increases when the number of direct transactions between them increases. (d). It is observed that these properties of $f(\cdot)$ are similar to those of cumulative distribution functions (CDF) of random variables. Therefore, the design of $f(\cdot)$ can borrow ideas from the probability theory.

In this paper, we propose two schemes that satisfy all the properties mentioned above to design the confidence factor $\alpha_{i,j}$. The two designs are given as follows.

$$\text{Confidence Factor Design A (CFDA): } \alpha_{i,j} = \frac{N_{i,j}^T}{N_{i,j}^T + c}, \quad (3)$$

where c is a positive constant.

$$\text{Confidence Factor Design B (CFDB): } \alpha_{i,j} = 1 - \beta^{N_{i,j}^T}, \quad (4)$$

where $0 < \beta < 1$ is a constant. It is worth pointing out that the value of c and β have a significant impact on the increasing rate of $\alpha_{i,j}$. In practice, they can be designed as a tunable parameter that can be tuned by users depending on the network environment.

2.2 Direct Trust

Direct trust is the trust of a peer on another peer based on their direct interacting experience. It is established only based on previous direct transactions between peers. Let $N_{i,j}^c(t)$ and $N_{i,j}^p(t)$ denote the total number of clean chunks and polluted chunks that user i has received from user j at time t , the direct trust $D_{i,j}(t)$ that user i has on user j at time t can be defined as

$$D_{i,j}(t) = e^{-\rho N_{i,j}^p(t)} \frac{N_{i,j}^c(t)}{N_{i,j}^c(t) + \eta}, \quad (5)$$

where ρ and η are positive constants, and $e^{(\cdot)}$ is the exponential function. It is easy to verify that the value of $D_{i,j}$ is within the range $[0, 1]$, and $D_{i,j}$ is an increasing function with regard to $N_{i,j}^c$ and a decreasing function with regard to $N_{i,j}^p$. It is worth pointing out that the proposed direct trust model is shown to be resistant to on-off attacks when ρ and η satisfy certain conditions.

Proposition 1: The trust management scheme $D_{i,j}(t) = e^{-\rho N_{i,j}^p(t)} \frac{N_{i,j}^c(t)}{N_{i,j}^c(t) + \eta}$ is resistant to on-off attack when $\rho > \ln(1 + \frac{1}{\eta})$.

Proof. The proof is omitted here due to the space limitation.

2.3 Indirect Trust

Indirect trust is the trust of a peer on another peer obtained via third-party peers' recommendations. Indirect trust is important when two peers have little or no direct interactions. Indirect trust is established through trust propagation, i.e., trustworthy peers are more likely to give honest feedbacks than distrusted peers. Indirect trust is determined by two key factors: the credibility of the third-party peer and its recommendation value of the trustee. In this paper, we define the indirect trust as

$$I_{i,j}(t) \triangleq \frac{\sum_{k \in S_{i,j}(t)} C_{i,k}(t) R_{k,j}(t)}{\sum_{k \in S_{i,j}(t)} C_{i,k}(t)}, \quad (6)$$

where $S_{i,j}(t)$ denotes the set of peers that has direct transactions with both peer i and peer j . $C_{i,k}(t)$ is the credibility of peer k , and $R_{k,j}(t)$ is user k 's recommendation value of user j based on their interaction experience. In this paper, we let $C_{i,k}(t) = D_{i,k}(t)$ and $R_{k,j}(t) = D_{k,j}(t)$, where $D_{i,k}(t)$ is the peer i 's direct trust on peer k , and $D_{k,j}(t)$ is the peer k 's direct trust on peer j .

2.4 Trust Updates

Intuitively, recent interactions should have more weight than old interactions in computing the trust. Here, we assume that the interactions made within the recent Δt time have the same weight, and the weight of the interactions made older than Δt will experience certain attenuation. Mathematically, the update functions can be written as

$$N_{i,j}^c(t') = e^{-\lambda \Delta t} N_{i,j}^c(t) + (N_{i,j}^c(t') - N_{i,j}^c(t)), \quad (7)$$

$$N_{i,j}^p(t') = e^{-\mu \Delta t} N_{i,j}^p(t) + (N_{i,j}^p(t') - N_{i,j}^p(t)), \quad (8)$$

where λ and μ are positive constants, and $t' = t + \Delta t$. In this paper, we refer to λ and μ as *forgetting factor* and *forgiving factor*, respectively and request $\lambda > \mu$. This makes our trust management system remembers the unpleasant interactions longer than the pleasant interactions, which further increases our system's resistant to on-off attacks.

2.5 Utilization of Trust Values

With the trust management system introduced in this section, peers can easily compute the trust values of other peers. The trust values can then be used by peers to identify polluters, and to determine whether to perform a transaction with another peer. Suppose peer i decides to make a transaction with peer j with probability $p_{i,j}(t)$ at time t , then $p_{i,j}(t)$ can be determined by

$$p_{i,j}(t) = \begin{cases} 0, & \text{if } T_{i,j}(t) < \theta_i^P, \\ \chi_{i,j}, & \text{if } \theta_i^P \leq T_{i,j}(t) < \theta_i^G, \\ 1, & \text{if } T_{i,j}(t) \geq \theta_i^G, \end{cases} \quad (9)$$

where $\chi_{i,j}$ is a constant, θ_i^P and θ_i^G are the thresholds for peer i to identify malicious and good peers, respectively. It is worth pointing out that peer i can set different $\chi_{i,j}$ for different peer j , depending on the content of the potential transaction. For example, peer i is willing to set a high value of $\chi_{i,j}$ for a peer j that has data chunks which are closer to its playback time.

3 Performance Evaluation under Potential Attacks

In this section, we give an introduction of the commonly seen attacks [1-3] in P2P streaming networks, such as *bad-mouthing attack* and *on-off attack*. The performance of the proposed trust management system are then investigated under these attacks.

3.1 Bad-Mouthing Attack

Bad-mouthing attack refers to the scenario that a single malicious peer or a group of malicious peers deliberately provides negative recommendations to frame up good peers. In our trust management system, the following two ways are adopted to fight against bad-mouthing attacks: (a). *Filtering out potential malicious recommendations*. When computing the indirect trust $I_{i,j}(t)$, peer i only select the top K peers based on the value of $D_{i,j}(t)$ from the set $S_{i,j}(t)$. Through this way, malicious recommendations from untrustworthy peers can be effectively avoided. (b). *Reducing the weight of indirect trust*. Bad-mouthing attacks are unavoidable as long as recommendations are taken into consideration. Therefore, reducing the weight of indirect trust in computing the trust is a good way to defend against bad-mouthing attacks.

The performance of our trust management system under the bad-mouthing attack is shown in Fig. 1. In this experiment, we let peer j keep uploading clean chunks to peer i . We assume that some malicious peers give bad recommendations on peer j for 80 percent of time. Peer i computes the trust values of peer j for 50 interactions based on the constant confidence factor scheme ($\alpha_{i,j} = 0.5$) and our dynamic confidence factor scheme, respectively. It is observed from Fig. 1 that the proposed dynamic confidence factors can effectively reduce the weight of indirect trust, and thus greatly increase our trust management system’s resistant to the bad-mouthing attack.

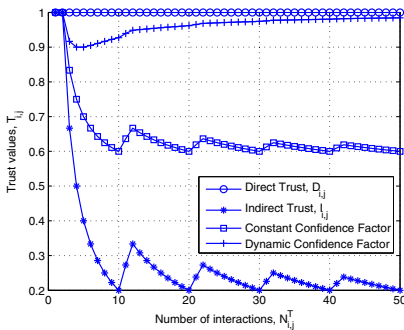


Fig. 1. Performance under bad-mouthing attacks

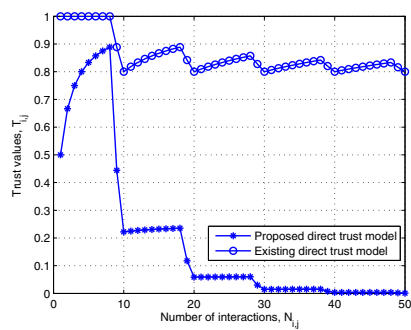


Fig. 2. Performance under on-off attacks

3.2 On-Off Attack

On-off attack refers to the scenario that a malicious peer sends clean and polluted chunks alternatively to other peers with an aim to keep its trust value above certain

threshold, and thus avoid being identified as a polluter. The on-off attack exploits the fact that most of the trust management mechanisms are designed to tolerate certain levels of unintentionally polluted chunks (such as incomplete and erroneous data chunks) due to bad network conditions. To combat the on-off attack, an effective way is to design a trust management system in which the dropping rate of trust value is larger than its increasing rate, i.e., the trust value drops sharply when the peer uploads polluted chunks, and accumulates slowly when the peer uploads the same number of clean chunks.

The performance of our trust management system under the bad-mouthing attack is shown in Fig. 2. In this experiment, peer j performs on-off attacks (20% on-off rate) to peer i . The trust values of peer j are computed for 50 interactions based on the existing direct trust model (EDTM) given in [7] and proposed direct trust model (PDTM), respectively. It is observed from Fig. 2 that the dropping rates are much larger than the increasing rate under PDTM. Therefore, the trust values obtained under PDTM are gradually decreasing in the long run. On the other hand, it is observed that the trust values computed under EDTM are maintained above certain thresholds, which indicates that EDTM is not resistant to the on-off attack.

4 Conclusion

In this paper, a trust management system to fight against various kinds of pollution attacks for P2P streaming systems are proposed by exploring the unique features of pollution attacks. A dynamic confidence factor is proposed to dynamically adjust the weight of direct and indirect trust in computing the trust, which is shown to be pretty effective in fighting against the bad-mouthing attack. Guidelines on how to design such a dynamic confidence factor are given, and two specific designs of the dynamic confidence factor are proposed. Besides, a novel direct trust model that is proved to be resistant to the on-off pollution attack is proposed and investigated.

References

1. Dhungel, P., Hei, X., Ross, K.W., Saxena, N.: The pollution attack in p2p live video streaming: measurement results and defenses. In: ACM SigComm Workshop on P2P Streaming and IPTV, Kyoto, Japan (August 2007)
2. Borges, A., Almeida, J., Campos, S.: Fighting pollution in p2p live streaming systems. In: IEEE Int. Conf. on Multimedia and Expo, Hannover, Germany (June 2008)
3. Hu, B., Zhao, H.V.: Pollution-resistant peer-to-peer live streaming using trust management. In: IEEE Int. Conf. on Image Processing (ICIP), Cairo, Egypt (November 2009)
4. Kamvar, S., Schlosser, M., Garcia-Molina, H.: The eigentrust algorithm for reputation management in p2p networks. In: ACM WWW Conf., Budapest, Hungary (May 2003)
5. Xiong, L., Liu, L.: Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities. IEEE Trans. Knowledge and Data Eng. 16(7), 843–857 (2004)
6. Zhou, R., Wang, K.H.: Powertrust: A robust and scalable reputation system for trusted peer-to-peer computing. IEEE Trans. Parallel and Distributed Systems 18(4), 460–473 (2007)
7. Sun, Y.L., Yu, W., Han, Z., Liu, K.J.R.: Information theoretic framework of trust modelling and evaluation for ad hoc networks. IEEE J. Select. Areas Commun. 24(2), 305–317 (2006)

A Framework for Anonymizing GSM Calls over a Smartphone VoIP Network

Ioannis Psaroudakis, Vasilios Katos, and Pavlos S. Efraimidis

Information Security and Incident Response Unit
Department of Electrical and Computer Engineering
Democritus University of Thrace
University Campus, Xanthi 67100, Greece
jpsaroud@duth.gr, {vkatos,pefraimi}@ee.duth.gr
<http://isir.ee.duth.gr/>

Abstract. The proposed framework describes a service for users that gives them the ability to make gsm calls from their smartphones without revealing their identity. The principle to achieve that is simple: instead of using your cell phone to make a call, pick somebody else's phone to do so. We proposed an infrastructure of smartphones, sip registrars and sip proxies to provide caller anonymity. We developed a testbed where a smartphone registers on a SIP registrar and can start GSM conversation through another smartphone acting as a GSM gateway, by using a SIP proxy. Empirical evaluation revealed no significant QoS degradation.

Keywords: privacy, sip, smartphone, gsm anonymity.

1 Introduction and Motivation

Resource sharing in community networks is a well established concept since the dawn of the Internet. This has recently evolved through peer to peer networking, to grid computing and finally to cloud computing services, as the benefits of the collective paradigm are non disputable.

In the meantime, the wide adoption and commercial success of mobile networks due to the advances of wireless communications has resulted to the smart phone being the most preferred device for communicating through a variety of platforms. However a mobile phone device is not built with privacy in mind; the users location, preferences and behavior in general is recorded and shared between parties that offer the infrastructure, software, and operating system.

GSM calls are well regulated. Users are bound to a mobile number according to a regulatory framework and carriers are obliged to keep logs of their telephony conversations for a period of up to two years (Data Retention Directive 2006/24/EC).

In this paper we argue that certain privacy goals could be achieved by the active participation and collaboration of a community of users. We focus on VoIP and mobile communications and present a proof of concept for performing telephone calls on a mobile network with caller anonymity. In the context of this

paper, caller anonymity relates to the caller id and the call metadata; the underlying audio stream will be susceptible to passive eavesdropping by the callee's provider.

2 Related Work

Requirements and specifications for offering caller anonymity over SIP are defined in RFC3323 [1] for three use cases relating to withholding the identity from the intermediary parties, the final destination(s), or both.

Quality of Service (QoS) is a critical factor that needs to be considered when designing and deploying any kind of telephony communication. A prevalent QoS feature for telephone communications is the delay [5], both while establishing a session and, most importantly, during the actual session for the voice stream. These parameters affect significantly the choice of the appropriate privacy enhancing technology. In [2] the authors present the main categories of PETs and conclude that a large number of technologies cannot be integrated with a VoIP solution like SIP due to their negative impact on the QoS attributes. For instance, Onion Routing [3] and Mixes [4] exhibit high call delays, whereas Hordes [9], DC-Nets and pMixes [7] may be more suitable but the latter has scalability issues as the underlying computational cost is in $O(n^2)$. In addition, many technologies were not designed or implemented with a view to be applied in VoIP communications (Onion Routing for example) and as such they do not inherently support UDP which makes them suitable only for the call initiation phases. The principle behind our proposed scheme is similar to that of crowds [8]. However the main difference is that in crowds anonymity is achieved by routing communication randomly within a group of similar users whereas in our proposition we do not allow direct communication between users and traffic is routed through special sip protocol capable entities.

3 The Proposed Scheme

The main idea behind the proposed scheme rests on the assumption that a participant (or smart phone owner) is voluntarily willing to offer her equipment for other users to make calls. This setting leads to two advantages. First, the carrier would not be able to establish the identity of the real caller. Second, there could be no charge at all if the offering person has an unlimited time contract with the carrier. Clearly the success of the proposed scheme relies on ease of use, reliability and level of participation in accordance to Metcalfe's Law [6].

The requirements and issues for a practical solution involve the discovery of users willing to offer their phones as SIP-GSM relays, the discovery of call destinations every user can offer and the protection of participants from malicious peers.

Throughout the scheme the following roles and entities are identified:

- *Caller*: This role refers to the main beneficiary of the infrastructure which is the user that wishes to make a call to a user (callee) with a selective preservation of her anonymity. Alice will be caller in our examples.
- *Callee*: The user that accepts a call. Bob will have this role.
- *SIP-to-GSM gateway*: The user that acts as a VoIP to GSM gateway and shares her GSM service. In our scheme, Carol will have this role.
- *SIP Registrar*: The registrars maintain the user SIP accounts and act as back-to-back user agents [11]. We assume the trusted entities Registrar A (for Alice) and Registrar C (for Carol).
- *SIP Proxy*: Proxies act as intermediaries on the communication path providing call routing. We assume a single SIP Proxy entity in one of the scenarios.
- *GSM Carrier*: This role offers the GSM mobile phone service. We assume the entities GSM carrier A, C and B, for Alice, Carol and Bob respectively. GSM Carrier A will be assumed to be malicious.

3.1 The Anonymous Communication Scenario

Alice wants to communicate with Bob using her smartphone whilst maintaining her anonymity from Bobs carrier. More importantly, the GSM carrier of Alice should not learn anything about this phone call. Alice knows that a community of VoIP and GSM users is willing to help by sharing their phone and credits from their contracts. The easiest way is to communicate with some appropriate member (Carol) of the community by using the Internet infrastructure. Carol's device must be able to communicate with Alice using the Internet through her WIFI or HSPA connection and at the same time to call Bob using the GSM connection. This particular operation fulfilled by Carol is a so-called a SIP-to-GSM gateway operation and it is the kernel function for the service. The operation has to take place without any actions taken from Carol apart from her declaration of consent to lend her resources.

The user registration process should cover both roles the participants will have, that is Caller and SIP2GSM gateway. The users register to the SIP registrars and provide data such as user credentials, sharing resources, policy data and SLAs. SIP registrars are assumed to be trusted and to act as agents on behalf of the users. Every SIP Registrar in turn needs to be affiliated with at least one SIP proxy server. The affiliation is initiated with the SIP registrar administrator who will provide the data similar to the user registration process to the proxy server during the application process.

We define the following privacy requirements:

- P1 *Caller anonymity in the GSM network*. Alice's identity should be hidden from GSM provider A.
- P2 *Mutual anonymity between the caller and the gateway*. Alice should not know that her call is routed through Carol and vice versa.
- P3 *SIP-to-GSM gateway privacy*. The gateway's personal information, including its contracts and capabilities should only be available to SIP Registrar C.
- P4 *SIP Registrar privacy*. There will be no leakage of the information maintained by the SIP registrars A and C.

Although in principle the caller’s anonymity in the GSM environment can be trivially offered due to the apparent “incompatibility” of the two networks, the caller’s identity could be discovered from the actual voice stream which the GSM has access to. In general terms, this is considered as a probabilistic side channel, since the probability of identifying the caller from the available audio data is not necessarily equal to one. Furthermore, mutual anonymity is also required between the caller and the gateway. Anonymity of the caller is required because in the opposite case if Carol (the gateway) is malicious or a passive eavesdropper or (even worse) belongs to the GSM carrier, then she will have access to both the caller and the callee information. Therefore, P1 depends on P2.

P3 is perhaps the most important requirement. All information provided by the gateway needs to be protected as in the opposite case a curious participant may collect valuable data and generate statistics over the users and their contracts, which then can be used for personal gain. P3 also depends upon P4 which is offered by design.

3.2 Private VoIP to GSM Gateway Discovery

A fully developed version of our system will have to address additional privacy issues that arise in auxiliary functions of the system. An example is the gateway discovery procedure discussed earlier. If the SIP Registrar A is trusted (as we assumed earlier) then the privacy of Alice is preserved while requesting to use the platform for a call to Bob. Similarly, if the Registrar C of Carol is trusted then Carol can safely advertise her readiness to act as SIP-to-GSM gateway for specific GSM carriers. The above scheme can be further improved by adding a SIP Proxy to it. However, in all these cases, privacy relies on assumptions about the participating entities and/or the introduction of a proxy. A challenging requirement would be to solve the same problem for Honest-but-Curious or even malicious entities, by applying advanced cryptographic techniques [12]. We are currently examining a similar service for the needs of our application.

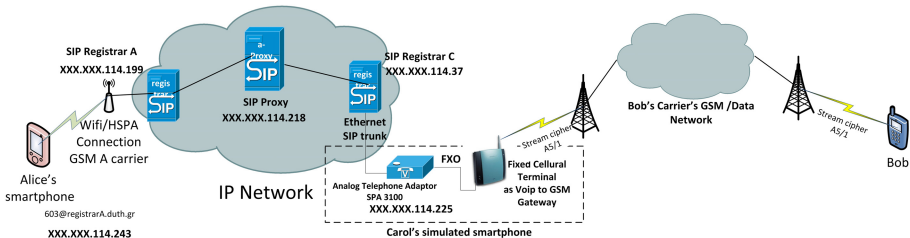


Fig. 1. The testbed environment

3.3 Performance Evaluation

Telephony applications require real time audio streaming and are tightly coupled with QoS network parameters. We therefore need to investigate whether the

proposed solution with the added security controls will not have negative impact on the user acceptance.

We proceed on implementing the proposed framework and making our first anonymous calls. We used two CentOS servers with Asterisk software as SIP registrar entities, a PC with TekSIP software as the Proxy server and a VoIP ATA (Linksys SPA3000) with Ericsson’s FCT as VoIP-to-GSM gateway (Fig. 1) simulating the application to be developed for smartphones. The initial caller (Alice) was an android smartphone running CSipSimple. For the data collection and analysis we used a 2960G Gigabit Cisco switch with port monitoring enabled to gather the call flow that shows the SIP negotiation (Fig. 2).

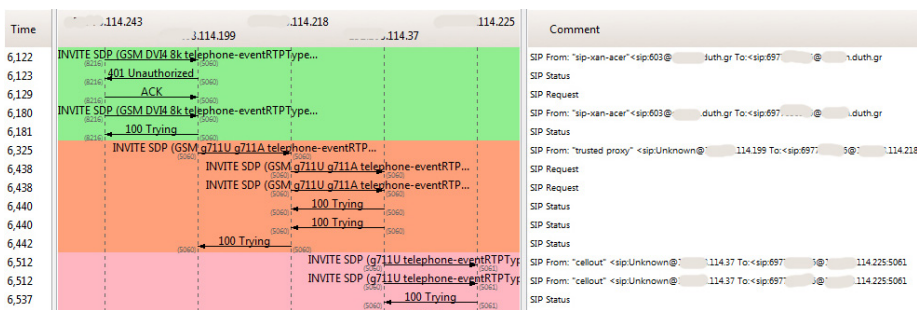


Fig. 2. Excerpt from the anonymised SIP flow using wireshark

From the call flow and the SIP methods it can be seen that Alice is unaware of the initial caller. None of the entities has full knowledge of the path of communication. As long as the proxy does not update the Via header in SIP INVITE as RFC 3261 [11] suggests, neither of SIP registrar has knowledge of each other:

```

INVITE sip:6977555555@x.x.114.37 SIP/2.0
Via: SIP/2.0/UDP x.x.114.218:5060;branch=z9hG4bK-2f6032f6;rport
Via: SIP/2.0/UDP x.x.114.199:5060;branch=z9hG4bK23f26f06;
    
```

Finally we performed a stress test with varying number of connections. It was established that the call on the SIP side, with one proxy (that is four SIP nodes in total) the majority of the calls can be established within a period of 500ms. This, compared to the GSM delay which is in the order of 8-12 seconds, is negligible.

4 Concluding Remarks and Areas for Future Research

We have described a framework for providing caller anonymity from their GSM by utilising a VoIP infrastructure and used SIP as a means to identify the issues and explore possible design and implementation alternatives. Following our empirical investigation, we concluded that adding such an infrastructure on a

GSM network will add negligible delays in the call establishment, as the bottleneck remains on the GSM side. Another area of research is in the development of a protocol so that each affiliated registrar advertises its network routing capabilities to the affiliated proxy in a dynamic way. As this proposed solution is defined over a novel configuration of a heterogeneous network, further security analysis of relevant threat vectors and corresponding countermeasures must be conducted. For example, a VoIP bot running on the proposed infrastructure could make excessive resource allocation and as such an anti spam over Internet Telephony mechanism must be deployed [10]. Lastly, legal issues must be looked into when offering such a service in public.

Acknowledgments. This work was performed in the framework of and partially funded by the GSRT/CO-OPERATION/SPHINX Project (09SYN-72-419) (<http://sphinx.vtrip.net>).

References

- Peterson, J.: A Privacy Mechanism for the Session Initiation Protocol (SIP) (2002), <http://cabernet.tools.ietf.org/html/rfc3323>
- Kazatzopoulos, L., Delakouridis, C., Marias, G.F.: Providing anonymity services in SIP. In: 19th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (2008)
- Reed, M.G., Syverson, P.F., Goldschlag, D.M.: Anonymous connections and onion routing. *IEEE Journal on Selected Areas in Communications* 16(4), 482–494 (1998)
- Chaum, D.L.: Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM* 24(2), 84–90 (1981)
- Stier, M., Eick, E., Koerner, E.: A Practical Approach to SIP, QoS and AAA Integration. *Integration The VLSI Journal* (2006)
- Metcalfe, R.: Metcalfe's Law. *IEEE Spectrum* 17(40), 53 (1995)
- Melchor, C.A., Deswarte, Y.: From DC-Nets to pMIXes: Multiple Variants for Anonymous Communications. In: Fifth IEEE International Symposium on Network Computing and Applications, pp. 163–172. IEEE Computer Society, Washington, DC (2006)
- Reiter, M., Rubin, A.: Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security* 1(1), 66–92 (1998)
- Levine, B.N., Shields, C.: Hordes: A multicast-based protocol for anonymity. *Journal of Computer Security* 10(3), 213–240 (2002)
- Gritzalis, D., Marias, G., Rebahi, Y., Soupionis, Y., Ehlert, S.: SPIDER: A platform for managing SIP-based Spam over Internet Telephony (SPIT). *Journal of Computer Security* 19(5), 835–867 (2011)
- Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., Schooler, E.: SIP: Session Initiation Protocol. RFC 3261 (2002), <http://tools.ietf.org/html/rfc3261>
- Kim, J., Baek, J., Kim, K., Zhou, J.: A Privacy-Preserving Secure Service Discovery Protocol for Ubiquitous Computing Environments. In: Camenisch, J., Lambri-noudakis, C. (eds.) EuroPKI 2010. LNCS, vol. 6711, pp. 45–60. Springer, Heidelberg (2011)

A Browser-Based Distributed System for the Detection of HTTPS Stripping Attacks against Web Pages

Marco Prandini and Marco Ramilli

Università di Bologna, DEIS, Viale del Risorgimento 2, 40136 Bologna, Italy
{marco.prandini,marco.ramilli}@unibo.it

Abstract. *HTTPS stripping* attacks leverage a combination of weak configuration choices to trick users into providing sensitive data through hijacked connections. Here we present a browser extension that helps web users to detect this kind of integrity and authenticity breaches, by extracting relevant features from the browsed pages and comparing them to reference values coming from different sorts of trusted sources. The rationale behind the extension is discussed and its effectiveness is demonstrated with some quantitative results, gathered on the prototype that has been implemented for Mozilla Firefox.

Keywords: HTTPS stripping, Peer-to-peer, Browser plugin.

1 Introduction

Stealing sensitive data from users is one of the most common targets pursued by attackers on the Web. There are many ways to lure users into providing their data over the wrong connection, leading to the attacker's server instead of the legitimate one. Eventually, the widespread usage of HTTPS seemed like the ultimate weapon against this kind of hijacking. However, the very success of HTTPS backfired as many high-traffic websites staggered under the computational load associated with serving every page through an encrypted connection. This led some sites to adopt a trade-off solution, foreseeing the usage of HTTPS only for the connections involving the transmission of sensitive data. However, the lack of integrity protection for the page containing the link for the submission opens a crack that an attacker can leverage to compromise the whole transaction. This paper illustrates a method for solving this problem based on a browser extension. In the following, section 2 details the attack; section 3 outlines the design principles of the proposed countermeasure; section 4 describes the extension implementation as a Mozilla Firefox plugin; finally section 5 draws conclusions.

2 Analysis of the Attack

Let's assume the common scenario in which a user on a client host (*CH*) wants to establish a secure transaction with a Web server on a server host (*SH*). Given that *CH* and *SH* must exchange data on the network, a Man In The Middle (MITM) attack is possible if the attacker host (*ATH*), by means of skillful manipulation of network devices, becomes a gateway for the traffic stream. The attacker intercepts the traffic from the source and



Fig. 1. Screenshot of the login box on the home page of a bank. Notice (a) that the page is served on HTTP, (b) the graphics suggesting a secure login process, and (c) the underlying HTML code, which sends data on HTTPS, that is, as long as a MITM attack does not modify it.

forwards it to the destination (and vice versa), preserving the illusion of *CH* and *SH* of being connected through an unaltered channel, but at the same time being able to modify messages and insert new ones. While this is not a completely trivial feat, there are sound reasons to worry about this possibility, if the attacker is on the same network of the victim but also if he is in a remote location, due to the insecure default configuration of many home access routers [52]. An attack to the professionally-managed infrastructure on the server side is less likely to succeed.

Any kind of MITM would fail if the very first page of the visited site is served on HTTPS (and the user checks it actually is!), because, with some exceptions [1], nobody can circumvent the cryptographic authentication and impersonate the real server. However, the initial page is usually the one responsible for a significant part of a web site traffic, and often is the starting point for a navigation through sections of the site that do not need protection. Thus, to avoid paying the high price associated with serving the first page on HTTPS, many sites use plain HTTP. Then, if the page contains a form for the user to provide identification data, the submission of the form is protected by pointing it to a HTTPS link, reassuring the user about the security of the process by means of graphical cues or textual explanations (Fig. 1).

However, the attacker is left free to become a MITM between *CH* and *SH* during the first, unprotected exchange of information. He can intercept the initial request/response between *CH* and *SH*, substituting HTTP for HTTPS in every link of the returned page before serving it to *CH*. When the browser on *CH* requests additional contents linked from the page, or submits a form, it actually makes a HTTP connection to *ATH*, where the attacker can read every byte in plaintext. The attacker then relays every request to *SH* using the correct protocol specified in the original page, to be sure of complying with the configuration of *SH*, and sends the decrypted response back to the browser; possibly, a favicon representing a secure lock is also injected (or crafted into the page), giving a false perception of a secure connection to the client.

The detailed implementation of this attack is described in [4].

3 The Proposed Countermeasure

All the browsers come with a default setting to alert users about to submit information over an insecure channel. This is a very effective countermeasure against the described attack. Unfortunately, web pages that submit user-provided, harmless information over an insecure channel are in the millions. Thus most users, after the first few false alarms, disable this check [7].

The proposed approach is to treat web pages like any other kind of potentially malicious content, subjecting them to the analysis of a security module very similar to anti-malware software, and comparing the content of the page against suitable information patterns to try and detect if a MITM has modified it. There are two key issues related to this approach, namely choosing a method to extract sensible page features and providing users with the reference features representing authentic pages.

The first issue arises because, nowadays, the vast majority of web pages are dynamically generated. They almost invariably include sections that change each time they are served. It is necessary to characterize a page by extracting only the invariant parts, but making sure that they represent all the contents whose integrity needs to be checked. The result should be a fingerprint of the page, a hash value that can be reliably computed each time the same page is visited and compared to a reference value computed over the authentic page. Then, the second issue comes into play. It is necessary to define how to provide the reference value to every user who is visiting a page in a trusted way.

Regarding the second issue, we envisaged three possible scenarios.

Local Database. In principle, each user can build a local database containing the reference values for the pages of his interest. While this method has the undeniable advantage of placing the user in full control of the database, it exhibits a significant drawback: the user must be absolutely sure that he is safe from the MITM attack when he computes the reference value.

Trusted Online Repository. If the users are willing to place their trust upon a third party of some sort, for example a directory, such a system can act as the authoritative source for computing and distributing reference values. This approach suffers from the usual drawbacks associated with putting a central entity in charge of essential functions: the entity itself becomes a very valuable target for attackers, who would be highly rewarded by a successful compromise of its database or even a simpler DoS attack.

Peer Exchange. At any given time, a web page is viewed by a set of clients. The more popular the page, the more interesting target it makes for an attacker, and the larger the set. Under the assumption that mass compromise of clients is unlikely, it is possible to share the reference values between every client through a peer-to-peer network, and to choose the most frequent value associated with a given URL as the correct one.

4 Prototype

We implemented the described solution as a browser plugin which can warn the user of a possible attack. The extension's architecture provides an easy means of porting the code on many different platforms, simply changing the browser-specific interface to the

core logic, written in Java. As of now, the SecureExtension (SecExt) plugin is available for Mozilla Firefox, chosen for being the most widespread open source browser, at <http://code.google.com/p/secureext/downloads/list>, and a usage demo can be viewed at <http://www.youtube.com/user/SecExt>. The plugin architecture is modeled around the three basic functions outlined in the general description: page characterization, page evaluation, and information sharing. The following paragraphs describe the detail of each phase.

4.1 Page Characterization

Web pages are usually composed of many different sections, including parts that are dynamically generated and thus differ each time the page is loaded. Trying to characterize a page by simply computing its hash with a message digest algorithm over its whole content would certainly fail to yield a sensible reference value. It would never be the same even if the page is authentic.

The process we devised for proper characterization starts by observing that, for our purposes, the only important kind of content is the set of links possibly pointing to the submission target of the login form, of other form collecting sensitive data from the user, or possibly opening such a form in a separate but closely related space (iframe, pop-up window, etc.). Every bit of the page which is not a URL is then discarded.

The characterization procedure then removes the parameters (i.e. anything following a “?” character, if present, that could make the same page look different each time it is loaded) from each URL. Their removal does not affect the reliability of attack detection, since the attacker aims simply at changing “https” into “http”. Actually, the URL cleaning could be pushed even further by removing everything but the protocol, host and port elements of the URL, to deal with sites that use “/” instead of “?” to have dynamic pages indexed by search engines, but we need further testing to decide whether the (rather small) increase in generality is worth the loss of captured information or not.

Finally, the string originated by the concatenation of the cleaned URLs is given as the input of a message digest algorithm, whose compact and fixed-size output is well suited to summarize the page characteristics.

A page can include code from separate sources, for example by means of `iframe` commands. The process can handle this possibility very easily: SecExt considers each piece of HTML code that can be referenced by a URL as an independent “page”. Let’s suppose that a separate piece of code is included by the main page to handle user login. If the main page is served on HTTP, the attacker will target the link pointing to the included code, and the attack will be recognized as a modification to the main page. If the main page is secured by HTTPS, but the included code is vulnerable to the stripping attack instead, the latter will be independently characterized and a successful attack against it will be explicitly reported.

4.2 Page Evaluation

Each time the user loads a page in the browser, the SecExt plugin computes its hash value according to the illustrated algorithm, then looks for records regarding the page

in the database (whose construction is detailed in the next section [4.3](#)). The query can yield different outcomes.

- No records are found for the page’s URL. No check can be made about the integrity status of the page. It is possible to envisage a plugin enhancement warning the user trying to submit data on HTTP from this kind of unverifiable pages. The evaluation of the consequences in terms of usability are under investigation.
- The hash of the current page matches the value most frequently associated with its URL in the database. SecExt deduces that most likely the browsed page has not been compromised through an HTTPS stripping attack.
- The hash of the current page does not match the value most frequently associated with its URL in the database. The current page then has a different content from the version most commonly seen in different times or places. The plugin alerts the user by visualizing a warning message on the screen. Before the user can interact with the browsed page he needs to confirm the warning message. Then it is up to the user browsing the page or not, possibly after in-deep verification of the underlying code.

4.3 Information Sharing

SecExt can build the database of hash values by composition of two different partial sources: a local database, containing only hashes computed by the local system, and a global database, which is itself a collation of the local databases shared by other users over a P2P network. The sum of these parts allows SecExt to leverage both local knowledge, possibly gathered in a controlled environment where the user can confidently assume to be safe from MITM attacks, and the same kind of knowledge gathered by users who run SecExt as well. In the latter case, we claim that a large enough user base will lead to the population of a global database containing a striking majority of hash values computed over pages which have not been tampered with.

The P2P network run in SecExt is based upon a Java implementation of the Chord protocol [\[6\]](#), chosen for this first prototype for its simplicity. The Chord daemon runs in a background process to keep the communication with peers active independently of the plugin activations. Chord exploits a distributed hash table to store key-value pairs by assigning keys to different computers (known as “nodes”); a node will store the values for all the keys for which it is responsible. Chord specifies how keys are assigned to nodes, and how a node can discover the value for a given key by first locating the node responsible for that key. In simpler terms, Chord lets the connected nodes to collectively build a virtual shared folder. Every peer shares its local database as a file, placed in the virtual folder, named by a unique node identifier. The file can get actually copied on other peers when they come online and search for new resources. The virtual global database that is the collation of all the local databases is then materially represented by a highly available collection of files, and the load to access it is spread among the peers.

4.4 Experimental Validation

We tested the SecExt plugin effectiveness in a lab environment. The results, which cannot be detailed here for space constraints, showed satisfactory detection rates and a

limited amount of false positives. An accurate judgment of our solution, however, must wait until some limitations regarding the security of the P2P exchange are solved and a real-world, wider testing campaign can be rolled out.

5 Conclusions and Future Work

We surveyed a large set of websites belonging mainly to financial institutions, which are particularly interesting for fraudsters looking for user credentials to steal, and found a significant fraction of them vulnerable to the HTTPS stripping attack. Since users cannot force webmasters to fix the problem where it should be fixed, we proposed a client-side, anti-malware-style approach to the detection of the attack. It leverages the distributed knowledge of a potentially large community of users to identify modified pages even if the user has never visited them before, exploiting peer-to-peer architectures to spread knowledge of the reference values representing unaltered pages without resorting to a trusted third party. We implemented the countermeasure as a plugin for Mozilla Firefox, and verified the practical feasibility and correctness of all its basic principles. The plugin was able to correctly characterize the pages used for testing, taking into account all the relevant data for evaluating its integrity but avoiding to include variable parts that could trigger false positives. Currently, we are working to achieve higher communications efficiency and better handling of updates through finer granularity, whereas for this first prototype we implemented the knowledge sharing as a distribution of the whole reference values database on the P2P network. We are also extending SecExt towards a more comprehensive architecture, to be able to easily “hook” different code-analysis modules into the core logic, timely adding new detection capabilities when new threats appear.

References

1. Dhamija, R., Tygar, J.D., Hearst, M.: Why phishing works. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2006, pp. 581–590. ACM, New York (2006)
2. Heffner, C.: How to hack millions of routers. In: Black Hat Conference 2010 (2010)
3. Nikiforakis, N., Younan, Y., Joosen, W.: HProxy: Client-Side Detection of SSL Stripping Attacks. In: Kreibich, C., Jahnke, M. (eds.) DIMVA 2010. LNCS, vol. 6201, pp. 200–218. Springer, Heidelberg (2010), doi:10.1007/978-3-642-14215-4_12
4. Prandini, M., Ramilli, M., Cerroni, W., Callegati, F.: Splitting the HTTPS stream to attack secure web connections. *IEEE Security and Privacy* 8, 80–84 (2010)
5. Stamm, S., Ramzan, Z., Jakobsson, M.: Drive-By Pharming. In: Qing, S., Imai, H., Wang, G. (eds.) ICICS 2007. LNCS, vol. 4861, pp. 495–506. Springer, Heidelberg (2007), 10.1007/978-3-540-77048-0_38
6. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. *SIGCOMM Comput. Commun. Rev.* 31, 149–160 (2001)
7. Sunshine, J., Egelman, S., Almuhiemedi, H., Atri, N., Cranor, L.F.: Crying wolf: an empirical study of SSL warning effectiveness. In: Proceedings of the 18th Conference on USENIX Security Symposium, SSYM 2009, pp. 399–416. USENIX Association, Berkeley (2009)

Privacy-Preserving Mechanisms for Organizing Tasks in a Pervasive eHealth System*

Milica Milutinovic¹, Vincent Naessens², and Bart De Decker¹

¹ KU Leuven, Dept. of Computer Science, DistriNet/SecAnon
`firstname.lastname@cs.kuleuven.be`

² Katholieke Hogeschool Sint-Lieven, Dept. of Industrial Engineering
`firstname.lastname@kahosl.be`

Abstract. In this paper, we describe privacy-preserving protocols for a scheduling service in eHealth applications. The scheduling mechanism that we propose protects sensitive information that are handled by the system. However, it still allows for a fair distribution of tasks and restricting the task assignment to caregivers with specific qualifications.

Keywords: eHealth, scheduling, privacy, fairness, caregiver, commercial.

1 Introduction

During the past decades, there has been a persistent trend in the age demographics of the developed countries. The average age of individuals is steadily increasing, and consequently there is a growing number of elderly that require continuous help. However, their guardians are often not in a position to provide this kind of daily assistance. Therefore, the potential of eHealth systems to provide efficient and cost-effective care in the home was recognized quite early.

In order to sustain certain independence of the elderly or patients, the home assistance systems should provide a wide range of services. These include continuous monitoring of health parameters, their automatic assessment and detection of anomalies, remote access to this data by authorized medical personnel, and communication with the caregivers. In addition to this, the patients or elderly need to be able to request assistance and these specific tasks should be assigned to their caregivers in a fair way. Examples are regular doctors' visits, catering or cleaning services, help with administering medication, etc. In order to take the burden of the close relatives of the patients, who are usually handling the organization of tasks, we explore its delegation to an eHealth system. Of course, for a pervasive system that will handle sensitive patient's data, such as health parameters or contacts with caregivers, one of the most important requirements

* This research is partially funded by the Interuniversity Attraction Poles Programme Belgian State, Belgian Science Policy, by the EU FP7 project NESSoS, by the Research Fund KU Leuven and by the IWT-SBO Project DiCoMas (Distributed Collaboration using Multi-agent System Architectures).

is protection of the privacy. Therefore, we are proposing privacy-preserving protocols that describe the scheduling service in a pervasive eHealth system. Additionally, the described protocols surpass the need to disclose any identifying information to the scheduling service, which allows for it to be offered by a commercial entity. Nevertheless, the patients are still able to specify required skills and medical qualifications and their preferences regarding caregivers.

The **contribution** of this paper is the design of a few protocols that allow the integration of a scheduling service into a pervasive but privacy-friendly eHealth system. In current eHealth systems or research, privacy is either not an issue or is at best protected through access control mechanisms. However, accidental or malignant leakage of privacy-sensitive data (e.g. by hackers or employees) can be detrimental to the patient's privacy. In our system, we ensure that such leakage cannot occur. All the necessary information is available, but is only accessible to authorized caregivers that are directly involved with the patient's care.

Due to space limitation we will not discuss related work. More extensive overview can be found in [2].

2 The System Architecture

An architecture for a pervasive eHealth system offering a wide range of services is proposed in [1]. Its high level description is given in this section and the development of the scheduling service is considered in this framework.

The proposed system design consists of four tiers. Patients' health parameters are continuously being measured by wearable, unobtrusive *sensors*. The recorded measurements are sent to and gathered by a *base station*, which represents a gateway towards the rest of the system. The base station can log and assess the measurements and request for a caregiver to be notified if a problem is detected. It also maintains patient-specific policies, which specify access control to stored data and thresholds and actions for normal, alert and emergency conditions.

The base station further communicates with a *dispatch centre* (DC), which can be a commercial organization that provides technical support and mediates the communication between the patients (i.e. the base stations) and the caregivers. It notifies the caregivers of tasks assigned to them or in case of emergency situations and follows up on their responses. It also relays the caregivers' requests to access the stored health-related data to the base stations. The data exchange is encrypted, in order to protect the information from the DC and its staff.

Finally, a separate entity, namely the *administration centre*, handles administrative tasks, such as user registration. This functionality is separated from the dispatch centre for privacy reasons. Since the dispatch centre mediates communication between patients and their caregivers, combining that information with the identities of all users would reveal sensitive data. For instance, knowing a specialist that is treating the patient reveals information about the patient's health. Therefore, all the users are identified in the dispatch centre only by their pseudonyms, but can be identified by a deanonymizing authority using the registration transcripts, when certain conditions are fulfilled.

Both dispatch and administration centres are equipped with tamper-free devices (TD_{DC} and TD_{AC} , respectively), which are used to re-key or decrypt sensitive information protected with their public key. These actions are possible only under certain conditions and are performed after strict checks in order to protect against attacks and prevent leakage of private information. For a detailed description of the functioning of these trusted devices, we refer the reader to [1].

2.1 System Functionality

Initially, all parties need to register with the administration centre. Upon registration, patients obtain a *smart card* recording their identity, address information and the service level agreement (SLA). The card also generates two public/private key pairs, one to be used for encryption, the other for signing. The public keys are then certified by the administration centre and the certificates are stored on the card. Similarly, the caregivers receive an *anonymous credential* certifying their identity, address information, medical qualifications and related proofs and a chosen random value which is not disclosed to the administration centre. The random value is used to generate provable unlinkable pseudonyms. Both patients and caregivers are then able to register with the dispatch centre.

When a patient registers with the dispatch centre (DC), her personal network is created and identified with her randomly-looking pseudonym. Linked with this pseudonym are a *vault*, i.e. an encryption of her real identity and address information obtainable only by the TD_{DC} , and a certificate issued by the TD_{DC} , linking the patient's pseudonym with her public keys. The trusted device only re-encrypts the information from the vault with the public key of an authorized party. The same holds for the caregivers' nodes. A caregiver anonymously approaches the dispatch centre when he is invited to join a patient's network. He then creates a new pseudonym as a function of the patient's pseudonym and the (secret) random number recorded in his credential. This way, he can pseudonymously authenticate with the DC, proving that the pseudonym is generated using his credential. Along with the caregiver's pseudonym, the DC stores his role¹ in the patient's network and a vault containing the identity, address, medical qualifications and appropriate proofs which only TD_{DC} can decrypt. Additionally, the caregiver generates a new key pair to be used for communication with the patient he is connected with and of which the public key is certified by the TD_{DC} .

The dispatch centre keeps the information about a patient's connections with her caregivers in the form of a network. These networks are patient-centric and both patients and caregivers are only identified with their pseudonyms.

3 The Scheduling Service

The scheduling service allows for fair allocation of patient-requested tasks to their caregivers, ensuring that every task is assigned and confirmed by a caregiver

¹ These roles are assigned by the patient at the time the connection is created. Examples are relative, neighbour, GP, specialist, cleaning service, etc.

with appropriate qualifications. It also allows the patients to specify the preferred caregivers or undesired ones. Therefore, the scheduling is offered by the system as a service the patients can subscribe to. The scheduling service can be offered by an external entity that registers with the dispatch centre. A patient can request to connect to it if she wants to use its services. In that case, the scheduling service becomes one of the caregivers of her network with an appropriate role. That is also true if the scheduling is offered by the dispatch centre itself.

3.1 Patient's Schedules and Caregiver's Profiles

The base station of every patient maintains a *schedule* of the caregivers' tasks. For every task, the schedule contains its identifier, the time frame, other details and the chosen caregiver. Until a caregiver confirms the assignment of a task, it remains conditional. For the task assignment, the base station may send a request to the scheduling service. However, the information about the tasks and the caregivers that is communicated to the scheduling service needs to be limited.

In order to allow the scheduling service to be offered by the system, caregivers specify their availability and store this information in a *profile* at the dispatch centre. They update this information regularly and the base stations or the scheduling service are able to retrieve the latest versions when a task is to be assigned. The profiles are linked with the caregiver's pseudonym and are stored encrypted with a secret key obtainable only by an authorized party (see further).

Besides the availability, the profiles contain other relevant information. They also state the caregivers' willingness to perform certain tasks and how long in advance they need to be notified about an assignment to be performed.

3.2 Protocol Assumptions

In the remainder of the text we will use the following assumptions about the creation of encrypted communication between two entities. All stakeholders in the system (patients and caregivers) interact via the dispatch centre (DC). This communication is SSL protected with server-side (DC) authentication. The initiating party (most often a caregiver) first pseudonymously authenticates with the DC. It then generates a fresh symmetric key and creates a *capsule*. The term 'capsule' denotes a symmetric key possibly concatenated with additional information, such as identifiers of the sending/receiving party, which is encrypted with the public key of the receiver. Next, the initiator's request (e.g. to access the patient's medical data) and the capsule are signed and sent to the DC. If the DC authorizes the request (based on the role of the sender), it relays the messages to the receiver. The receiving party then verifies the signature and extracts the session key from the capsule which is subsequently used for encrypting the communication between the two parties. We will refer to this encrypted communication between two parties as the creation of a *protected virtual link*. Once this link is established, all the messages DC relays are hidden from it.

A similar approach is used when some data needs to be stored at the DC in order to make it available to different parties. The entity that is sending the data

initially authenticates pseudonymously to the DC. It then generates a symmetric key with which it encrypts the data to be stored. The key is enclosed in a capsule along with the pseudonym of the sender and possibly the parties authorized to access the data. The capsule is encrypted with the public key of TD_{DC} . Both the encrypted data and the capsule are stored at the DC. The trusted device can make the protected data available to an authorized party by re-encrypting the capsule with this party's public key. However, this is only performed after thorough checks.

3.3 Scheduling a Task

If one or more tasks need to be assigned to the caregivers of a patient, the base station sends a request to the scheduling service via the dispatch centre. For every task, the base station creates task-assignment request containing the specified task, the time frame, the required caregiver's role and/or qualifications and preferred or undesired pseudonyms. This request is then signed and sent via a protected virtual link (see Sect. 3.2). It is then used by the scheduling service to prove its authorization to retrieve the profiles of the concerned caregivers. After verifying the request, signature and scheduling service's public key certificate, the trusted device re-encrypts the profiles' encryption keys for the scheduling service. If special qualifications are necessary for a task, the scheduling service can prompt TD_{DC} whether a particular caregiver has these qualifications. The TD_{DC} will check the caregiver's vault and reply 'Yes' or 'No'. When the scheduling provider obtains the profiles, it can assign the task, taking into account the required role of the caregiver and patient's preferences. Additionally, along with the initial request, the base station sends to the scheduling service relevant policies which are to be taken into account. Examples are limitation of hours that can be assigned to a role or a caregiver, restrictions on using commercial providers and additional requirements. In order to ensure a fair distribution of tasks, the base station also sends a summary (e.g. total number of hours) of current and past assignments for each of the caregivers. This way, the scheduling service can consider the load that is placed on each of the caregivers.

The assignments are then sent via a protected virtual link to the base station. The task assignments are now stored in the schedule, but remain conditional until the assigned caregiver approves the task.

3.4 Retrieval of Assignments

Every caregiver is assumed to be collecting his assignments at regular intervals (e.g. twice a week) via a web application. This dedicated application establishes a protected virtual link with the base station of the patient in order to retrieve the tasks for which the caregiver's pseudonym was assigned. The caregiver can then inspect them and reply, i.e. accept or reject the assignments. Subsequently, the web application needs to make changes to the availability information specified in his profile. For this, it authenticates pseudonymously to the DC, sends the new profile and corresponding capsule, both signed with the caregiver's anonymous

credential, and receives a receipt from the DC. This way, if a problem is detected, the DC can prove that the schedule was received from the caregiver, and the caregiver can also prove whether or not a version of the profile was sent by him.

However, when a caregiver wants to retrieve his complete profile, which might be the case if he is using the web application from a new device, strict checks are necessary. First, he will pseudonymously authenticate to the DC using his anonymous credential. The DC then asks the TD_{DC} to verify whether the pseudonym matches the one in the capsule. If this verification passes, the TD_{DC} re-encrypts the original capsule with the public key of the caregiver, which is sent to him with the profile. This way, only owners of the profiles are able to retrieve them.

3.5 Contacting a Caregiver

In case of emergency situations, a caregiver is urgently required to assist the patient. In this case, the base station chooses a caregiver itself, according to the patient's policies, so that no valuable time is lost. The base station sends an encrypted request to the dispatch centre to contact the chosen caregiver. The request can only be decrypted by TD_{DC} , which re-encrypts it for a calling module installed at the DC. For communication with the calling module, a symmetric key is used, which is embedded in the module through whitebox cryptography. The module sends the message specified in the request to the caregiver's phone number. This number is extracted by the TD_{DC} after appropriate checks, from the caregiver's vault stored at the DC. Once the message has been sent, the calling module will store the patient's pseudonym and a secure hash of the telephone number and delete any other information to protect from eavesdroppers. This hash is then compared to the incoming messages' origin. If the expected response is received, the calling module sends it encrypted to the base station.

For a detailed description of the protocols and evaluation of the security and privacy properties, we refer the reader to [2].

4 Conclusion

In this paper we have described a scheduling mechanism that can be integrated into a pervasive eHealth system. The focus of the design was preserving privacy of patients, but also their caregivers. Furthermore, the disclosure of information is performed on a need-to-know basis allowing the service to be offered by a commercial company, which is an important impetus for its deployment on a large scale.

References

1. Milutinovic, M., Decroix, K., Naessens, V., De Decker, B.: Commercially-run home assistance centres. Technical Report, vol. CW612, KU Leuven (2011)
2. Milutinovic, M., Naessens, V., De Decker, B.: Privacy-preserving scheduling mechanism for eHealth systems. Technical Report, vol. CW618, KU Leuven (2012)

Web Services Security Assessment: An Authentication-Focused Approach

Yannis Soupionis and Miltiadis Kandias

Information Security and Critical Infrastructure Protection Research Laboratory
Dept. of Informatics, Athens University of Economics & Business (AUEB)
76 Patission Ave., Athens, GR-10434 Greece
{jsoup, kandiasm}@aueb.gr

Abstract. Web services may be able to publish easily their functions to the rest of the web world. At the same time they suffer by several security pitfalls. Currently, there is limited research on how the proposed web-services security countermeasures affect performance and applicability. In this paper, we introduce the threats/attacks vs. web-services authentication, present the most widely used security method for protecting it, and identify the threats/attacks tackled by those methods. Moreover, we evaluate the web service authentication mechanism proposed in these implementations, not only on a theoretical level (by taking into consideration all the security issues of the implementing authentication sub-mechanisms), but also in a laboratory environment (by conducting extensive experiments). Finally we demonstrate the trade-offs between sophisticated web-service security methods and their performance.

Keywords: Security Assessment, Web Services, Authentication, Performance.

1 Introduction and Related Work

The explosive growth of the World Wide Web has introduced a wide array of new technological advances and several sophisticated end-user services. Development in data networks facilitated the introduction of web services [1] [2] [3], which have been increasingly penetrating the web market. Web services are a realization of a more abstract architectural style, called Service-Oriented Architecture. While web services introduce a variety of new functionalities, they also increase the complexity of the system and can be subject to a significant variety of attacks. Currently, there is limited evaluation on how web services security are practically enforced. Existing work focuses on high level aspects of security, namely business-oriented ones, while it has paid limited attention to technical aspects. A relevant publication shows an extension of the WS-Security standard [6]. Another publication [7] defines three security policy assertions provided in SOAP Message Security, WS-Trust and WS-Secure Conversation. The paper defines a basic set of assertions, which describes how messages are to be secured, without providing enough information on development or performance trade-offs. Both the above papers propose mechanisms to enforce

specific security aspects, but they provide no metric, capable of assessing a balance between performance and security.

In our paper, we provide a model capable of measuring the effectiveness and performance of security mechanisms and put in place with web services. In particular, we focus on the authentication process. A web service cannot function properly if the authentication mechanism is either compromised, or complex enough to inadequately downgrade the performance of the web service. Our approach is based on (a) identifying all possible attacks/threats of web services authentication, (b) assessing the key methods of securing web authentication, and (c) evaluating the system performance.

In Section 2, we present authentication in web services and we illustrate the web services authentication threats. We also show whether they are tackled by the authentication methods. In Section 3 we provide the results of a series of experiments on how different implementations of authentication mechanisms affect the system performance. Finally, we describe our conclusions, together with plans for future work.

2 Web Services Threats and Countermeasures

In addition to software development best practices that are proposed mainly by the various WS standards, numerous countermeasures exist. An authentication process consists of: a) the authentication data of the requesting entity, b) the channel through which this data is transmitted, and c) the data validation according to an authentication database.

The security countermeasures, which were chosen to protect the service, are: a) a username and a password (credentials), b) encryption over the channel, and c) password hashing [7]. Each of the above countermeasures may be implemented by different technologies, which are:

1. Password: a) One-Time-Password, b) good practice based password, c) based on no good practice (i.e., chosen once and valid forever, no rules about using special characters, etc.).
2. Channel: a) VPN, b) Asymmetric SSL, c) Symmetric SSL, d) unprotected channel (HTTP).
3. Password storage: a) MD5, b) SHA1, c) Plaintext.

The vulnerabilities introduced by the software implementing these methods, which can introduce new attacks to the service, are out-of- scope of this paper.

In order to study the effectiveness of the above security countermeasures, we produced a list of known web-service attacks. The list is based on WS standards [9-12] and research output. From the produced list of 24 attacks we excluded 7, because they do not affect authentication mechanism. In Table 1, if a countermeasure/mechanism deals with an attack, then the corresponding cell is checked. This table is a useful tool for those in charge of either the security or the development of a web service. Furthermore, based on the results of a risk assessment review, the analyst can choose a combination or a single countermeasure to tackle attacks assessed as important.

Table 1. Countermeasures vs. Threats

Mechanism Threats	Password			Channel				Password Storage		
	OTP	Good Practice Based	No Good Practice	VPN	Symmetric SSL	Asymmetric SSL	HTTP	SHA1	MD5	Plaintext
Replay attack	✓			✓	✓	✓				
Session hijack				✓	✓	✓				
Security tokens alteration/stealing	✓			✓	✓	✓				
System misconfiguration attacks										
Man-in-the-middle				✓	✓	✓				
Eavesdropping	✓			✓	✓	✓				
Password guessing	✓	✓		✓	✓	✓				
Cryptanalytic password attacks				✓	✓	✓		✓	✓	
Brute force attacks	✓	✓						✓	✓	
Denial of Service							✓			✓
Spoofing attacks	✓			✓	✓	✓				
Information gathering				✓	✓	✓				
Unauthorized access	✓	✓	✓	✓	✓	✓		✓	✓	
Insecure audit/ log - repudiation attacks	✓	✓	✓	✓	✓	✓				
Retrieval of clear text configuration				✓	✓	✓		✓	✓	
Encryption keys' theft				✓		✓				
SQL injection										

3 Experimentation Results

In this section we identify the performance issues raised when a secure mechanism is used for a web-service authentication. The performance issue depends on the resources consumed, as well as on the time needed for establishing a web-service session. In order to determine and evaluate how the authentication method affects the system performance, we had to record the corresponding values of the monitoring subjects (time, computational resources) during significant overload. The tests were performed so as to evaluate a web service authentication mechanism, strengthened by the countermeasures mentioned in Section 2. The password policy was not tested through experimentation, as it is based solely on human factor parameters and do not differentiate, in terms of computational cost. The purpose of the first two tests is to evaluate the performance of each countermeasure, separately. In the final test, we implemented combinations of countermeasures in order to get a more holistic view of the authentication web service's performance.

Mechanism Scenarios and Results. In order to implement the scenarios, we used ten external clients, which initiated web-service requests. The scenarios consisted of five phases. During the first phase, each external client made only one request. Thus, the web-service received a total of 10 requests. During the second phase, every external

client made 100 requests. Thus, the proxy received a total of 100 requests, while in the last one, with proportional way, the web-service received 105 requests.

Table 2. Channel type experimental performance

Requests		10	100	1.000	10.000	100.000
Average time per session (msec)	VPN	29,3	28,9	29,1	171,1	298,0
	Asymmetric SSL	27,4	27,6	27,4	188,2	282,3
	Symmetric SSL	20,4	21,9	20,7	112,1	204,4
	HTTP	12,0	12,0	12,0	28,0	125,0

Only one mechanism was evaluated for each scenario. The other mechanism was implemented so as to produce the minimum overload to the test. If the channel type was the one tested, then the password storage parameter was set permanently to plaintext. The results of the evaluation of the channel type are summarized in Table 2.

The average processing time, for a request to be properly handled, was <300 msec. The maximum time for handling a request was 2.5 sec. This appears to be excessively high. It happened when many requests were to be handled and thus a very high CPU and memory overload occurred. Therefore, if a web service expects many simultaneous requests, then a VPN implementation is not a good solution. The results produced by the evaluation of the password storage policy are summarized in Table 3.

Table 3. Password storage experimental performance

Requests		10	100	1.000	10.000	100.000
Average time per session (msec)	MD5	9,5	9,2	12,6	75,4	102,8
	SHA1	9,8	10,0	16,8	107,8	143,2
	Plaintext	7,1	6,9	6,7	8,1	12,4

The average processing time, for a request to be properly handled, is <150 msec. The maximum time for retrieving a password was 1.9 sec for the SHA1 and 1.3 sec for MD5. These time periods may not be always acceptable, as several applications that cooperate with the web-services have lower timeouts. The long time is needed in case there were many requests to be handled. In this case, there was a high I/O to the hard drives, while both CPU and memory were overloaded.

Extended Test Scenarios and Results. In order to implement these scenarios, we used ten external clients, which initiated web-service requests. The extended test scenario consisted of three phases (1000, 10000, 100000 requests). During these tests, all authentication sub-mechanisms were implemented, in all combinations. In Figure 1, we illustrate the most important fraction of these implementations, as well as the average time needed to serve an incoming request. Our main conclusions are: a) the symmetric SSL implementation is from 15-20% quicker than asymmetric, b) the performance is

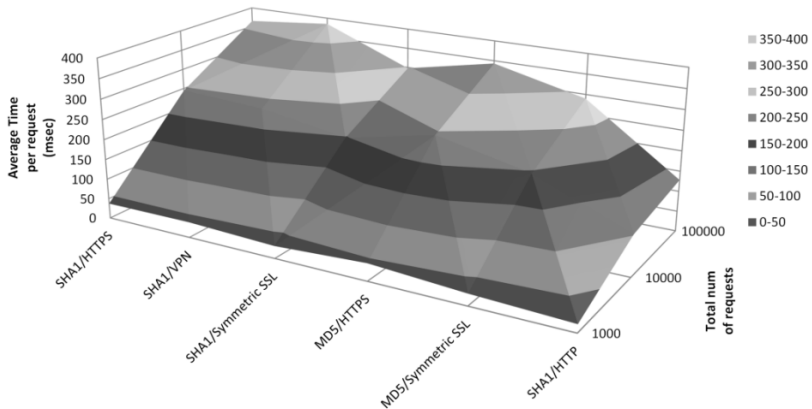


Fig. 1. The average time needed for serving a web-service request

slightly affected by the use of SHA1 or MD5 (5%), and c) the non-encrypted channel improves significantly the web service performance.

Taking into consideration the above mentioned conclusions, as well as the maximum time period to serve a request, which is ~3 sec (except of SHA1/HTTP), it turns out that the web-service providers should use these results when selecting which countermeasure to use for strengthening a web-service authentication mechanism.

Concluding, the main questions a web-service provider should answer, before hardening the a web service security, are: a) how many requests are going to be received?, b) is it possible for the web-service and the participating machines to handle the key management for a symmetric SSL implementation?, c) should all the transmitted data between the corresponding entities, apart from authentication process, be encrypted in order to justify the VPN overload?, and d) is the processed web-service data useful enough to worth the additional overload of the encrypted channel?

4 Conclusions and Further Research

In this paper, we described a security assessment capable of evaluating web service authentication mechanisms. Based on security mechanisms, such as communication channel type, password storage, and password alteration policy, we managed to identify how these mechanisms deal with the corresponding attacks. Then, we evaluated the mechanisms by implementing a web-service. Our experimentation aimed at determining the performance in various combinations of the authentication implementations. We showed that security countermeasures/mechanisms selection should be based not only on their ability to tackle attacks, but also under the prism of their computational cost. Thus, web services, which are expected to serve requests under heavy load, must be developed with this parameter in mind. The correlation between security and computational cost can be used to explain to security unaware stakeholders the requirements put and the decisions made. Our assessment proved to be a useful tool not only for those implementing web-services, but also for those who

make management decisions. Further research could extend the proposed security assessment with additional goals, such as data integrity, non-repudiation, or authorization. As the additional security-related goals may add complexity, we may adopt different methods for security assessment, e.g., entropy algorithms [8], or take into consideration information for detected authentication vulnerabilities through formal analysis [14]. Finally, we intend to study the aforementioned characteristics in composed web-service environments (mashups) [5],[13].

Acknowledgments. This work was performed in the framework of the SPHINX (09SYN-72-419) Project, which is partly funded by the Hellenic General Secretariat for Research and Technology (<http://sphinx.vtrip.net>).

References

1. Papazoglou, M.: Web Services and Business Transactions. *World Wide Web* 6(1), 49–91 (2003)
2. Erl, T.: *Service-Oriented Architecture: Concepts, Technology and Design*. Prentice Hall (2005)
3. Erl, T.: *Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services*. Prentice Hall (2004)
4. Alonso, G., Casati, F., Kuno, H., Machiraju, V.: *Web Services: Concepts, Architecture and Applications*. Springer (2004)
5. Zeng, L., et al.: QoS-Aware Middleware for Web Services Composition. *IEEE Transactions on Software Engineering* 30(5), 311–327 (2004)
6. Vedamuthu, A.S., et al.: Web Services Policy 1.5 - Framework. W3C Recommendation (2007), <http://www.w3.org/TR/ws-policy/> (last visit January 5, 2012)
7. OASIS Standard, WS-SecurityPolicy 1.3 (2009), <http://docs.oasis-open.org/ws-sx/ws-securitypolicy/v1.3/os/ws-securitypolicy-1.3-spec-os.html> (last visit January 5, 2012)
8. Serjantov, A., Danezis, G.: Towards an Information Theoretic Metric for Anonymity. In: *Privacy Enhancing Technologies (PETS)*, pp. 41–53 (2002)
9. Nadalin, A., et al.: OASIS WS-Trust 1.4, OASIS (2008)
10. Nadalin, A., Kaler, C., Monzillo, R., Hallam-Baker, P.: *Web Services Security: SOAP Message Security 1.1 (WS-Security 2004)*, OASIS Standard (2006)
11. Vedamuthu, A., Orchard, D., Hirsch, F., Hondo, M., Yendluri, P., Boubez, T., Yalcinalp, U.: *Web Services Policy 1.5 - Attachment*. W3C Recommendation (2007)
12. Nadalin, A., et al.: *WS-SecureConversation 1.3*, OASIS Standard (2007)
13. Rosenberg, F., Khalaf, R., Duftler, M., Curbera, F., Austel, P.: *End-to-End Security for Enterprise Mashups*. In: Baresi, L., Chi, C.-H., Suzuki, J. (eds.) *ICSOC-ServiceWave 2009*. LNCS, vol. 5900, pp. 389–403. Springer, Heidelberg (2009)
14. Basagiannis, S., Katsaros, P., Pombortsis, A.: *Intrusion Attack Tactics for the Model Checking of e-Commerce Security Guarantees*. In: Saglietti, F., Oster, N. (eds.) *SAFECOMP 2007*. LNCS, vol. 4680, pp. 238–251. Springer, Heidelberg (2007)

Open Issues and Proposals in the IT Security Management of Commercial Ports: The S-PORT National Case

Nineta Polemi* and Theodoros Ntouskas

Department of Informatics, University of Piraeus,
Karaoli & Dimitriou 80, 185 34 Piraeus, Greece
{dpolemi, tdouskas}@unipi.gr

Abstract. Commercial ports are large scale infrastructures which their Information and Telecommunication (PIT) systems offer critical services and host sensitive data. However the current maritime legislation or standardization efforts do not sufficiently cover the IT security of the commercial ports. Identifying these needs we propose a collaborative environment offering security management services including a targeted risk management methodology which will help commercial ports to self manage their security.

Keywords: Security Management, Commercial Ports, Maritime environment, Critical Infrastructures, Collaboration, S-PORT project.

1 Introduction

The tragic accident of Titanic (1912) imposed the new concept of "safety" in the maritime sector and numerous legislations and directives are published since then concentrating in the physical protection of the ships, crew, passengers, cargo and sea. The terrorist events in New York and Washington (2001), Madrid (2004), and London (2005) imposed the concept of "security" in the maritime sector with additional directives and legislation (e.g. IMO / ISPS) in order to avoid such malicious acts. However these new efforts concentrated on the organizational and auditing aspects of physical security of the ships, maritime companies and the commercial ports; they did not consider the IT and cyber security aspects of the maritime environment.

In this paper we will concentrate on the ports and specifically on the security management of Ports' Information and Telecommunication (PIT) systems. The gaps and barriers of the existing security management maritime related legislation, methodologies and tools in the PITs will be presented at this paper. Proposals will be provided that have been partially presented in [3], [19], [15], [18] by the authors.

* Corresponding author.

2 Existing Efforts and Open Issues

The maritime environment is a complex one, as shown in Figure 1, involving and interacting with many entities including ports, ships (with passengers, crew, cargo), port authorities, maritime and insurance companies, customs, ship-industry, banks, ministries, other commercial providers, other critical infrastructures (e.g. railroads, airports) hosting and interacting with complex, heterogeneous Information and Telecommunication (IT) Systems. Commercial ports, are the central entities in the maritime environment and among the transportation critical infrastructures [2] since they are large-scale infra-structures that their degradation, interruption or impairment of their IT Systems has serious consequences on national security, health, safety, economy and welfare of citizens and nations characterized with multiplicity of interdependencies between them and the other entities in the maritime environment.



Fig. 1. Maritime Environment

The Ports Information and Telecommunication (PIT) systems consist (as all IT systems) of the following seven (7) consecutive layers: *Infrastructure* (e.g. buildings, platforms, servers, stationary / mobile terminals, nautilus systems, databases); *Telecom Systems* (e.g. networks equipment, satellites, Geographic Information Systems - GIS); *Software and Manuals* (e.g. identification, maritime, navigation, routing software, Enterprise Resource Planning -ERPs, ticketing); *Information and electronic data* (e.g. marine and coastal data, trade data); *Services* (e.g. invoicing, navigation, luggage/cargo management, logistics, e-health); *Other equipment* (e.g. fire alarm systems, cameras); *Users*: a. internal users (e.g. administrators, personnel), external users (e.g. port authorities, maritime companies, customs, insurance companies, IT and commercial provides), objects (e.g. ships, crew cargo, luggage, vehicles).

However the existing maritime security standards, methodologies and tools concentrate only on the physical security of the ports (safety) especially in the access control of the ports' infrastructures in relation to the safety of the ships. They consider only the first and the last layer of the PIT system and only the

two last components (access control, availability) of security ignoring all its other layers and components making ports as weak security links.

To be more specific, as it has been described in [18] and [15], the various maritime standardization bodies (e.g. IMO, EMSA, EASA, TEN-T EA) do not include in their memorandum IT/cyber security. Similarly, existing risk assessment methodologies for ports (e.g. MSRAM, MARISA, CMA) only detect safety risks and not IT risks.

Furthermore they do not consider the threats rising from the interdependency of the ports with the other ports or other critical infrastructures (e.g. railways) which may cause cascading effects. They do not examine the PIT security independently but with relation to ships' safety. The implementation of the safety directive IMO/ISPS is left at national level i.e. there is not a standard providing the exact procedures and measures that need to be undertaken (e.g. like the ISO27002) in order for a port to become IMO/ISPS compatible. Finally all maritime security management methodologies are not user centric; they consider the user as a passively involved entity that needs to follow the methodology with no interaction or collaboration.

3 Proposals for Enhancing Maritime IT Security Management

A proposed solution to better address the PIT security is to combine the IMO / ISPS with common IT security management standards. However the well known security management standards from the standardization bodies NIST [8] and ISO ([4], [5], [6]) will need further modifications in order to adapt to the port IT security so they can address specific sectoral threats, i.e. interdependent threats rising from all entities in the maritime environment (see Fig.1), specific threats (e.g. weather conditions, strikes) and maritime legislation (e.g. IMO/ISPS). Well known security management methodologies (e.g. OCTAVE, CRAMM, MAGERIT, MEHARI, COBIT) [18], [15] which have been successfully applied in various environments (e.g. health, business) they may be useful in ports' environments with appropriate adoptions.

Furthermore we need to consider the fact that ports are critical infrastructures (CIs) and we need to take into account the known legislation, recommendations and standards for CIs. Most of these standards are from the energy sector and more specifically from the U.S. Department of Energy, North American Reliability Corporation (NERC) [9] and the USA national program National Infrastructure Protection Plan [25]. There is not a standardized way in examining the criticality of an infrastructure or/and addressing cyber threats.

Commercial Ports belong in the special category of transportation CIs [24]. The various security management efforts [23], found in this sector-specific, focus on the physical security of the transportation means (targeted to railways), on material transportation and assessment of hazardous material. These security management methodologies for the CIs are at national level and use different impact assessment criteria (e.g. economic, public, health & safety, psychological

/ social / public confidence, complexity, environmental, business, national / territorial security). These methodologies may be used in the ports with appropriate modifications.

A holistic approach to security management of the PIT systems that the authors propose is the creation / enhancement of maritime standard(s) respecting and be compliant with: the ISPS code, modified IT and CIs security management standards.

4 S-PORT: A National Project

In the national S-PORT project [19], the ports are viewed as CIs and it addresses the following two main security management needs: The development of a targeted security management collaborative methodology for the PIT-systems based on IT security management standards and the ISPS code involving all PIT users; The promotion of collaboration and interaction among PIT users in the security management of the PIT-systems in an automated user friendly approach.

The first need is addressed by S-PORT by applying the STORM-RM collaborative risk management methodology [13], [16] which views risk management as a collaborative decision making problem and combines the Analytic Hierarchy Process (AHP) [22], the security management standards ISO27001 [4] and AS/NZS 4360 [1] and has been modified in order to address the specific needs of PIS and the requirements of the IMO/ISPS directive. In STORM-RM a multicriteria collaborative decision making technique is applied enabling all users (internal and external) to evaluate the impacts depending upon their experience and role in their interaction with the IT system under assessment (where most of the existing risk management methodologies enable only the security team to be involved in the impact evaluation). The STORM-RM has been parameterised for the PIT-systems [14], [20] considering sectoral PIT and cyber threats, setting criteria in the impact evaluation rising from the fact that ports are CIs and assigning roles/ opinion weights according to the ports' users. The basic goal of this parameterised methodology is the collection of knowledge and experience from all port's users (i.e. managers, administrators, security team, local users, cooperate users) in order to evaluate the impacts, sectoral threats, vulnerabilities and risks more accurately.

The second security management need is addressed in S-PORT by developing a collaborative environment [14] (a recent trend in Web services applied in various sectors e.g. e-commerce [10], [11], e-government [7] and e-migration, [17]) using innovative Web2.0 technologies. The S-PORT environment [14], [15] is a parameterisation of the STORM security management environment [12] addressing the specific needs of PIT and involving all PIT users in the security management procedures.

In the S-PORT collaborative environment we embedded the targeted risk management methodology offered as an S-PORT collaborative service to the PIT-users. More specifically, each phase of S-PORT-RM methodology has been implemented as a distinct module in the S-PORT environment. Additional S-PORT

collaborative security management services will be offered thru the S-PORT environment including cartography: graphical representation of the PIT infrastructure and identification of all assets of the PIT-systems; reporting: generation of security documents, e.g. security policies, business continuity plans, disaster recovery plans (as growing documents); collaborative Web2.0 services: forums, chat rooms, questionnaires, for building social interactions in resolving and discussing security issues.

5 Conclusions and Acknowledgments

The fact that commercial ports are critical infrastructures and their Information Systems offer critical services and host sensitive data, makes security management a necessary concern for their business continuity and productivity. In this context, S-PORT environment is expected to become a prototype of the next generation Information Security Managements Systems, being capable of providing high levels of confidentiality, reliability, interactivity and interoperability, for the critical infrastructures of the commercial ports. This work has been performed in the framework of the S-PORT project [21] and the authors would like to thank the GSRT (General Secretariat for Research and Technology Development Department) for funding the S-PORT project and the S-PORT partners.

References

1. AS/NZS 4360: Risk management standards australia. Strathfield (1999)
2. Brunner, E., Suter, M.: International CIIP Handbook 2008/2009: An Inventory of 25 National and 7 International Critical Infrastructure Protection Policies. Center for Security Studies, ETH Zurich, Switzerland (2008)
3. ENISA: workshop on cyber security aspects in the maritime sector (2011), <http://www.enisa.europa.eu/act/res/workshops-1/2011/cyber-security-aspects-in-the-maritime-sector>
4. ISO/IEC 27001: Information technology - security techniques - information security management system - requirements (2005), <http://www.iso.org>
5. ISO/IEC 27002: Information technology - security techniques - code of practice for information security management (2005), <http://www.iso.org>
6. ISO/IEC 27005: Information technology - security techniques - information security risk management (2008), <http://www.iso.org>
7. Karantjias, A., Polemi, N.: An innovative platform architecture for complex secure e/m government services. Int. J. Electronic Security and Digital Forensics (IJESDF) 2, 338–354 (2009)
8. National Institute for Standards and Technology: Risk management guide for information technology systems. NIST Special Publication 800-30, <http://csrc.nist.gov/publications/PubsSPs.html> (accessed October 15, 2011)
9. North American Reliability Corporation (NERC), <http://www.nerc.com> (accessed December 7, 2011)

10. Ntouskas, T., Papanikas, D., Polemi, N.: A collaborative system offering security management services for SMEs/mEs. In: Bashroush, R., et al. (eds.) 7th IEEE International Conference in Global Security, Safety and Sustainability (ICGS3 2011). Springer, Thessaloniki (August 2011) (to appear)
11. Ntouskas, T., Papanikas, D., Polemi, N.: Trusted collaborative services for the IT security management of SMEs/mEs. *Int. J. Electronic Security and Digital Forensics (IJESDF)* (to appear)
12. Ntouskas, T., Pentafronimos, G., Papastergiou, S.: STORM - Collaborative Security Management Environment. In: Ardagna, C.A., Zhou, J. (eds.) WISTP 2011. LNCS, vol. 6633, pp. 320–335. Springer, Heidelberg (2011)
13. Ntouskas, T., Kotzanikolaou, P., Polemi, N.: Impact Assessment through Collaborative Asset Modeling: The STORM-RM approach. In: 1st International Symposium & 10th Balkan Conference on Operational Research, Thessaloniki, Greece (to appear)
14. Ntouskas, T., Polemi, N.: A secure, collaborative environment for the security management of port information systems. In: Proceedings of the Fifth International Conference on the Internet and Web Applications and Services, ICIW 2010, pp. 374–379. IEEE Computer Society Digital Library, Barcelona (2010)
15. Ntouskas, T., Polemi, N.: Collaborative security management services for port information systems. In: International Conference on e-Business, ICE-B 2012, Rome, Italy (submitted, 2012)
16. Ntouskas, T., Polemi, N.: STORM-RM: A collaborative and multicriteria risk management methodology. *Int. J. Multicriteria Decision Making (IJMCDM)* (to appear)
17. Pentafronimos, G., Karantjias, A., Polemi, N.: Odysseus: An advanced, collaborative and trusted framework for the provision of migration services. In: Proceedings of the Fifth International Conference on the Internet and Web Applications and Services, ICIW 2010, pp. 531–537. IEEE Computer Society Digital Library, Barcelona (2010)
18. Polemi, N.: Security management of the ports' information systems. ENISA Personal study (to appear)
19. S-PORT Deliverable 1.2: State of the art and user requirements in the Port Information and Telecommunication (PIT) Systems Security, <http://s-port.unipi.gr/>
20. S-PORT Deliverable 1.4: Port Information and Telecommunication (PIT) Systems Security requirements-A targeted PIT-risk assessment methodology, <http://s-port.unipi.gr/>
21. S-PORT Project: A secure, collaborative environment for the security management of Port Information Systems, <http://s-port.unipi.gr/>
22. Saaty, T.L.: Decision making with the analytic hierarchy process. *Int. J. Service Sciences* 1, 83–98 (2008)
23. Theoharidou, M., Kandias, M., Gritzalis, D.: Securing transportation-critical infrastructures: Trends and perspectives. In: Bashroush, R., et al. (eds.) 7th IEEE International Conference in Global Security, Safety and Sustainability (ICGS3 2011). Springer, Thessaloniki (August 2011) (to appear)
24. Transportation Security Administration: Critical infrastructure and key resources sector-specific plan as input to the national infrastructure protection plan. Dept. of Homeland Security, USA (2007)
25. US Dept. of Homeland Security: National Infrastructure Protection Plan (2009), <http://www.dhs.gov/> (accessed December 2010)

A Response Strategy Model for Intrusion Response Systems

Nor Badrul Anuar^{1,2}, Maria Papadaki¹, Steven Furnell^{1,3}, and Nathan Clarke^{1,3}

¹ Centre for Security, Communications and Network Research (CSCAN),
Plymouth University, United Kingdom
info@cscan.org

² Faculty of Computer Science and Information Technology, University of Malaya,
Kuala Lumpur, Malaysia
badrul@um.edu.my

³ School of Computer and Security Science, Edith Cowan University, Perth, Western Australia

Abstract. There are several types of security systems, which focus on detecting, mitigating and responding to incidents. Current response systems are largely based on manual incident response selection strategies, which can introduce delays between detection and response time. However, it would be beneficial if critical and urgent incidents are addressed as soon as possible before they jeopardised critical systems. As a result, the Risk Index Model (RIM) has been proposed earlier in our previous study, as a method of prioritising incidents based upon two decision factors namely impact on assets and likelihood of threat and vulnerability. This paper extends RIM by using it as the basis for mapping incidents with various response options. The proposed mapping model, Response Strategy Model (RSM) is based on risk response planning and time management concepts and it is evaluated using the DARPA 2000 dataset. The case study analysis upon the dataset has shown a significant result in mapping incident into different quadrants. In particular, the results have shown a significant relationship between the incident classification with incident priorities where false incidents are likely to be categorised as low priority incidents and true incidents are likely to be categorised as the high priority incident.

Keywords: intrusion response systems, risk response planning, response strategy model.

1 Introduction

At present, there are several types of security systems like Intrusion Detection Systems (IDSs), Intrusion Prevention Systems (IPSs) and Intrusion Response Systems (IRSs), which focus on detecting, mitigating and responding to incidents. There are many well-known and widely-recognised response options in mitigating incidents such as blocking traffic, terminating user and notifying system administrators. Therefore, to mitigate incidents effectively, it is important to have a proper response strategy in place, in order to minimise the delay between detection and response.

Hence, an automated approach is preferred. To illustrate the importance of timely response, [1] highlights that the longer the delay between detection and response, the higher the attack success rate is. Therefore, it is important that critical and urgent incidents are addressed as soon as possible before critical systems are jeopardised.

The paper is organised as follows. Section 2 presents the related works that have had significant impact to this paper. Section 3 presents the proposed response strategy model. Section 4 presents a case study to illustrate how the model can be used. Finally, the last section concludes the paper.

2 Related Work

There are two types of decision-making models in response selection; a static and a dynamic mapping [2]. To mitigate incidents, the dynamic mapping provides more efficient results in comparison to the static mapping. In addition to the dynamic mapping in a response model, different response strategies are adopted, such as response goal strategy [2, 3], response stopping power [4] and adaptive response strategy [5, 6].

An article in [3] proposed a response goal strategy where a sequence of actions (also called subtasks) is arranged to achieve a specific goal. One or multiple goals need to be selected from the list and normally the selection of them is done manually by security analysts. Similarly with [3], a proposal in [2] developed an automated intrusion response system by adopting the hierarchical task network planning approach in their response decision-making model. An approach in [4] used a rule based module to identify the most appropriate response characteristics based on a Response Policy. The policy aimed to determine the most appropriate Response Phases and it is similar to the response goals used in [3]. Furthermore, the cost-sensitive model proposed in [5] applied an adaptive response strategy and updated response options based upon the status of the previous triggered response and the value of cost as a decision factor. Their approach closely follows the approach proposed in [6].

This paper proposes an alternative approach by considering risk assessment as decision factors in Risk Index Model [7], risk response planning as well as time management concept in improving the timeliness of response. In addition, this paper improves the response strategy by grouping incidents into a similar group based on their priority and it also allows a simultaneous response. With a static policy with a dynamic decision-making, the proposed model reduces the delay problem upon making an appropriate decision and response; hence it is more suitable and practical to be applied in live traffic network with online monitoring systems.

3 Response Strategy Model (RSM)

This paper presents Response Strategy Model (RSM) which can be used as an alternative to the existing model that applied a dynamic response mapping model. The model creates a relationship between incidents and different types of response option with different levels of priority. Based upon attack metrics and system states as

decision factors proposed in the earlier proposal in [7], this paper uses an alternative approach in exclusively mapping an appropriate response option with an appropriate incident by considering risk response planning and a time management concept in addressing the importance of response time. In addition, this paper proposes the response strategy by grouping incidents into a similar group based on their priority and it also allows for a simultaneous response.

The time management concept applied in RSM aims to create effective responses to critical incidents. In time management concepts, [8] presents four categories of tasks which are mapped onto four different quadrants; Q1: important and urgent, Q2: important but not urgent, Q3: not important but urgent, and Q4: not important and not urgent. However, in order to fit with the model, this paper modifies the quadrant to address the time management in responding to incidents. Instead of using “important”, this paper uses “critical” to show the relationship between time and impacts; therefore, the new quadrants consider the combination between urgent and critical incidents. The quadrants for the time management concept contain four different quadrants as tabulated in Table 1.

In order to establish a strategic RSM, this paper uses the risk response planning concept. It contains four different strategies: avoidance, transfer, mitigation and acceptance. According to [9], the risk response planning can be prioritised where avoidance can be the first option followed by transfer, mitigation and acceptance. With the response categorisation proposed by our study in [10], Table 1 shows the relationship map between them and their correspondent quadrants as well as some related examples for their response options.

Table 1. Response Strategy Planning with Response options

Risk Response Planning (Threshold)	Quadrants	Response options
Avoidance (0.75-1.00)	1 st Quadrant: Urgent incident and for a critical asset	<ul style="list-style-type: none"> • Block users, processes or network traffic in preventing future attacks. • Adjust users, processes or network traffic configuration in minimising impacts but maintain system’s performances.
Mitigation (0.50-0.75)	2 nd Quadrant: Not an urgent incident but for a critical asset	<ul style="list-style-type: none"> • Collaborate with other appliances by limiting users, processes or network traffic for delaying the process of attacks (Example: using access control, firewall, enabling other countermeasures or antivirus). • Terminate users, processes or network traffic in preventing continuous attacks (Example: locking OS, resetting connection, dropping user and killing process).

Table 2. (continued)

Transfer (0.25-0.50)	3 rd Quadrant: Urgent incident but for a noncritical asset	<ul style="list-style-type: none"> • Collect information about incidents for passive responses, proactive responses as well as forensic evidence (Example: trace connections, decoy systems, honeypots, forensic evidence, recovery, incidents' blacklisting and white listing). • Escalate to administrator for a further investigation (Example: attack verification, damage recovery and assessment).
Acceptance (0.00-0.25)	4 th Quadrant: Not an urgent incident and not for a critical asset	<ul style="list-style-type: none"> • Establish passive responses like enabling a notification via syslog, console alert, email, pager, PDA or mobile.

4 Case Study

In order to investigate the effectiveness of the proposed model, this paper discusses one case study and aims to satisfy two goals. Firstly, the case study used to investigate the distribution of incidents using RSM in comparison with other approaches like CVSS v2 [11] and Snort Priority [12]. Secondly, the case studies used to investigate the relationship between the response strategies and its ability to differentiate between true and false incidents in the classification of incidents. The case study were conducted using the MIT 2000 DARPA (i.e. LLDOS 1.0) intrusion detection data set [13]. In order to further discuss the case study, this paper uses the incident rating results from a prioritisation model, Risk Index Model (RIM) [7] and also use the rating threshold as in Table 1.

4.1 Case Study Results

Table 2 shows the distribution of incidents using the DARPA dataset. It contains 1,068 incidents. The first column on the left refers to the phases of the dataset and is followed by the total of incidents. The incidents are divided into two classes of incidents: true and false incidents. The other columns summarise the percentage of incidents with regards to specific rows; either they are true or false incidents. The distribution is separated between Snort Priority, CVSS v2 and the Response Strategy Model. The Snort Priority is divided into 3 different priorities which are high, medium and low. With CVSS v2, there are three main categories and there are high, medium and low. The last column is an additional column and it refers to incidents without priority. Response Strategy Model (RSM) divides its priority into four quadrants, including avoidance, mitigation, transfer and acceptance.

In general, a total of 904 incidents or 84.64% of incidents are considered as true incidents and this includes critical incidents as well as non-critical incidents. Only 15.36% or 164 incidents are considered as false incidents. As can be seen in the table, there is a clear distribution between true and false incidents. With RSM, an average of 92.68% of the false incidents was prioritised as the lowest quadrant in the acceptance strategy column. This percentage is better compared to only 67.07% of the false incidents being prioritised under low priority with Snort Priority.

Table 3. With the DARPA datasets

Phase	Time	No. of Incidents	Type	No. of Incidents	Snort Priority			CVSS v2				Response Strategy Model			
					High	Medium	Low	High	Medium	Low	None	Avoidance	Mitigation	Transfer	Acceptance
Pre 1	09:21:36 - 09:51:35	25	TRUE	-	-	-	-	-	-	-	-	-	-	-	-
			FALSE	25	-	36.00%	64.00%	-	-	-	100.00%	-	-	4.00%	96.00%
1	09:51:36 - 09:52:00	40	TRUE	40	-	-	100.00%	-	-	-	-	-	-	7.50%	92.50%
			FALSE	-	-	-	-	-	-	100.00%	-	-	-	-	-
Pre 2	09:52:01 - 10:08:06	21	TRUE	-	-	-	-	-	-	-	-	-	-	-	-
			FALSE	21	-	14.29%	85.71%	-	-	-	100.00%	-	-	-	100.00%
2	10:08:07 - 10:18:05	243	TRUE	224	-	67.86%	32.14%	33.93%	32.14%	-	33.93%	-	17.86%	43.30%	38.84%
			FALSE	19	-	15.79%	84.21%	-	-	-	100.00%	-	-	-	-
Pre 3	10:18:06 - 10:33:09	4	TRUE	-	-	-	-	-	-	-	-	-	-	-	-
			FALSE	4	-	100.00%	-	-	-	-	100.00%	-	-	-	-
3	10:33:10 - 10:35:01	64	TRUE	60	-	100.00%	-	46.67%	-	-	53.33%	-	46.67%	23.33%	30.00%
			FALSE	4	25.00%	75.00%	-	25.00%	-	-	75.00%	-	-	-	-
Pre 4	10:35:02 - 10:50:00	28	TRUE	-	-	-	-	-	-	-	-	-	-	-	-
			FALSE	28	-	35.71%	64.29%	-	-	-	100.00%	-	-	-	-
4	10:50:01 - 10:50:54	10	TRUE	8	100.00%	-	-	-	-	-	100.00%	-	-	50.00%	50.00%
			FALSE	2	-	100.00%	-	-	-	-	100.00%	-	-	-	-
Pre 5	10:50:55 - 11:26:14	12	TRUE	-	-	-	-	-	-	-	-	-	-	-	-
			FALSE	12	-	66.67%	33.33%	-	33.33%	-	66.67%	-	-	-	33.33%
5	11:26:15 - 11:34:21	579	TRUE	572	-	100.00%	-	-	-	-	100.00%	-	-	-	100.00%
			FALSE	7	-	42.86%	57.14%	42.86%	-	-	57.14%	-	-	100.00%	-
Post 5	11:34:22 - 12:35:48	42	TRUE	-	-	-	-	-	-	-	-	-	-	-	-
			FALSE	42	-	19.05%	80.95%	-	-	-	100.00%	-	-	-	-
Total	09:21:36 - 12:35:48	1068	TRUE	904	0.88%	86.73%	12.39%	11.50%	7.97%	-	80.53%	-	7.52%	13.05%	79.43%
			FALSE	164	0.61%	32.32%	67.07%	2.44%	2.44%	-	95.12%	-	-	7.32%	92.68%

There is a huge percentage or 79.43% of true incidents identified under the acceptance quadrant and this figure can be considered as misclassification, as in the ideal situation a true incident should be classified under the first or second quadrant. However, this percentage clearly shows that those true incidents are not really critical; therefore it is acceptable to be considered under that quadrant. The results are also consistent with the DARPA dataset where most of the true incidents were identified as failed incidents, especially in the last main phase.

Furthermore, the distribution of true incidents using RSM is better compared to Snort Priority and CVSS v2. To look at them closer, in the 3rd phase, the true incidents were prioritised into three different groups using RSM compared to only one group with Snort Priority. In this case, the distribution allows any automated response systems to initiate multiple actions on incidents. This means incidents will have different types of response depending on their criticality and priority.

5 Conclusion

The results presented in the previous section are encouraging, as the Risk Index Model (RIM) works well with the Response Strategy Model (RSM) by mapping all

incidents into their appropriate quadrants. The model has also shown a significant result in mapping between the quantitative indexes with the qualitative group of priorities. In addition, the results presented have shown a significant relationship between incident priorities and their classification. The case studies in this stage have shown a significant relationship in addressing false and true incidents.

References

1. Cohen, F.: Simulating cyber attacks, defences, and consequences. *Computers & Security* 18(6), 479–518 (1999)
2. Mu, C., Li, Y.: An intrusion response decision-making model based on hierarchical task network planning. *Expert Systems with Applications* 37(3), 2465–2472 (2010)
3. Carver, C.A.: Adaptive Agent-Based Intrusion Response. PhD Dissertation. Texas A&M University (2001)
4. Papadaki, M., Furnell, S.M.: Informing the decision process in an automated intrusion response system. *Information Security Technical Report* 10(3), 150–161 (2005)
5. Stakhanova, N., Basu, S., Wong, J.: A Cost-Sensitive Model for Preemptive Intrusion Response Systems. In: 21st International Conference on Advanced Information Networking and Applications (AINA 2007), Niagara Falls, Canada, pp. 428–435 (2007)
6. Foo, B., Wu, Y.S., Mao, Y.C., Bagchi, S., Spafford, E.: ADEPTS: adaptive intrusion response using attack graphs in an e-commerce environment. In: International Conference on Dependable Systems and Networks (DSN 2005), Yokohama, Japan, pp. 508–517 (2005)
7. Anuar, N.B., Furnell, S., Papadaki, M., Clarke, N.: A Risk Index Model for Security Incident Prioritisation. In: 9th Australian Information Security Management Conference, Perth, Australia, pp. 25–39 (2011)
8. Covey, S.R.: *7 Habits of Highly Effective People*, 15th Anniversary edn. Simon & Schuster Ltd. (2004)
9. Hillson, D.: Developing Effective Risk Responses. In: 30th Annual Project Management Institute 1999 Seminars & Symposium, Philadelphia, Pennsylvania, USA (1999)
10. Anuar, N.B., Papadaki, M., Furnell, S., Clarke, N.: An investigation and survey of response options for Intrusion Response Systems (IRSs). In: Information Security for South Africa (ISSA), Johannesburg, South Africa, pp. 1–8 (2010)
11. Mell, P., Scarfone, K., Romanosky, S.: Common Vulnerability Scoring System. *IEEE Security & Privacy* 4(6), 85–89 (2006)
12. Snort: The open source network intrusion detection system, <http://www.snort.org>
13. DARPA Intrusion Detection Data Sets, <http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/index.html>

Intrusion Tolerance of Stealth DoS Attacks to Web Services

Massimo Ficco and Massimiliano Rak

Department of Information Engineering, Second University of Naples (SUN)
{massimo.ficco,massimiliano.rak}@unina2.it

Abstract. This paper focuses on one of the most harmful categories of Denial of Service attacks, commonly known in the literature as “stealth” attacks. They are performed avoiding to send significant volumes of data, by injecting into the network a low-rate flow of packets in order to evade rate-controlling detection mechanisms. This work presents an intrusion tolerance solution, which aims at providing minimal level of services, even when the system has been partially compromised by such attacks. It describes all protection phases, from monitoring to diagnosis and recovery. Preliminary experimental results show that the proposed approach results in a better performance of Intrusion Prevention Systems, in terms of reducing service unavailability during stealth attacks.

Keywords: stealth attacks, intrusion tolerance, web services.

1 Introduction

Denial of Service (DoS) attacks are serious threats to the Internet causing billions of dollars in economic loss. In particular, brute force and flooding attacks against application-layer services, like the Web Services (WSs), pose a huge risk to several business-critical services. The recent tide of DoS attacks against high-profile WSs, including PayPal, MasterCard and Amazon, demonstrate how devastating DoS attacks are. In general, attackers are aware of the presence of protection mechanisms: they thus attempt to perform their activities in a stealthy fashion in order to elude local security mechanisms. From an attacker point of view, one of the most effective way of circumventing these security countermeasures consists in distributing the attacks both in ‘form’ and/or ‘time’ executing. Thus, a particular categories of DoS attacks are low-rate DoS attacks, commonly known in the literature as “stealth” attacks [1]. Unlike high-rate attacks, they minimize the visibility of the attack, at the same time they can be as harmful as common brute force attacks [2]. The attack is carried out by directing a flow of packets to a particular system at such a low-rate that would evade DoS detection mechanisms protecting it. Finer detection and prevention mechanisms have to be defined to to reduce the intrusion on the target system [3].

In this paper, we focus on low-rate DoS attacks to WSs that exploit application-level vulnerabilities. In general, network defenses, including firewalls and network Intrusion Detection/Prevention Systems (IDS/IPS) are useless against such attacks to software systems, since they do not analyzes packet contents. Such

attacks are based on application-level exploits, which in most cases are indistinguishable from normal use. We propose an intrusion tolerant approach that aims at mitigating application-level low-rate DoS attacks from generating a service unavailability. The approach consists of monitoring anomalous resources consumption of the target machine and correlating this information with some attack symptom. If an attack is detected, *i.e.*, both the system resources consumption exceeds a critical level and some malicious requests to the WS are observed, an “adaptive” filtering is applied. It consists of filtering application requests on the base of a threshold that is dynamically adapted during the attack. Our experiments consist to inject low-rate of malicious requests that exhaust the computational resources of the host system. In particular, we focus on an example of stealth DoS attacks that exploit XML vulnerabilities [4]. Our preliminary results shown that the proposed approach results in a better performance of the IPSs that adopt “static” threshold-based prevention mechanisms, in terms of reducing the service unavailability during a stealth attack.

2 Building Stealth Attacks

Denial-of-Service attacks aim at reducing services availability by exhausting the resources of the services host system, like memory, processing resources and network bandwidth. Some classic way to perform such attacks to WSs, consists of: *(i)* querying a service using a very large request message, in order to exploits the high memory consumption of the request processing; *(ii)* forcing to decrypt a large number of encrypted messages in order to lead to high CPU load; *(iii)* creating a new process instance each time a message arrives; and *(iv)* formulating well-formed messages that require complex processing on the target system [5]. Stealth attacks have been defined in [7] as those aiming at keeping the attackers virtually invisible to network-based defenses. These attacks can be significantly harder to detect compared with more traditional brute-force and flooding style attacks [2]. It is an interesting open issue to detect and react to such attacks that exhibit “variable” and “polymorphic” behaviors [12]. It is hard to specify quantitative time constraints in stealth attacks. Attackers can perform polymorphic attacks that change their time constraints.

Therefore, since such kind of dynamic changes are very hard to predict, intrusion tolerant mechanisms that use “adaptive”, instead of “static”, thresholds could overcome such limitations (*i.e.*, thresholds that do not depend from an initial training phase). Obviously, the kind of detection and reaction mechanisms to be used cannot be absolute, but must be defined with respect to an intrusion model, that is, the specification of effects that the successful attack has on the target system (*e.g.*, the degree of success of the intruder in terms of resource consumption). Therefore, in order to define an adaptive mechanism to detect and react to a low-rate attack, it is necessary to identify good attack examples and analyze/model their effects of the target system. We considered flooding attack that exploits the XML vulnerability, which is one of main factor that makes WSs vulnerable to DoS attacks. In particular, we consider the *Coercive Parsing* attack, also called *Deeply-Nested XML DoS (XDoS)*. It is a resource exhaustion attack, which exploits the XML message format by inserting of a large number of nested XML tags in the message body. The goal is to force the XML parser

within the server application, to exhaust the computational resources of the host system by processing numerous deeply-nested tags. In a previous work [8], we analyze the CPU consumption depending on the number of nested XML tags and the frequency with which the malicious messages are injected. The experiment shows that a message of 500 nested tags is sufficient to produce a peak of CPU load of about 97%, whereas with 1000 tags the CPU is fully committed to process the malicious message for about 3 seconds. A flooding attack example that exhausts fully the system's CPU resource, consists to inject malicious sequences of XML messages with 150 nested tags every 200 ms.

In this work, in order to identify good candidate stealth attacks, we analyze the CPU consumption in presence of sustained XDoS attack. We perform different attack scenarios. Each scenario takes about 2 hours and consists of a sequence of messages injected with a fixed frequency and a fixed number of nested tags. The experiments show that it is sufficient to inject messages with about 5 nested XML tags every 5 ms, to make unavailable the target machine (*i.e.*, to exhaust the CPU resource). On the base of such analysis, we define a distributed low-rate XDoS attack. It consists in distributing the attacks both in 'form' and 'time' executing, *i.e.*, injecting messages with a different number of nested XML tags, in order to circumvent fixed thresholds (attack distribution in form), and delays single messages so as to bypass time window and rate controls (attack distribution in time). Such an attack flow keeps the computational capacity of target host constantly busy, hence affecting the WS availability. The sequence of injected messages is generated through an algorithm that randomly varies the frequency and the number of nested tags in order to ensure a constant CPU overloading.

3 The Intrusion Tolerant Approach

In order to detect low-rate DoS attacks, which exhaust the computational resources of the target machine for extended periods, an active resource monitoring approach is adopted. It consists in observing anomalous resource usage load (monitoring), and analyzing the possible causes of such resource usage overloading, in order to decide if such anomalous behavior is due to either an attack or a normal operation (diagnosis). When a stealth DoS attack is detected, a threshold-based filtering reaction is triggered on the base of the target resource usage load (recovery).

The CPU monitoring mechanism has to deal with application scenarios highly variable, which alternate periods of heavy workloads, which involve high number of software components and heterogeneous type of service requests, with periods of low computational activities. We assume that during 'normal' operation, the monitored CPU behavior can be modeled as a random walk, which alternates stable periods, during which the CPU load has some 'stable' behavior (*i.e.*, it is within a specific range), with transient periods (smaller compared with the stable), during which significant variation of CPU consumption occurs. During the transient period, changes in the monitored behavior consists in continuous increments or decrements with respect to the 'stable' behavior due to a workload variation (*e.g.*, due to a burst of requests). Thus, a count-and-threshold over-time monitoring mechanism using heuristic approaches is adopted to detect anomalous CPU consumption. It consists to monitor extended excessive

CPU consumption and detect when a threshold is reached. The diagnosis activity consists to verify the presence of some stealth attack symptom in the application requests. It combines several information collected by using different anomaly-based detection models that estimate the anomaly degree of monitored features, including: (i) the type of sequence of XML nested tags included in the message, (ii) the actual distribution of nested XML tags in the message sequence, and (iii) the number of nested XML tags of each message. Monitored symptoms are correlated (by means of a simple weighted sum) and rearranged based on a confidence level, which indicates the likelihood that they are symptomatic of an ongoing attack on the system. Finally, if the confidence exceeds a fixed threshold, a recovery action is performed [9,10]. The objective of the reaction is to reduce the CPU load on the target system, in order to reduce the period in which the service is unavailable. In particular, it filters each XML message that contains a number of nested tags greater than a given threshold T_A . In order to face stealth attacks, we have to adapt the threshold, so that it cuts even low-rate attack messages (*i.e.*, messages with a low number of nested tags). The ‘adaptive’ threshold T_A is decreased until the CPU consumption falls below a severity level. The drawback of such an approach is the presence of false positive results (*i.e.*, the filter cuts valid messages). We adopt the following exponential function to decrease the filtering threshold T_A , which is equal to $B\lambda e^{-\lambda k}$ when $k > 0$, and equal to 0 when $K < 0$. k is a discrete time variable, and B is the maximum number of nesting tags that is initially admitted.

4 Experiments and Results

In this section, we present preliminary results achieved by applying the proposed intrusion tolerant approach in our experimental testbed. The experiment consists of the following steps: (1) launch attacks from attacker machine, while maintaining a simulated stress load on the Web server; (2) trigger an alert only when an intrusion is detected (*i.e.*, whether both an excessive CPU consumption is observed and the attack symptoms are diagnosed); (3) perform the intrusion reaction. In order to assess the effectiveness of the proposed solution, TPC Benchmark W (TPC-W) is adopted [11]. It is a transactional Web benchmark. The benchmark simulates the activities of a business oriented transactional Web server. The performance metric reported by TPC-W is the number of Web interactions processed per second (WIPS), which is used to evaluate the reaction mechanism. During the experimental campaign, in order to evaluate the proposed solution, a workload based on real traffic is adopted. It is collected from production Web server at the university, in which the considered application runs, during an interval time of 12 hours. It consists of about 768.154 messages containing not more than 17 nested XML tags. Using results described in Section 2, during the detection phase, we performed several experiments, in which we injected low-rate of malicious SOAP messages. It is characterized by 215.531 messages with a number of nested XML tags including within the range [5-4500]. Finally, we overlapped the stealth XDoS attack traffic on top of the collected workload and TPC-W background traffic.

In Figure 1 compares the results archived by using the proposed *adaptive-threshold* based recovery approach (ATB), with an intrusion prevention approach

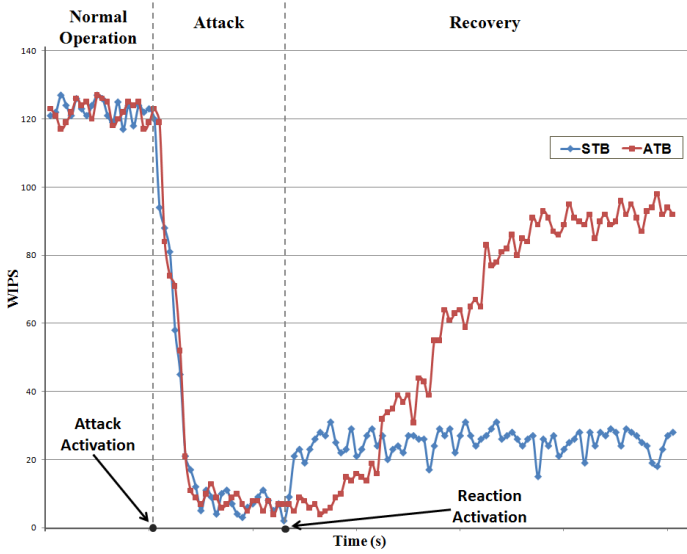


Fig. 1. WIPS evaluation during a recovery process to a stealth XDoS attack

that adopts a *static-threshold* (STB). The STB approach exploits a worst-case threshold T_S , which is estimated during a training phase. In particular, the threshold T_S is fixed to the greater number of nested tags contained in the used workload. Thus, all the received messages that contains more than $T_S = 17$ nested tags, are preventively filtered. Figure 1 represents the WIPS variations with respect to the time, during the three temporal windows. During the first two windows the recovery mechanism is disabled. In particular, the first window shows the values of the WIPS in absence of the attack. The second one shows the intrusion effects. As Figure 1 shows, during the attack, the number of TPC-W interactions processed is very low, about 7% respect to the normal operation. Finally, during the last window, the recovery mechanism is enabled. Experimental results confirm the limitations of the STB approach with polymorphic behavior of stealth attacks. Although, the reaction latency of STB approach is shorter than ATB approach (*i.e.*, the system filters immediately all messages with more than T_S nested tags), the number of TPC-W interactions processed increases by only about 12% respect to the system under the attack. Such approach allows to filter only a part of malicious messages. Moreover, the attacker could use this information to reduce progressively the number of nested tags in order to fully evade the fixed threshold. As Figure 1 shows, by using the ATB approach, the number of TPC-W interactions processed increases by about 69% respect to the system under the attack. On the other hand, the excessive reduction of the threshold can produce ‘false positives’, *i.e.*, correct messages are filtered. Results show that, in order to reduce the CPU load under the 85%, about 0,009% of correct messages are filtered.

5 Conclusion and Future Work

The proposed approach emphasizes the relation among intrusion monitoring, diagnosis and recovery. It consists to monitor anomalous behaviors of resources consumption on the target system, and then, verify/diagnosis if they are caused by an low-rate attack. The experiment results show that the proposed approach is able to cope with the polymorphism of such attacks as well as improve the availability/performance of the WS during the intrusion. The objectives of our future work will be define a more robust model to correlate the symptoms of the attack with its intrusion effects, and extend the proposed approach to a larger set of stealth DoS attacks.

Acknowledgment. This research is partially supported by FP7-ICT-2009-5-256910 (mOSAIC) project and the MIUR-PRIN 2008 project “Cloud@Home”.

References

1. Kuzmanovic, A.: Low-rate tcp-targeted denial of service attacks and counter strategies. *IEEE/ACM Trans. on Networking* 14(4), 683–696 (2006)
2. Zhang, Y., Mao, Z.M., Wang, J.: Low-Rate TCP-Targeted DoS Attack Disrupts Internet Routing. In: *Proc. of the 14th Network and Distributed System Security Symposium, NDSS 2007* (February 2007)
3. Boggs, N., Hiremagalore, S., Stavrou, A., Stolfo, S.J.: Experimental Results of Cross-Site Exchange of Web Content Anomaly Detector Alerts. In: *Proc. of the IEEE Int. Conf. on Technologies for Homeland Security*, pp. 8–14 (November 2010)
4. Jensen, M., Gruschka, N., Herkenh, R.: A survey of attacks on web services. *Computer Science* 24(4), 185–197 (2009)
5. Jensen, M., Gruschka, N., Herkenh, R., Luttenberger, N.: SOA and Web Services: New Technologies, New Standards - New Attacks. In: *Proc. of the Fifth European Conference on Web Services*, pp. 35–44. *IEEE CS* (2007)
6. Kuzmanovic, A., Knightly, E.W.: Low-Rate TCP Targeted Denial of Service Attacks: the shrew vs. the mice and elephants. In: *Proc. of the International Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*. *ACM Press* (2003)
7. Antonatos, S., Locasto, M., Sidiroglou, S., Keromytis, A.D., Markatos, E.: Defending against next generation through network/endpoint collaboration and interaction. In: *Proc. of the 3rd International Conference on Computer Network Defense*. *LNCS*, vol. 30, pp. 131–141. *Springer US* (2008)
8. Ficco, M., Rak, M.: Intrusion Tolerant Approach for Denial of Service Attacks to Web Services. In: *Proc. of the 1st International Conference on Data Compression, Communications and Processing (CCP 2011)*. *IEEE CS Press* (June 2011)
9. Ficco, M.: Achieving Security by Intrusion-Tolerance Based on Event Correlation. *Journal of Network Protocols and Algorithms (NPA)* 2(3), 70–84 (2010)
10. Ficco, M., Romano, L.: A Correlation Approach to Intrusion Detection. In: Chatzimisios, P., Verikoukis, C., Santamaría, I., Laddomada, M., Hoffmann, O. (eds.) *MOBILIGHT 2010*. *LNICST*, vol. 45, pp. 203–215. *Springer, Heidelberg* (2010)
11. TPC Benchmark W (TPC-W), <http://www.tpc.org/tpcw/>
12. Li, Z., Wang, L., Chen, Y., Fu, Z.: Network-based and Attack-resilient Length Signature Generation for Zero-day Polymorphic Worms. In: *Proc. of the IEEE Int. Conf. on Network Protocol*, pp. 164–173. *IEEE CS Press* (October 2007)

Towards Use-Based Usage Control

Christos Grompanopoulos and Ioannis Mavridis

Department of Applied Informatics, University of Macedonia,
156 Egnatia Street, 54006, Thessaloniki, Greece
{groban,mavridis}@uom.gr

Abstract. In this paper, a new Use-based usage CONtrol (UseCON) approach that supports recording of usages with the help of a new entity, named use, is presented. Uses provide information for the latest state (requested, active, denied, completed or terminated) of every usage and facilitate the fine-grained definition and proper association of attributes to various system entities. The proposed approach provides enhanced contextual information modeling, support of complicated access modes and an alternative approach in obligations modeling. Moreover, UseCON is characterized by high expressiveness and ability to define policy rules in almost natural language.

Keywords: access control, usage control, UCON.

1 Introduction

Despite Usage CONtrol's (UCON) [3] virtues, supporting complex usage scenarios of modern computing environments is a challenging procedure [4,1]. Such a complexity usually results in involving a large number of entities and in utilizing multi party contextual information during the decision making process of a particular usage. Usage control for modern computing environments should support novel access modes on resources, along with new socio-technical abstractions and relations, which are created during the usage process [4]. Enhanced expressiveness is an additional requirement that emerges as of vital importance for next generation usage control models. Furthermore, requirements as the ability to define administrative policies in an ease and efficient way (e.g. expressing policies as closer to the natural language) should be fulfilled.

2 Main Definitions

The UCON family of models [3] is mainly characterized by fine grained control of resources, support for continuity of decisions, and attribute mutability. However, UCON reveals a number of challenging issues when it comes to support modern computing environments. In the rest of this section, the proposed Use based usage CONtrol (UseCON) approach is defined, as an enhancement of UCON.

2.1 Elements

UseCON approach consists of four components viz. subject, object, action and use. Each component is associated with attributes. Moreover, only authorizations are utilized as a usage decision factor.

Components. Subjects are entities that request to exercise operations on objects. A subject can be a human, a device or a software agent acting on behalf of a human. Objects can be physical entities, logical entities or services (e.g. a printer, a file or a database migration service). An entity operating as subject in one usage may be the object of another usage [6].

Actions are operations that subjects exercise on objects. The types of subjects or objects determine the types of the actions that can be exercised. For example, in case of a file, possible actions could be read, write and execute. In a similar manner, different types of subjects (e.g. humans, or agent programs) can exercise different types of actions on the same object (e.g. reload paper into a printer or cancel a job in a printing queue, respectively). Moreover, subjects can exercise through actions direct operations on objects, or delegations of actions to other subjects, or administrative operations on various system elements (in a same way with rights in [3]). In this paper, we further elaborate only on actions that exercise directly operations on objects.

A core component of the UseCON approach is *use*. A use materializes all the characteristics of a usage that are critical for the decision making process. The information contained in a use is not predetermined but is composed at the time a usage is requested. A use actually records the relation between the subject, the object and the action of a particular usage. We define the set *components* (C) containing all the components ($c_i, i = 1, 2, \dots, n$), of a system, in the form of subjects (S), objects (O), actions (A) and uses (U).

Attributes. Subjects and objects are associated with security-relevant properties or capabilities, called attributes. In addition, contextual information, which in UCON is stored in condition variables [3], is now proposed to be associated with subject or object attributes. The enhancement of subject or object attributes with context results to the fact that the update of the value of some attributes (those related with contextual information) is performed not through an administrative process, but automatically due to a system's context modification (e.g. time). Additionally, the frequency of attribute mutation has a great range, varying from very rare-changing attributes, e.g. the identity, to always-changing attributes, e.g. time [1]. In order to support complicated operations of modern computing systems, it is required for actions to be associated also with attributes. An example of an action attribute in a file-related operation as write, could be an encryption key. Uses are also associated with attributes that include information related to any combination of a subject, object and action (e.g. the price of a service).

The set of component's attributes (CA) contains the attributes ($ca_i, i = 1, 2, \dots, m$) of all components. A relation $ATT(c_i)$ denotes the association of a

component $c_i \in C$ with a tuple of attributes. We adopt the function notation in order to represent the value (range) that is assigned to an attribute (function) of a specific component (domain). For example, in expression $Age(Alice) = 34$, 'Alice' is a component (subject) that has associated to attribute Age with assigned value '34'. Every subject, object and action is associated with at least the id attribute and through this it is assigned a unique value. The value of an id attribute remains constant during the life of the usage control system [5]. When a use instance is created, a tuple of special attributes, namely sid , oid and aid , is initially associated with it. These attributes have the same values to the id attributes of the subject, the object and the action, respectively, of the usage that the specific use describes. Moreover, an additional $time$ attribute is associated with each one use [1]. The tuple $(sid, oid, aid, time)$ is unique for each one use (the usage of a subject on an object with an action at a specific time is also unique) and consequently acts as the identifier of the use.

Each use is further associated with a $state$ attribute, which embodies the accomplished status of the usage in progress, as it is described in [5] and augmented in [2]. The $state$ attribute represents the possible states of a use, as depicted in Fig. 1 and each time it receives one of the following values:

- *Requested*: On a request for a usage, appropriate attributes are associated to a use and proper values are assigned to them. The pre-authorization rules, which govern the requested usage, have not been evaluated yet.
- *Activated*: After a requested usage is allowed, as a result of successfully fulfilled pre-authorization rules, and is now currently being executed.
- *Denied*: After a requested usage has been denied, because it failed to satisfy the pre-authorization rules.
- *Stopped*: After an allowed / ongoing usage has been terminated by the system due to a violation of an ongoing authorization rule.
- *Completed*: A usage that has completed due to a subject's intervention.

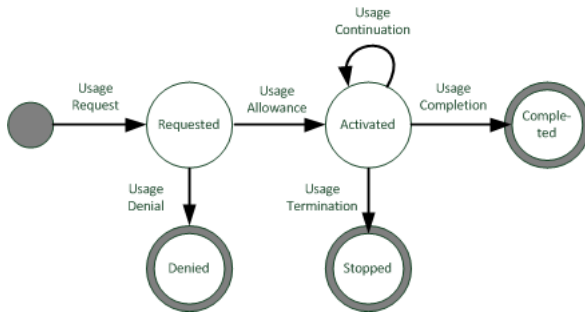


Fig. 1. Use state-transition diagram

¹ The specific value of time attribute could vary from the time of usage request to the time of usage termination/completion and is left open as implementation choice.

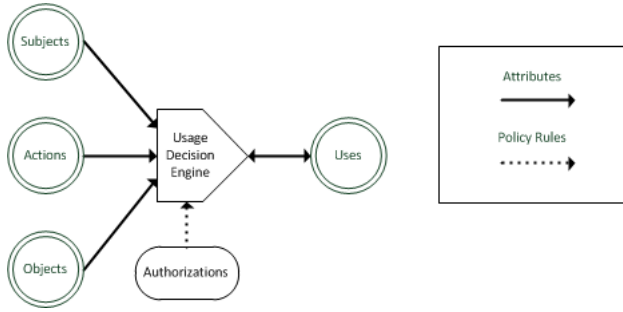


Fig. 2. UseCON elements and relations

Authorizations. Authorizations are functional predicates that utilize component attributes. However, a predicate could be fulfilled by the values of attributes either of the components that are involved in the usage in request, or of the components of other usages in the system. The aforementioned modification makes UseCON authorizations much more flexible than UCON ones, in which the allowance of a usage requested by a subject upon an object is based only on their attributes. For example, an authorization in UCON can model a policy rule requiring that a student can present the work of his team if and only if he has previously been registered in the system. However, UseCON's authorizations can support more complicated rules. Hardening the previous example, it may be required that the presentation of team work by the student is allowed if any member of his team has been registered.

Associating contextual information with component attributes results in the replacement of UCON's conditions with authorizations. Moreover, operations required by UCON's obligations are handled as usual usages in UseCON. Therefore, the exercise of obligation operations in UseCON is verified by searching the uses of the system, supported by authorizations. Based on the aforementioned suggestions, the UseCON elements and their relations are depicted in Fig. 2.

2.2 Decision Factors

UseCON utilizes only authorizations as decision factors for the allowance of a usage request. Applying continuity of decision in authorizations results in two UseCON approaches, namely on pre-authorizations and ongoing-authorizations.

Pre-Authorizations. A subject is permitted to exercise an action on an object only if a number of predicates $preA_i, i = 1, 2, \dots, n$, connected with logical operators are satisfied. There is no further (ongoing) control after the usage's allowance, which is terminated after a subject's request. More specifically, a UseCON pre-authorization rule is defined as:

$$allowed(s, o, a) \Rightarrow preA_1(ATT(u)) \oplus preA_2(ATT(u)) \oplus \dots \oplus preA_n(ATT(u))$$

where the symbol \oplus is replaced by a logical AND/OR. Each one $preA_i(ATT(u))$, $i = 1, 2, \dots, l$, is a predicate that utilizes either the attributes from the use that materializes the requested usage or the attributes from other (previous or concurrent) uses in the system. Moreover, each use is associated with a tuple of attributes ($sid, oid, aid, time$) that contains the values of the id attributes of the components participating in the usage. Consequently, with the application of the reverse id^{-1} function (e.g. $id^{-1}(sid(u)) = s$) on the id values of the components, a $preA_i$ predicate is able to utilize the attributes of the components participating in the usage. The semantics of $preA_i$ are defined as follows:

$$preA_i(ATT(u)) \Rightarrow \left\{ \begin{array}{l} sp_i(ATT(u)) \\ | \{u' \in U \wedge cp_i(ATT(u), ATT(u'))\} | \otimes k_i \end{array} \right.$$

where $k_i \in \mathbb{N}$, the symbol \otimes is replaced by \geq or \leq , accordingly, and $|B|$ denotes the number of elements of set B. The term sp_i denotes a simple authorization predicates that is evaluated on the attribute values of the requested use. Alternatively, cp_i is a complex authorization predicate that is evaluated on the attribute values of the requested and the rest uses of the system.

Ongoing-authorizations. An ongoing-authorization utilizes the same elements as a pre-authorization. However, an ongoing-authorization has not to be fulfilled before the usage request, but during the exercise of the usage. A UseCON ongoing-authorization rule is defined as:

$$allowed(s, o, a) \Rightarrow true$$

$$stopped(s, o, a) \Leftarrow \neg(onA_1(ATT(u)) \oplus onA_2(ATT(u)) \oplus \dots \oplus onA_n(ATT(u)))$$

The semantics of an onA_i functional predicate are the same with the $preA_i$ ones defined in pre-authorizations and \oplus is replaced by a logical AND/OR.

3 Example

The advantages of the proposed approach are highlighted in this section, via an example, which is a combination of UCON's examples 25 and 26 as presented in [3]. More specifically, a policy rule in a hospital, requires that a doctor can operate a patient only if the patient has given his consent to be operated. In addition, the doctor must have proven experience with at least three operations in the past. The corresponding UseCON modeling is:

AREA : $S, O \rightarrow 2^{Speciality}$ Speciality is a set of medical speciality names
 SROLE : $S \rightarrow 2^{Role}$ Role is an unordered set of roles

$$\begin{aligned} Allowed(s, o, a) \Rightarrow & | \{u' \in U \wedge status(u') = \text{“completed”} \wedge sid(u') = id(s) \wedge \\ & \text{“doctor”} \in srole(s) \wedge area(s) \cap area(o) \neq \emptyset \wedge aid(u') = \text{“operate”} \} | \geq 3 \wedge \\ & | \{u'' \in U \wedge status(u'') = \text{“completed”} \wedge sid(u'') = id(o) \wedge \\ & aid(u'') = \text{“consent”} \} | \geq 1 \end{aligned}$$

The above modeling is composed of two search operations for previous uses of the system. The first search operation returns the number of completed operations performed by the doctor, which must be at least three. The second search operation examines if the patient has exercised the *consent* action. The decision making is based on the conjunction of the logical results of the two search operations.

4 Conclusion

In this paper, the definition of the use entity and a new use-based usage control approach was proposed. A use materializes all the characteristics of a usage that are critical for the decision making process. Consequently, a policy administrator can obtain through uses a detailed view of usages in the system. In addition, UseCON supports an enhanced handling of contextual information together with the support of complicated access modes of subjects on objects and an alternative approach to the utilization of obligations and conditions in UCON. The augmented expressiveness of UseCON was demonstrated in an example composed of two particular ones that have been presented in [3]. Another virtue of the model, as illustrated in the same example, is the similarity between the UseCON modeling and the expression of policy rules in almost natural language.

References

1. Grompanopoulos, C., Mavridis, I.: Towards differentiated utilization of attribute mutability for access control in ubiquitous computing. In: Panhellenic Conference on Informatics, pp. 118–123 (2010)
2. Katt, B., Zhang, X., Breu, R., Hafner, M., Seifert, J.P.: A general obligation model and continuity: enhanced policy enforcement engine for usage control. In: Proceedings of the 13th ACM Symposium on Access Control Models and Technologies, SACMAT 2008, pp. 123–132. ACM, New York (2008)
3. Park, J., Sandhu, R.: The ucon abc usage control model. *ACM Transactions on Information and System Security* 7, 128–174 (2004)
4. Thomas, R.K., Sandhu, R.: Models, protocols, and architectures for secure pervasive computing: Challenges and research directions. In: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, PERCOMW 2004, pp. 164–170. IEEE Computer Society, Washington, DC (2004)
5. Zhang, X., Parisi-Presicce, F., Sandhu, R., Park, J.: Formal model and policy specification of usage control. *ACM Transactions on Information and System Security* 8(4), 351–387 (2005)
6. Zhang, X., Sandhu, R., Parisi-Presicce, F.: Safety analysis of usage control authorization models. In: Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security, ASIACCS 2006, pp. 243–254. ACM, New York (2006)

Author Index

- Adi, Kamel 187
Anuar, Nor Badrul 573
Armando, Alessandro 13
Athanasopoulos, Elias 103
Atluri, Vijayalakshmi 150
Augusto, Alexandre B. 61
- Bai, Kun 388
Bal, Gökhan 327
Banse, Christian 235
Barbu, Guillaume 37
Bella, Giampaolo 273
Beres, Yolanta 424
Berger, Bernhard J. 25
Boulares, Sofiene 187
Boyar, Joan 287
- Cai, Shaoying 501
Caldeira, Mário 352
Chen, Ping 138
Christofi, Maria 299
Clarke, Nathan L. 465, 573
Coles-Kemp, Lizzie 273
Coquery, Emmanuel 525
Coronges, Kathryn 457
Correia, Manuel Eduardo 61
Cui, Xiang 261
Cuppens, Frédéric 174
Cuppens-Boulahia, Nora 174
- D'Acquisto, Giuseppe 412
De Capitani di Vimercati, Sabrina 199
De Decker, Bart 339, 555
Dhillon, Gurpreet 352
Ding, Baozeng 115
Dodge, Ronald 457
Drosatos, George 223
- Efraimidis, Pavlos S. 223, 543
- Federrath, Hannes 235
Ficco, Massimo 579
Flamini, Marta 412
Foresti, Sara 199
Furnell, Steven M. 465, 573
- Gadaleta, Francesco 126
Gadyatskaya, Olga 364
Garcia-Alfaro, Joaquin 174
Gessiou, Eleni 103
Ghorbani, Ali 87
Giraud, Christophe 37
Gouget, Aline 299
Griffin, Jonathan 424
Gritzalis, Dimitris 249, 443
Grompanopoulos, Christos 585
Guerin, Vincent 37
- Hacid, Mohand-Saïd 525
Hao, Zhiyu 261
He, Yeping 115
Herrmann, Dominik 235
Herzberg, Amir 315
- In, Hoh Peter 376
Ioannidis, Sotiris 103
- Jajodia, Sushil 199
Jiao, Han 400
John, John C. 150
Joosen, Wouter 126
Jung, Sung-Oh David 376
- Kandias, Miltiadis 561
Kang, Xin 537
Katos, Vasilios 49, 543
Keromytis, Angelos D. 103
Khambhammettu, Hemanth 187
Kolkowska, Ella 339
- Langer, Josef 1
Lee, Heejo 376
Leicher, Andreas 75
Li, Jiuyong 400
Li, Min 388
Li, Shuhao 261
Li, Weihan 211
Li, Yingjiu 501
Liu, Chengfei 400
Liu, Jixue 400
Logrippo, Luigi 187

- Lopez, Javier 162
 Lu, Wei 87

 Mao, Bing 138
 Margulies, Ronen 315
 Massacci, Fabio 364
 Mavridis, Ioannis 585
 Meletiadis, Vasilis 249
 Menesidou, Sofia Anna 49
 Merlo, Alessio 13
 Migliardi, Mauro 13
 Milutinovic, Milica 555
 Mitrou, Lilian 249
 Moataz, Tarik 174
 Mühlberg, Jan Tobias 126
 Mylonas, Alexios 249, 443

 Naessens, Vincent 555
 Nait-Bahloul, Sarah 525
 Naldi, Maurizio 412
 Nikiforakis, Nick 126
 Ntantogian, Christoforos 475
 Ntouskas, Theodoros 567

 Oliveira, Tiago 352
 Onieva, Jose A. 162

 Panda, Brajendra 211
 Papadaki, Maria 573
 Pappas, Vasilis 103
 Paraboschi, Stefano 199
 Park, Hyundo 376
 Peralta, René 287
 Philippov, Anton 364
 Polemi, Nineta 567
 Prandini, Marco 549
 Psaroudakis, Ioannis 543

 Rak, Massimiliano 579
 Ramilli, Marco 549
 Rimasson, Xavier 174
 Rios, Ruben 162
 Roland, Michael 1
 Rovira, Ericka 457

 Saad, Sherif 87
 Saevanee, Hataichanok 465
 Samarati, Pierangela 199
 Sayed, Bassam 87
 Scharinger, Josef 1
 Schmidt, Andreas U. 75
 Shah, Yogendra 75
 Sohr, Karsten 25
 Soupionis, Yannis 561
 Stavrakakis, Ioannis 475
 Sural, Shamik 150
 Susarapu, Santa 352

 Tasidou, Aimilia 223
 Theoharidou, Marianthi 443
 Thing, Vrizlynn L.L. 513
 Traore, Issa 87
 Tsoumas, Bill 249

 Vaidya, Jaideep S. 150
 Verderame, Luca 13

 Wang, Feifei 138
 Wang, Yipeng 261
 Wang, Yongge 489
 Wu, Yanjun 115
 Wu, Yongdong 531, 537

 Xenakis, Christos 475
 Xie, Li 138

 Yang, Shuzhe 327
 Yao, Fufeng 115
 Yaseen, Qussai 211
 Ying, Hwei-Ming 513
 Yu, Meng 388
 Yun, Xiaochun 261

 Zang, Wanyu 388
 Zhang, Yongzheng 261
 Zhang, Yulong 388
 Zhao, David 87
 Zhao, Yunlei 501
 Zhao, Zhigang 531