Bart De Decker
Ingrid Schaumüller-Bichl (Eds.)

LNCS 6109

Communications and Multimedia Security

11th IFIP TC 6/TC 11 International Conference, CMS 2010
Linz, Austria, May/June 2010
Proceedings

ifip

Lecture Notes in Computer Science 6109

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Bart De Decker Ingrid Schaumüller-Bichl (Eds.)

Communications and Multimedia Security

11th IFIP TC 6/TC 11 International Conference, CMS 2010
Linz, Austria, May 31 – June 2, 2010
Proceedings

Springer

Volume Editors

Bart De Decker
K.U.Leuven, Department of Computer Science - DistriNet
Celestijnenlaan 200A, 3001 Leuven, Belgium
E-mail: bart.dedecker@cs.kuleuven.be

Ingrid Schaumüller-Bichl
Upper Austria University of Applied Sciences
School of Informatics, Communications and Media
Softwarepark 11, 4232 Hagenberg, Austria
E-mail: ingrid.schaumueller-bichl@fh-hagenberg.at

Library of Congress Control Number: 2010926923

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

LNCS Sublibrary: SL 4 – Security and Cryptology

ISSN 0302-9743
ISBN-10 3-642-13240-5 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-13240-7 Springer Berlin Heidelberg New York

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Preface

Over the last decade, we have witnessed a growing dependency on information technology resulting in a wide range of new opportunities. Clearly, it has become almost impossible to imagine life without a personal computer or laptop, or without a cell phone. Social network sites (SNS) are competing with face-to-face encounters and may even oust them. Most SNS-adepts have hundreds of "friends", happily sharing pictures and profiles and endless chitchat. We are on the threshold of the Internet of Things, where every object will have its RFID-tag. This will not only effect companies, who will be able to optimize their production and delivery processes, but also end users, who will be able to enjoy many new applications, ranging from smart shopping, and smart fridges to geo-localized services. In the near future, elderly people will be able to stay longer at home due to clever health monitoring systems. The sky seems to be the limit! However, we have also seen the other side of the coin: viruses, Trojan horses, breaches of privacy, identity theft, and other security threats. Our real and virtual worlds are becoming increasingly vulnerable to attack. In order to encourage security research by both academia and industry and to stimulate the dissemination of results, conferences need to be organized.

With the 11th edition of the joint IFIP TC-6 TC-11 Conference on Communications and Multimedia Security (CMS 2010), the organizers resumed the tradition of previous CMS conferences after a three-year recess. It is with great pleasure that we present the proceedings of CMS 2010, which was held in Linz, Austria on May 31 – June 2, 2010. The conference was organized by the Department of Secure Information Systems, Upper Austria University of Applied Sciences, School of Informatics, Communications, and Media, Hagenberg.

The program committee (PC) received 55 submissions out of which 23 papers were accepted. We would like to thank all the authors who submitted papers. Each paper was anonymously reviewed by three to four reviewers. In addition to the PC members, several external reviewers joined the review process in their particular areas of expertise. We are grateful for their sincere and hard work. We tried to compile a balanced program covering various topics of communications and multimedia security: VoIP, TLS, web services, watermarking, biometrics, risk management, just to name a few.

This year, the conference featured a poster session in which authors could present "work in progress". We are grateful to Taher Elgamal (Axway Inc), Edward Humphreys (XiSEC) and Klaus Gheri (phion AG - a Barracuda Networks company) for accepting our invitation to deliver keynote speeches.

We hope that you will enjoy reading these proceedings and that they may inspire you for future research in communications and multimedia security.

May 2010

Bart De Decker
Ingrid Schaumüller-Bichl

Organization

General Chair

Ingrid Schaumüller-Bichl	Upper Austria University of Applied Sciences, Campus Hagenberg, Austria

Program Chair

Bart De Decker	K.U.Leuven, Belgium

Organizing Chair

Robert Kolmhofer	Upper Austria University of Applied Sciences, Campus Hagenberg, Austria

Program Committee

Helen Armstrong	Curtin Business School, Australia
Wilhelm Burger	University of Applied Sciences, Hagenberg, Austria
Jan Camenisch	IBM Zurich Research Lab, Switzerland
David Chadwick	University of Kent, UK
Howard Chivers	Cranfield University, UK
Gabriela Cretu-Ciocarlie	Columbia University, USA
Frédéric Cuppens	ENST-Bretagne, France
Sabrina De Capitani di Vimercati	Università degli Studi di Milano, Italy
Gerhard Eschelbeck	Webroot Software Inc., USA
Simone Fischer-Hübner	University of Karlstad, Sweden
Jürgen Fuß	University of Applied Sciences, Hagenberg, Austria
Sébastien Gambs	INRIA, France
Christian Geuer-Pollmann	EMIC, Germany
Dieter Gollmann	University of Hamburg, Germany
Rüdiger Grimm	University of Koblenz-Landau, Germany
Eckehard Hermann	University of Applied Sciences, Hagenberg, Austria
Jaap-Henk Hoepman	Radboud University Nijmegen, Netherlands
Russ Housley	Vigil Security, USA
Ted Humphreys	Xisec, UK
Witold Jacak	University of Applied Sciences, Hagenberg, Austria
Lech Janczewski	University of Auckland, New Zealand

Stefan Katzenbeisser	TU Darmstadt, Germany
Markulf Kohlweiss	K.U.Leuven, Belgium
Herbert Leitold	Technical University of Graz, Austria
Javier Lopez	University of Malaga, Spain
Louis Marinos	ENISA, Greece
Chris Mitchell	Royal Holloway, University of London, UK
Refik Molva	Institut Eurécom, France
Yoko Murayama	Iwate Prefectural University, Japan
Jörg R. Mühlbacher	Johannes Kepler University Linz, Austria
Vincent Naessens	Kath. Hogeschool Sint-Lieven, Gent, Belgium
Günther Pernul	University of Regensburg, Germany
Gerald Quirchmayr	University of Vienna, Austria
Jean-Jacques Quisquater	UCL, Belgium
Kai Rannenberg	Goethe University Frankfurt, Germany
Vincent Rijmen	K.U.Leuven, Belgium & TU Graz, Austria
Pierangela Samarati	Università degli Studi di Milano, Italy
Riccardo Scandariato	K.U.Leuven, Belgium
Ingrid Schaumüller-Bichl	University of Applied Sciences, Hagenberg, Austria
Jörg Schwenk	Horst Görtz Institute, Germany
Hermann Sikora	GRZ IT Group, Austria
Leon Strous	De Nederlandsche Bank, Netherlands
Andreas Uhl	University of Salzburg, Austria
Rossouw von Solms	Nelson Mandela Metropolitan University, South-Africa
Tatjana Welzer	University of Maribor, Slovenia

Local Organization

Department of Secure Information Systems, Upper Austria University of Applied Sciences, Campus Hagenberg, Austria

Robert Kolmhofer	Margit Lehner
Johannes Edler	Alexander Leitner
Jürgen Fuß	Anna Perschl
Eckehard Hermann	Dieter Vymazal
Yvonne Horner	Markus Zeilinger

External Reviewers

Isaac Agudo	Matei Ciocarlie
Cristina Alcaraz	Gouenou Coatrieux
Claudio Agostino Ardagna	Nora Cuppens-Boulahia
Goekhan Bal	Leucio-Antonio Cutillo
Stefan Berthold	Stefan Dürbeck
Martin Centner	Kaoutar Elkhiyaoui

Christoph Fritsch
Oliver Gmelch
Hans Hedbom
Marvin Hegen
Stephan Heim
Maarten Jacobs
Tibor Jager
Burt Kaliski
Pablo Najera
Michael Netter
Anna Perschl
Christian Rathgeb
Andreas Reisser
Ruben Rios
Sven Schäge
Koen Simoens
Peter Teufl
Julien Thomas
Dieter Vymazal
Christian Weber
Li Weng
Lars Wolos
Ge Zhang

Table of Contents

Analysis and Detection of Malicious Code and Risk Management

VoIP Security

Biometrics

Applied Cryptography

Secure Communications

Keynotes

Profitable Information Security Policies

Edward Humphreys[1]

When you go to Japan you hear businesses talking about implementing "ni nana zero zero ichi" and the benefits they have gained from their endeavors of implementing information security management systems. The same business excitement can be heard in other Asian, North American, European and Middle-eastern countries. So what is this buzz all about? Quite simply implementing information security policies that enable businesses to do well, to take business opportunities that are profitable. These companies are all talking about implementing ISO/IEC 27001 the international standard on information security management which has become the common language for securing their business operations and engaging in profitable business relationships with their clients, customers and business partners.

So what does it mean to have a profitable information security policy? Clearly a company is in the market to make profits, to have a viable business future, to be the best of its kind in the marketplace and to protect its current investments and invest in new business opportunities. As they say information is an important business commodity which can be bought and sold, and like a currency, helps your business to make its profits, to invest in future business opportunities and to protect itself in the marketplace. Having the right information, at the right time and appropriate use of this information can be driver for a company to become richer and more profitable, to better its market position and for self preservation. A company is in the business of taking risks, to make profits, to be a viable market player and to be 'best of its kind'. Every time it makes an investment, makes a management decision to offer a new range of products, offer new services, to back new opportunities the company is taking a risk. A profitable information security policy should be able to help the company to take the business risks it needs to take without worrying about the protection of its business information - that information it uses to run its business and the same information it uses to formulate its business strategy, make well-informed business decisions and to maximizing its business opportunities.

Therefore the lack of a viable information security policy or an information security policy ill suited to the business can be detrimental to the business, its hopes and vision, and its future business plans. A profitable information security policy should balance out the maximum protection it needs to protect against the risks to its business information, that is to minimize its security risks, and the minimum protection it needs for it to take the business risks it needs to take for the good of the company that is to maximize its investments

[1] Visiting Professor at Hagenberg University of Applied Sciences, Austria and BIT University Beijing, China, and Chair of the ISO/IEC JTC1/SC27 working responsible for the ISO information security management system standards.

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 1–3, 2010.

and opportunities. So a profitable information security policy is a risk-based strategy and instrument that will ensure that profitable benefits will accrue if the policy is implemented by the company. Of course this policy be designed in a way that reflects the company's business vision and objectives.

This keynote talk goes through the past, present and future of ISO/IEC 27001 from its early beginnings as a British Standard to its rapid rise as the most successful, best selling international information security standard of all time to hit the international business community. It is a standard that companies around the global have used and continue to use to design their profitable information security policies.

The Impact of Cryptographic-Hardware-Research on Development of Next Generation Firewalls

Klaus Gheri[2]

The old network firewall paradigm has changed significantly. Today's next generation firewall no longer contents itself with making a traffic flow decision based on information readily available from packet and protocol header information. Additional computationally intense and latency critical traffic flow processing is required to determine the very nature of the application causing the traffic and to inspect the application data payload for data leakage, malware or exploit patterns. Since HTTP and its SSL encapsulated counterpart HTTPs have become the de-facto standard transport protocol for a huge number of applications these protocols need to be closely inspected. Especially for HTTPs the application protocol inspection involves a significant amount of CPU processing power. Furthermore a massive crypto load stemming from high bandwidth VPN tunnels and data compression must often be handled by the same device. With the market moving to full duplex wire speed performance requirements of 20 Gbps or more with all detection technologies switched on economically viable and future proof concepts for task acceleration are needed.

Network and Security Architectures - What Works, What Does Not, and What is Really Missing

Taher Elgamal[3]

Information security has grown from a few tools to protect enterprise networks to becoming an integral part of conducting commerce. This presentation provided the overall network and security architectures as they progressed over the last 15 years. The introduction of Internet connectivity in all our business interactions has required the industry to modify its thinking when it comes to securing the networks. This presentation also provided an overview of what worked, what did not work and what is really missing. Special attention was given to the new

[2] Chief Technical Officer, phion AG - a Barracuda Networks company.
[3] CEO of IdentityMind and CSO of Axway Inc.

waves of online fraud and intellectual property theft attacks. Some very well-known sites have been successfully exploited in the recent months, together with growing online fraud due to the exposure of private and confidential information. The presentation provided insight into why such attacks are possible and why are they successful in today's environment. A call for an improved security model with certain new controls can improve the situation and help organizations mitigate against these attacks. There are many existing technologies and commercial products that can help solve these issues, however, the deployment and implementations may be difficult or introduce unnecessary steps to hinder the use of these products. We discussed issues in authentication, access control, and network and application layer defense as well as various content protection and security ideas. There are still many open issues in solving the overall problem. We also discussed the how the use of the Internet to conduct commerce has opened many security issues. Areas for further research and development were discussed especially when it comes to securing content.

Statement from the Industrial Sector

"After a 10 year long research career as a theoretical physicist in quantum communication I was all set and on track for a life in academia. But I decided against the obvious and instead started up phion with a few colleagues. We grew the company in EMEA, went public and last year merged it with Barracuda Networks. This makes another 10 years of my life. In both these work lives I have been deeply involved in R&D both on the doing and management sides. Context and constraints of R&D were however widely different. Basic research as is done over years with great sophistication within academia is virtually impossible to conduct in a commercial environment. Economic constraints, much shorter timescales and a pushier socioeconomic environment require a very pragmatic approach in commercial product development. Therefore conferences such as CMS 2010 are immensely valuable to all of us in the industry because here we can learn firsthand about profound leading edge research, establish relationships on a personal level, and also get a chance to feed back our views and needs to the research community."

Klaus Gheri, Chief Technical Officer, phion AG - a Barracuda Networks company.

A Scalable Wireless Routing Protocol Secure against Route Truncation Attacks

Amitabh Saxena[1] and Ben Soh[2]

[1] Dept. of Computer Science, University of Mannheim, Germany
[2] Dept. of Computer Science, La Trobe University, Australia

Abstract. Wireless routing protocols allow transmitting nodes to have some knowledge of the topology in order to decide when to forward a packet (via broadcast) and when to drop it. Since a routing protocol forms the backbone of any network, it is a lucrative target for attacks. Routing protocols for wired networks (such as S-BGP) are not scalable in an ad-hoc wireless environment because of two main drawbacks: (1) the need to maintain knowledge about all immediate neighbors (which requires a discovery protocol), and (2) the need to transmit the same update several times, one for each neighbor. Although information about neighbors is readily available in a fairly static and wired network, such information is often not updated or available in an ad-hoc wireless network with mobile devices. Consequently, S-BGP is not suitable for such scenarios. We propose a BGP-type wireless routing protocol for such networks that does not suffer from such drawbacks. The protocol uses a novel authentication primitive called Enhanced Chain Signatures (ECS).

1 Introduction

With the advent of wireless technology, mobile ad-hoc networks (MANET) is an active area of research. The primary challenge in MANETs is equipping each device to continuously maintain the information required to properly route traffic. We investigate the use of BGP-type routing protocols in such networks.

The Border Gateway Protocol (BGP) [1,2] is a *path vector* routing protocol [3], in which routers repeatedly advertise routes to their immediate neighbors. On receiving an update, a router checks if the advertised route is better than an existing one. If so, the router updates its table and advertises the new route to all its other immediate neighbors. BGP, however, has many security vulnerabilities [4,5]. For instance, a rogue router could claim a shorter route to some destination in order to intercept traffic. Therefore, modern implementations use a modified variant called Secure-BGP (S-BGP) [6,7]. In S-BGP, routers authorize each neighbor using route attestations (RAs) and updates are peer-specific. In wireless networks, a node with several receivers in its vicinity must first establish the identity of each receiver, and then broadcast as many updates. Such control plane traffic is a bottleneck when the devices are densely distributed.

In [8], an authentication primitive called Chain Signatures (CS) is proposed. As an application, a secure routing protocol called Stateless Secure BGP

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 4–15, 2010.

(SS-BGP) is also briefly discussed. The attack scenario described in [8] is that of a *route truncation attack* in wired networks. SS-BGP differs from S-BGP in that updates are not peer-specific and can therefore be broadcast. However, it is not clear if SS-BGP has any real advantage in wired networks, since true broadcast channels do not exist. Wireless networks, on the other hand, present a perfect application scenario for SS-BGP because they provide true broadcast channels without the ability to control or determine who receives a broadcast.

We extend the work in [8] by presenting an attack scenario for wireless networks that clearly demonstrates the advantages of SS-BGP. We then describe an improved SS-BGP protocol using an extension of CS called Enhanced Chain Signatures (ECS). Although we focus only on path vector routing using BGP, the main ideas carry over to other routing methods. In the rest of this paper, we consider a highly simplified variant of (S)-BGP that is sufficient for our purpose.

2 Route Truncation Attacks

The discussion of this section is based on the network of Fig. 1.

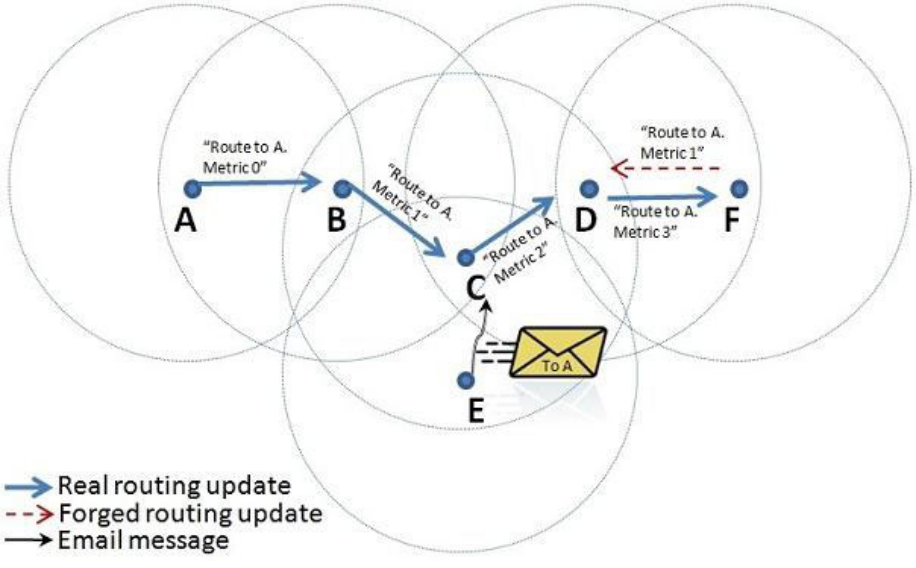


Fig. 1. A typical scenario for a route truncation attack in wireless networks

The circles represent coverages of the nodes located at their centers and the arrows represent broadcasts. Although the arrows point in particular directions, every node within a circle receives that message. Two non-overlapping nodes communicate by using the intermediate nodes as forwarders. $X \rightarrow: m$ indicates that m is broadcast by X. $\mathsf{S}_X(m)$ denotes a signature of X on m (for brevity, assume that signatures provide message recovery). $X \Leftarrow Y$ denotes the string

"*There is a metric 1 path from Y to X*" and $X \Leftarrow$ denotes the string "*There is a metric 0 path to X*". Each node acts as an autonomous system (AS).

BGP updates: (control plane) Refer to Fig. 1. The following updates are sent for routes to A. Each signature (except the first) implies a hop of metric 1.

1. $A \rightarrow: \mathsf{S}_A(A \Leftarrow)$
2. $B \rightarrow: \mathsf{S}_A(A \Leftarrow),\ \mathsf{S}_B(A \Leftarrow B)$
3. $C \rightarrow: \mathsf{S}_A(A \Leftarrow),\ \mathsf{S}_B(A \Leftarrow B),\ \mathsf{S}_C(B \Leftarrow C)$
4. $D \rightarrow: \mathsf{S}_A(A \Leftarrow),\ \mathsf{S}_B(A \Leftarrow B),\ \mathsf{S}_C(B \Leftarrow C),\ \mathsf{S}_D(C \Leftarrow D)$
   $E \rightarrow: \mathsf{S}_A(A \Leftarrow),\ \mathsf{S}_B(A \Leftarrow B),\ \mathsf{S}_C(B \Leftarrow C),\ \mathsf{S}_E(C \Leftarrow E)$
5. $F \rightarrow: \mathsf{S}_A(A \Leftarrow),\ \mathsf{S}_B(A \Leftarrow B),\ \mathsf{S}_C(B \Leftarrow C),\ \mathsf{S}_D(C \Leftarrow D),\ \mathsf{S}_F(D \Leftarrow F)$

From this, each node updates its routing table with a tuple (*destination*, *next-hop*, *metric*) [1]. For instance C's table would contain the entry $(A, B, 2)$.

Forwarding: (data plane) Forwarding uses the following rule: if a data packet arrives from a node that is on the route to the destination, then it is dropped, otherwise it is forwarded (i.e., broadcast). In our example, node E transmits a data packet destined to A. On receiving this, node C would activate and broadcast this packet. Both B and D would receive this broadcast, but only B will activate to forward it further. Finally, B's broadcast is received by A.

Route truncation attack: Assume that F is an attacker who wishes to intercept the above data packet. First note that D's table is set to discard data packets received from C and addressed to A. Thus, such packets would never be received by F. To launch its attack, F claims a shorter route to A than C by replacing its routing update of Step 5 with the following:

$$F \rightarrow: \mathsf{S}_A(A \Leftarrow),\ \mathsf{S}_F(A \Leftarrow F)$$

On receiving this update, D would believe that the route to A via F is shorter than that via C. Consequently, every data packet sent by E and addressed to A will be received by F. This general attack is called *route truncation*.

One way to circumvent this attack is to use Secure-BGP (S-BGP) [6,9,7].

S-BGP updates: S-BGP updates are recipient specific, and therefore every node is required to be aware of its immediate neighbors. Nodes authorize their neighbors using Route Attestations (which are essentially the signatures in the updates shown below). Let X and Y denote nodes within E's and F's coverages respectively, but not covered by others. The S-BGP updates are as follows.

1. $A \rightarrow: \mathsf{S}_A(A \Leftarrow B)$
2. $B \rightarrow: \mathsf{S}_A(A \Leftarrow B),\ \mathsf{S}_B(B \Leftarrow C)$
3. $C \rightarrow: \mathsf{S}_A(A \Leftarrow B),\ \mathsf{S}_B(B \Leftarrow C),\ \mathsf{S}_C(C \Leftarrow D)$
   $C \rightarrow: \mathsf{S}_A(A \Leftarrow B),\ \mathsf{S}_B(B \Leftarrow C),\ \mathsf{S}_C(C \Leftarrow E)$
4. $D \rightarrow: \mathsf{S}_A(A \Leftarrow B),\ \mathsf{S}_B(B \Leftarrow C),\ \mathsf{S}_C(C \Leftarrow D),\ \mathsf{S}_D(D \Leftarrow F)$
   $E \rightarrow: \mathsf{S}_A(A \Leftarrow B),\ \mathsf{S}_B(B \Leftarrow C),\ \mathsf{S}_C(C \Leftarrow E),\ \mathsf{S}_E(E \Leftarrow X)$
5. $F \rightarrow: \mathsf{S}_A(A \Leftarrow B),\ \mathsf{S}_B(B \Leftarrow C),\ \mathsf{S}_C(C \Leftarrow D),\ \mathsf{S}_D(D \Leftarrow F),\ \mathsf{S}_F(F \Leftarrow Y)$

Drawbacks of S-BGP: Although secure against route truncation, S-BGP has two drawbacks: (1) Each router must have knowledge of its neighbors, and (2) Router C can no longer broadcast the same update for every neighbor. That is, firstly, every router must discover all its neighbors, and secondly, since updates are peer-specific, therefore route changes would result in a large number of broadcasts by a node with many neighbors. This causes scalability issues.

Stateless Routing: Scalability could be improved if the routing protocol resisted route truncation attacks without requiring nodes to have prior knowledge of neighbors, thereby allowing a single broadcast per update. We call such a protocol *stateless* because nodes need not maintain states of their neighbors.

Related Work: Existing works are based on S-BGP and try to reduce the number and/or processing time of signatures. For instance, aggregate signatures keep the payload to a constant size [10]. *Signature Amortization* [11] coupled with (sequential) aggregate signatures [12] reduce the size of updates and signing time [6]. All the above works, however, assume that information about neighbors is known a priori before updates are sent, and are therefore not stateless.

Our Contribution: We present a stateless S-BGP variant, called Stateless S-BGP (SS-BGP), which is an optimized version of the protocol in [8].

It should be emphasized that we address only the scalability issues arising from using S-BGP in wireless networks, and not the security issues. In particular, any security issues arising from the use of S-BGP will be inherited by SS-BGP.

3 The BGP Routing Protocol

Our protocol is an extension of the basic BGP protocol [1], which we describe in slightly more detail here. In BGP, routing updates propagate in a tree structure rooted at the destination. For $i > 0$, consider any node at level $i+1$ of the tree, and label as $R_1, R_2, \ldots, R_{i+1}$, the nodes in the path from the root to this node (both inclusive). Denote by Sign_i, Verify_i the sign and verify functions of node R_i under an existentially unforgeable signature scheme. R_0 is a constant string used for notational convenience. BGP has two phases: Initialize and Update.

Initialize. Let t_1 be a timestamp. The initiator (R_1) sets $m_1 \leftarrow (R_0, R_1, t_1)$; $sig_1 \leftarrow \mathsf{Sign}_1(m_1)$; and $U_1 \leftarrow (m_1, sig_1)$. It broadcasts U_1.

Update. On receiving update U_i, node R_{i+1} sets $m_{i+1} \leftarrow (R_i, R_{i+1}, t_{i+1})$, where t_{i+1} is a timestamp. The update phase consists of two stages:

1. *Validation:* Parse U_i as $(m_1, sig_1), (m_2, sig_2), \ldots, (m_i, sig_i)$. Then ensure the following:
   (a) For each $j \in [1..i] : m_j$ is of the form (R_{j-1}, R_j, t_j).
   (b) For each $j \in [1..i] : R_{j+1} \notin \{R_1, R_2, \ldots, R_j\}$.
   (c) The route to R_1 given by the ordered tuple $(R_1, R_2, \ldots, R_i)$ is either new or better than an existing route.
   (d) For each $j \in [1..i]$: The difference in timestamps, $t_{j+1} - t_j$ is within a pre-defined threshold.

(e) For each $j \in [1..i] : \mathsf{Verify}_j(m_j, sig_j) = \mathsf{VALID}$.
Abort if any of the above checks fail, otherwise, proceed to the next step.

2. *Propagation:* Set $sig_{i+1} \leftarrow \mathsf{Sign}_{i+1}(m_{i+1})$ and $U_{i+1} \leftarrow (U_i, (m_{i+1}, sig_{i+1}))$. Update routing table and broadcast U_{i+1}.

We say that an update is *accepted* if it is successfully validated and propagated.

Correctness: Step 1(a). ensures that the update is of the correct format. If all nodes are honest then the only other steps where validation can fail are

- Step 1(b), when $R_{j+1} \in \{R_1, R_2, \ldots, R_j\}$ for some $j \in [1..i]$. For instance, referring to Fig. 1, when C's routing update reaches B.
- Step 1(c), when the existing routing table has a better route.

In either case, the protocol behaves correctly and implements BGP [1].

Security: The security is captured in the Validation stage as follows:

- Step 1(d) prevents replay attacks.[1]
- Step 1(e) ensures each node $j \in [1..i]$ accepted this update.

Weakness: The weakness in this protocol is that although Step 1(e). ensures that each node $j \in [1..i]$ accepted this update, it does not ensure that these are the only nodes that accepted this update. Specifically, it does not prevent the route truncation attack discussed in Section 2. For instance, when R_{i+1} receives this update, it cannot ensure that R_i received this update from R_{i-1}, since R_i could have truncated several intermediate entries.

Statelessness: The primary advantage is that of statelessness - any node R_i never needs knowledge of R_{i+1}. Therefore, there is no need to establish knowledge of immediate neighbors in order to use the protocol. Secondly, since update R_i is independent of R_{i+1}, the same update can be used by several nodes at level $i+1$.

4 Enhanced Chain Signatures

Notation: We first give some notation to deal with ordered elements.

1. A *sequence* is similar to a set except that the order of its elements is important. Elements of a sequence are written in order from left to right, and enclosed within the symbols $\langle, \rangle$. For instance, $\langle y_1, y_2, y_3 \rangle$ is a sequence. The symbol θ denotes the empty sequence with zero elements.
2. Let $\ell_a = \langle y_1, y_2, \ldots, y_k \rangle$ be some non-empty sequence. For any other sequence ℓ_b, we say that $\ell_b \prec \ell_a$ if and only if $\ell_b = \langle y_1, y_2, \ldots, y_i \rangle$ and $0 \leq i \leq k$. We say that two sequences $\{\ell_a, \ell_b\}$ **overlap** if there exists a non-empty sequence ℓ such that $\ell \prec \ell_a$ and $\ell \prec \ell_b$. For instance, $\{\langle x, y \rangle, \langle x \rangle\}$ overlap, while $\{\langle x, y \rangle, \langle y \rangle\}$ do not.

[1] Note that neither BGP nor S-BGP uses timestamps. In S-BGP, an expiration time carried in RAs is used to prevent replay attacks [7]. The timestamps here provide greater flexibility and may be used in conjunction with an expiration time in m_i.

3. For any two sequences ℓ_a, ℓ_b, the symbol $\ell_a \cup \ell_b$ denotes the **set** of elements that belong to at least one of $\{\ell_a, \ell_b\}$. Similarly $\ell_a \cap \ell_b$ denotes the **set** of elements that belong to both ℓ_a and ℓ_b. We denote by $\ell_a \odot \ell_b$ to be the **set** of elements from the largest sequence ℓ such that $\ell \prec \ell_a$ and $\ell \prec \ell_b$. For example, $\langle x, y, z\rangle \odot \langle x, z, y\rangle = \{x\}$ and $\langle x, y, z\rangle \cap \langle x, z, y\rangle = \{x, y, z\}$
4. Any Sequence $\langle \langle y_1, y_2, \ldots y_i\rangle, y_{i+1}\rangle$ is equivalent to $\langle y_1, y_2, \ldots y_{i+1}\rangle$.

We are now ready to describe Enhanced Chain Signatures (ECS), on which our routing protocol is based. As mentioned earlier, ECS are an extension of Chain Signatures (CS) proposed in [8]. The difference being that, while in CS, every signer signs the same message, in ECS, signers may sign different messages.

Algorithms: ECS are defined by 3 algorithms: **ECS-(KeyGen**, **Sign**, **Verify**).

ECS-KeyGen(τ) takes input $\tau \in \mathbb{N}$. It outputs a private-public key pair (x, y).

ECS-Verify(ℓ, σ) takes input $\ell = \langle (m_1, y_1), (m_2, y_2), \ldots\rangle$, a finite sequence of (message, public key) pairs, and a string σ.

1. If $\ell = \theta$ and $\sigma = 1$, the algorithm outputs VALID.
2. If $\ell = \theta$ and $\sigma \neq 1$, the algorithm outputs INVALID.
3. If any public key y_i repeats in ℓ, the algorithm outputs INVALID.
4. If this step is executed, the algorithm invokes a deterministic poly-time procedure after which it outputs either VALID or INVALID.

ECS-Sign$(x_i, y_i, m_i, \ell_j, \sigma_j)$ takes five inputs, which can be grouped into three parts: (1) a (private key, public key, message) tuple (x_i, y_i, m_i), (2) a sequence $\ell_j = \langle (m_1, y_1), (m_2, y_2), \ldots, (m_j, y_j)\rangle$ of j (message, public key) pairs for $j \geq 0$, and (3) a string σ_j (a purported ECS signature on ℓ_j).

1. If $y_i \in \{y_1, y_2, \ldots, y_j\}$, the algorithm outputs ERROR.
2. If **ECS-Verify**$(\ell_j, \sigma_j) =$ INVALID, the algorithm outputs ERROR.
3. If this step is executed, the algorithm computes an ECS signature σ_i on ℓ_i, where $\ell_i = \langle \ell_j, (m_i, y_i)\rangle$. It outputs (ℓ_i, σ_i).[2]

A string σ is an ECS signature on a sequence ℓ if **ECS-Verify**$(\ell, \sigma) =$ VALID.

Correctness: If the input (ℓ, σ) to **ECS-Verify** is the output of **ECS-Sign** then **ECS-Verify** algorithm must output VALID.

Difference with a CS scheme: If for every sequence $\langle (m_1, y_1), (m_2, y_2), \ldots\rangle$ to be signed, holds $\forall i : m_i = m_1$, then the ECS scheme is also a CS scheme.

Security Model: As in [8], we define the security of ECS under *adaptive known key and chosen message attack*. We define two variants using a parameter $\omega \in \{1, 2\}$ such that setting $\omega = 1$ results in the weaker variant.

Game ECS-UNF$_\omega(\tau)$

[2] Observe that in addition to the ECS signature σ_i on ℓ_i, we require **ECS-Sign** to also output the sequence ℓ_i. This is purely for notational convenience.

Setup. The challenger $\mathcal{C}$ gives the security parameter τ to the adversary $\mathcal{A}$, who then selects a game parameter n (the number of public keys) and an n bit string $extr$. Let $extr[i]$ denote the i^{th} bit of $extr$.

On receiving $(n, extr)$, $\mathcal{C}$ generates $(x_i, y_i) \stackrel{R}{\leftarrow}$ **ECS-KeyGen**(τ) for $i \in [1..n]$ and gives the set $Y = \{y_i\}_{1 \leq i \leq n}$ of n public keys along with set $X = \{x_i | extr[i] = 1\}_{1 \leq i \leq n}$ of extracted private keys to $\mathcal{A}$.

In the following, we denote by L the set of all non-empty sequences of pairs $(m, y) \in \{0,1\}^* \times Y$ such that no y is repeated.

Queries. Working adaptively, $\mathcal{A}$ makes queries as follows:

1. *Extract*(y): If $y \in Y \wedge \omega \neq 1$, $\mathcal{C}$ responds with the private key for y, otherwise it returns $\perp$.
2. *ECS-Sign*(ℓ): If $\ell \in L$, $\mathcal{C}$ responds with a valid ECS signature on ℓ, otherwise it returns $\perp$.

Output. $\mathcal{A}$ outputs a pair $(\ell_A, \sigma_A) \in L \times \{0,1\}^*$.

Result. $\mathcal{A}$ wins if **ECS-Verify**$(\ell_A, \sigma_A) = \mathsf{VALID}$ and ℓ_A is *non-signable* (Def. 1).

Definition 1. ***(Non-signable Sequence)*** *In the game, let $Y_X \subset Y$ and $L_S \subset L$ be the set of inputs to the extract and ECS-sign queries respectively (Note: $\{y_i | extr[i] = 1\}_{1 \leq i \leq n} \subseteq Y_X$). For any $\ell_A \in L$, define the set $L_A = \{\ell_i | \ell_i \in L_S \wedge \{\ell_A, \ell_i\}$ overlap$\}$. We say ℓ_A is* non-signable *if $\ell_A \notin L_S$, and:*

1. $\{(m_i, y_i) | (m_i, y_i) \in \ell_A \wedge y_i \notin Y_X\} \neq \emptyset$.
2. $\forall \ell_i \in L_A : \{(m_j, y_j) | (m_j, y_j) \in (\ell_i \cup \ell_A) \backslash (\ell_i \odot \ell_A) \wedge y_i \notin Y_X\} \neq \emptyset$.

In the following definition, we also consider the use of hash functions.

Definition 2. *The ECS scheme Σ is $(n, \tau, t, q_s, q_e, q_h, \epsilon)$-***UNF***-$\omega$-secure for $\omega \in \{1, 2\}$ if, for game parameters τ and n, there is no adversary $\mathcal{A}$ that runs for time at most t; makes at most (q_s, q_e, q_h) chain-sign, extract and hash queries respectively; and wins Game ECS-UNF$_\omega(\tau)$ with probability at least ϵ.*

Discussion: Roughly speaking, the above security notions imply security under two types of forgeries [8]. We illustrate this with an example. The first forgery (called *ordinary forgery*) occurs when the adversary manages to output a valid ECS signature on a sequence, say $\ell = \langle (m_1, y_1), (m_2, y_2) \rangle$ after making an ECS-sign query on $\ell_1 = \langle (m_1, y_1) \rangle$ but without making an extract query on y_2. This is the type of forgery that all multisignatures schemes (including the ones discussed in Section 2) resist. The second type of forgery in ECS (called *extraction forgery*) occurs when the adversary manages to output a valid ECS signature on $\ell = \langle (m_1, y_1), (m_2, y_2) \rangle$ after making an ECS-sign query on $\ell_3 = \langle (m_1, y_1), (m_2, y_2), (m_3, y_3) \rangle$ but without making an extract query on y_3 and one of $\{y_1, y_2\}$. A scheme secure against an extraction forgery is said to be *truncation resilient*. Note that neither aggregate signatures [10], nor sequential aggregate signatures [12,13] consider extraction forgery.

Construction: A construction of ECS is given in Appendix A.

5 Stateless Secure BGP Using ECS

This protocol fixes the weaknesses of BGP. As in Section 3, let $(R_1, R_2, \ldots)$ be any ordered tuple of nodes that would be affected by a given update, and t_i be the timestamp at which the update originated/arrived at node i. Let (x_i, y_i) be the (private, public) key-pair of node R_i under an ECS scheme. Assume that every node has a distinct public key. The SS-BGP protocol is as follows.

Initialize. Initiator R_1 sets $m_1 \leftarrow t_1$ and $(\ell_1, \sigma_1) \leftarrow$ **ECS-Sign**$(x_1, y_1, m_1, \theta, 1)$. Note that $\ell_1 = \langle (m_1, y_1) \rangle$. Finally it sets $L_1 \leftarrow (m_1, R_1)$; $U_1 \leftarrow (L_1, \sigma_1)$ and broadcasts the update U_1.[3]

Update. On receiving update $U_i = (L_i, \sigma_i)$, node R_{i+1} sets $m_{i+1} \leftarrow t_{i+1}$ and does the following:

1. *Validation:* Parse L_i as $(m_1, R_1), (m_2, R_2), \ldots, (m_i, R_i)$. Then ensure the following:
   (a) For each $j \in [1..i] : m_j$ is of the form t_j.
   (b) For each $j \in [1..i] : R_{j+1} \notin \{R_1, R_2, \ldots, R_j\}$.
   (c) The route to R_1 given by the ordered tuple $(R_1, R_2, \ldots, R_i)$ is either new or better than an existing route.
   (d) For each $j \in [1..i]$: The difference in timestamps, $t_{j+1} - t_j$ is within the pre-defined threshold t.
   (e) Construct the sequence $\ell_i = \langle (m_1, y_1), (m_2, y_2), \ldots, (m_i, y_i) \rangle$ and test if **ECS-Verify**$(\ell_i, \sigma_i) = \mathsf{VALID}$.
   Abort if any of the above checks fail, otherwise, update routing table and proceed to the next step.
2. *Propagation:* Set $(\ell_{i+1}, \sigma_{i+1}) \leftarrow$ **ECS-Sign**$(x_{i+1}, y_{i+1}, m_{i+1}, \ell_i, \sigma_i)$ and $L_{i+1} \leftarrow (L_i, (m_{i+1}, R_{i+1}))$. Note that $\ell_{i+1} = \langle \ell_i, (m_{i+1}, y_{i+1}) \rangle$. Finally, set $U_{i+1} \leftarrow (L_{i+1}, \sigma_{i+1})$ and broadcast U_{i+1}.

Correctness: Referring to the basic BGP protocol of Section 3, the only real difference is in Step 1(e). Assuming that the validation in this step always passes, the above protocol then implements BGP in the presence of honest users.

Security: Consider Step 1(e), where **ECS-Verify** is used. Observe that if the ECS scheme is **UNF**-1-secure then the protocol indeed ensures that each node $j \in [1..i]$ accepted this update. Thus, SS-BGP prevents *false route injection*. Next, we consider two further attacks on BGP-type protocols.

Route Truncation attacks: Let us analyze this using the example of Section 2, Fig. 1. The nodes in the path of the update received by F are (A, B, C, D). In the attack based on BGP, given the update

$$\mathsf{S}_A(A \Leftarrow),\ \mathsf{S}_B(A \Leftarrow B),\ \mathsf{S}_C(B \Leftarrow C),\ \mathsf{S}_D(C \Leftarrow D),$$

attacker F extracts the update $\mathsf{S}_A(A \Leftarrow)$. The corresponding SS-BGP update received by F is $U_D = (L_D, \sigma_D)$, where $L_D = ((t_A, A)(t_B, B)(t_C, C)(t_D, D))$, and the corresponding attack would amount to F extracting the ECS signature

[3] Our basic BGP variant only needs a timestamp. However, additional information may be included in m_i at node i. An example of such information is an expiration time of this update (see Footnote 1) or the GPS coordinates of i.

σ_A on ℓ_A given the ECS signature σ_D on ℓ_D. Under the **UNF**-1 security notion of ECS, this is not possible unless F has extracted the private keys of B, C, D. It can be seen that truncation attacks are not possible in SS-BGP.

Repeating Attacks: In this attack, F acts as a man-in-the-middle and simply repeats the message received from D to masquerade as D and make the route one hop shorter. In the more general case, called wormhole attacks [14,15,16,17,18], the adversary repeats the update at one or more distant regions using an out-of-band channel. As observed in [19], such attacks are unavoidable in BGP-type protocols (including both SS-BGP and S-BGP) if an attacking node can forward messages without modification such that its presence is never detected.[4] Thus, S-BGP does not offer any more protection than SS-BGP under these attacks.

Other attacks based on timestamps are possible on SS-BGP. However, these attacks are also possible in S-BGP, and so we do not consider them here.

Overhead: Assuming that public keys can be uniquely identified by identities, the sequences ℓ_i can be constructed at the receiver's end by knowing the timestamps $(t_1, t_2, \ldots, t_i)$ and identities $(R_1, R_2, \ldots, R_i)$. Consequently, the only overhead in this protocol is that of one single ECS signature σ_i, which is an element of G_1 and is less than 30 bytes using the parameters of [8]. Contrast this with basic BGP or S-BGP, both of which incur an overhead of i signatures.

Storage and Performance: The keys are elements of G_1 and can be stored in ≤ 30 bytes [8]. The benchmarks of [21] indicate the following performance estimates of the above protocol (assuming n nodes in the path):

1. **Update Propagation:** one exponentiation and addition in G_1, and one computation of $\mathcal{H}$ (total $<$ 2ms).
2. **Update Verification:** n pairing computations and multiplications in G_2, and n computations of $\mathcal{H}$ (giving $<$ 1 second for $n = 100$).

To conclude, SS-BGP based on ECS is as secure and as computationally efficient as S-BGP based on the signature schemes of [10,13] without the extra overhead of neighbor discovery, multiple signature computation, multiple broadcasts and multiple signatures in each broadcast.

Multiple Updates Aggregation: In the above description, we assumed that each advertisement U_i contains only one route and is transmitted instantaneously. In the real world, each advertisement contains multiple routes and is sent periodically. Fortunately, the ECS scheme allows *signature aggregation* and *aggregate verification* where a large number of ECS signatures are verified at once [10].

SS-BGP using Identity-Based ECS: SS-BGP using Identity-Based ECS (IBECS) would avoid the overhead of public-key management. However, at this stage a

[4] A possible way to detect this attack in wireless networks would be to encode GPS coordinates in the updates (see Footnote 3). It may then be possible to determine if a message was sent via a repeater by making assumptions on a transmitter's range. Other techniques to detect or prevent wormhole attacks are discussed in [17,18,20,15]

construction of IBECS is not known. A possible construction may arise from the Identity-Based Sequential Aggregate Signatures (IBSAS) of [13].

6 Conclusion

The Border Gateway Protocol (BGP) is a path vector protocol allowing nodes to build routing tables without prior knowledge of neighbors, and requires a single broadcast per node irrespective of the number of neighbors. We call such a protocol *stateless*. However, BGP does not resist route truncation attacks. Consequently, modern implementations use a stateful variant of BGP called Secure-BGP (S-BGP). S-BGP requires forwarding nodes to have prior knowledge of immediate neighbors and necessitates as many broadcasts per node as there are neighbors. Although sufficient for wired networks, this causes scalability problems in mobile ad-hoc wireless networks with low data plane traffic.

We presented a secure stateless variant of BGP for use in wireless networks by extending the work of [8]. The protocol, called Stateless Secure BGP (SS-BGP), is an augmentation of BGP that resists route truncation attacks. The main ingredient of SS-BGP is an authentication primitive called Enhanced Chain Signatures (ECS), which is an enhancement of the Chain Signatures (CS) of [8].

In summary, wireless S-BGP incurs the following overhead: (1) neighbor discovery, (2) multiple signature computation, (3) multiple broadcasts, and (4) multiple signatures in each broadcast. Although the signature schemes of [10,13] are able to address issue (4), they do not address the remaining three. SS-BGP based on ECS is as secure and computationally efficient as S-BGP based on the signature schemes of [10,13] without any of the above overheads. This feature makes SS-BGP particularly attractive for wireless networks.

References

1. Rekhter, Y., Li, T., Hares, S.: RFC 4271: A Border Gateway Protocol 4 (BGP-4) (Janaury 2006)
2. Rexford, J., Wang, J., Xiao, Z., Zhang, Y.: BGP routing stability of popular destinations. In: ACM SIGCOMM IMW (Internet Measurement Workshop) (2002)
3. Estrin, D., Rekhter, Y., Hotz, S.: A Unified Approach to Inter-Domain Routing. RFC 1322 (Informational) (May 1992)
4. Murphy, S.: RFC 4272: BGP security vulnerabilities analysis
5. Mahajan, R., Wetherall, D., Anderson, T.: Understanding BGP misconfiguration. In: SIGCOMM '02: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications, pp. 3–16. ACM Press, New York (2002)
6. Zhao, M., Smith, S.W., Nicol, D.M.: Aggregated path authentication for efficient BGP security. In: CCS '05: Proceedings of the 12th ACM conference on Computer and communications security, pp. 128–138. ACM Press, New York (2005)
7. Kent, S.: Securing the border gateway protocol. The Internet Protocol Journal 6(3), 2–14 (2003)

8. Saxena, A., Soh, B.: One-way signature chaining: A new paradigm for group cryptosystems. International Journal of Information and Computer Security 2(3), 268–296 (2008)
9. Kent, S.: Securing the border gateway protocol: A status update. In: Seventh IFIP TC-6 TC-11 Conference on Communications and Multimedia Security, pp. 2–3 (2003)
10. Boneh, D., Gentry, C., Lynn, B., Shacham, H.: Aggregate and verifiably encrypted signatures from bilinear maps. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 416–432. Springer, Heidelberg (2003)
11. Park, J.M., Chong, E.K.P., Siegel, H.J.: Efficient multicast packet authentication using signature amortization. In: SP '02: Proceedings of the 2002 IEEE Symposium on Security and Privacy, Washington, DC, USA, p. 227. IEEE Computer Society, Los Alamitos (2002)
12. Lysyanskaya, A., Micali, S., Reyzin, L., Shacham, H.: Sequential aggregate signatures from trapdoor permutations. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 74–90. Springer, Heidelberg (2004)
13. 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 '07: Proceedings of the 14th ACM conference on Computer and communications security, pp. 276–285. ACM, New York (2007)
14. Karlof, C., Wagner, D.: Secure routing in wireless sensor networks: Attacks and countermeasures. In: First IEEE International Workshop on Sensor Network Protocols and Applications, pp. 113–127 (2002)
15. Poovendran, R., Lazos, L.: A graph theoretic framework for preventing the wormhole attack in wireless ad hoc networks. Wirel. Netw. 13(1), 27–59 (2007)
16. Wang, W., Bhargava, B.: Visualization of wormholes in sensor networks. In: WiSe '04: Proceedings of the 3rd ACM workshop on Wireless security, pp. 51–60. ACM, New York (2004)
17. Maheshwari, R., Gao, J., Das, S.R.: Detecting wormhole attacks in wireless networks using connectivity information. In: INFOCOM, pp. 107–115. IEEE, Los Alamitos (2007)
18. Hu, Y.-C., Perrig, A., Johnson, D.B.: Wormhole attacks in wireless networks. IEEE Journal on Selected Areas in Communications 24(2), 370–380 (2006)
19. Torgerson, M., Leeuwen, B.V.: Routing data authentication in wireless networks. Technical Report SAND2001-3119, Sandia National Labs., Albuquerque, NM (US); Sandia National Labs., Livermore, CA, US (2001)
20. Khabbazian, M., Mercier, H., Bhargava, V.K.: Severity analysis and countermeasure for the wormhole attack in wireless ad hoc networks. Trans. Wireless. Comm. 8(2), 736–745 (2009)
21. Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. Cryptology ePrint Archive, Report 2005/028 (2005)
22. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. J. Cryptology 17(4), 297–319 (2004)
23. Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. SIAM J. Comput. 32(3), 586–615 (2003)

A Construction of ECS

The following construction of ECS is adapted from [8].

1. **Bilinear Maps:** Let G_1 and G_2 be two cyclic multiplicative groups both of prime order q such that computing discrete logarithms in G_1 and G_2 is intractable. A bilinear pairing is a map $\hat{e} : G_1 \times G_1 \mapsto G_2$ that satisfies the following properties [22,23,10].
   - *Bilinearity*: $\hat{e}(a^x, b^y) = \hat{e}(a, b)^{xy}$ $\forall a, b \in G_1$ and $x, y \in \mathbb{Z}_q$.
   - *Non-degeneracy*: If g is a generator of G_1 then $\hat{e}(g, g)$ is a generator of G_2.
   - *Computability*: The map $\hat{e}$ is efficiently computable.

   In a practical implementation, G_1 is a subgroup of the additive group of points on the elliptic curve and G_2 is the multiplicative subgroup of a finite field [22,23]. The security of our protocol depends on the hardness of the Computational Diffie-Hellman (CDH) problem in G_1 defined as follows: given $g, g^x, g^y \in G_1$ for $g \notin \{0_{G_1}, 1_{G_1}\}$, and unknowns x, y, compute $g^{xy} \in G_1$ [10].
2. **Common Parameters:** Let $\hat{e} : G_1 \times G_1 \mapsto G_2$ be a bilinear map over cyclic multiplicative groups (G_1, G_2) of prime order q, and g be any generator of G_1. The prime q is chosen so that the CDH problem in G_1 is requires $\approx 2^\tau$ operations for some security parameter τ. See [10] for details. Let $\mathcal{H} : \{0,1\}^* \mapsto G_1$ be a hash function. In the rest of this section, these parameters will be considered common and public.
3. **The Algorithms:** The following are the three algorithms in the scheme.

   ECS-KeyGen. Each user j generates $x_j \xleftarrow{R} \mathbb{Z}_q$. The private key of j is x_j. The corresponding public key is $Y_j = g^{x_j} \in G_1$.

   ECS-Sign. Let $\ell_i = \langle (m_1, Y_1), (m_2, Y_2), \ldots, (m_i, Y_i) \rangle$ be some sequence of i distinct (message, public key) pairs. A valid ECS signature on sequence ℓ_i is an element $\sigma_i \in G_1$, where:

$$\sigma_i = \prod_{j=1}^{i} \mathcal{H}(\langle (m_1, Y_1), (m_2, Y_2), \ldots, (m_j, Y_j) \rangle)^{x_j},$$

   $\ell_0 = \theta$ and $\sigma_0 = 1_{G_1}$. Given $(\ell_{i-1}, \sigma_{i-1})$, the ECS signature σ_i on ℓ_i is computed by user $i \in \{1, 2, \ldots\}$ as

$$\sigma_i \leftarrow \sigma_{i-1} \cdot \mathcal{H}(\ell_i)^{x_i}.$$

   ECS-Verify(ℓ_i, σ_i). Let $\ell_i = \langle (m_1, Y_1), (m_2, Y_2), \ldots, (m_i, Y_i) \rangle$. To verify the signature σ_i, check that all Y's are distinct and the following holds:

$$\hat{e}(\sigma_i, g) \stackrel{?}{=} \prod_{j=1}^{i} \hat{e}(\mathcal{H}(\langle (m_1, Y_1), (m_2, Y_2), \ldots, (m_j, Y_j) \rangle), Y_j).$$

4. *Security:* The CS scheme of [8] is shown to be **UNF**-1-secure. We observe that the same proof of [8] can be made to work for the above ECS construction with an appropriate modification to the simulator. Hence, the above ECS construction is also **UNF**-1-secure in the random oracle model under the computational Diffie-Hellman (CDH) assumption in bilinear maps. Due to lack of space, the proof is deferred to the full version of this paper.

   Whether the above ECS scheme is also **UNF**-2-secure is an open question.

Probabilistic Vehicular Trace Reconstruction Based on RF-Visual Data Fusion

Saif Al-Kuwari[1,2] and Stephen D. Wolthusen[1,3]

[1] Information Security Group, Department of Mathematics, Royal Holloway, University of London, Egham Hill, Egham TW20 0EX, United Kingdom
[2] Information Technology Center, Department of Information and Research, Ministry of Foreign Affairs, P.O. Box 22711, Doha, Qatar
[3] Norwegian Information Security Laboratory, Gjøvik University College, P.O. Box 191, N-2802 Gjøvik, Norway

Abstract. Geolocation information is not only crucial in conventional crime investigation, but also increasingly important for digital forensics as it allows for the logical fusion of digital evidence that is often fragmented across disparate mobile assets. This, in turn, often requires the reconstruction of mobility patterns. However, real-time surveillance is often difficult and costly to conduct, especially in criminal scenarios where such process needs to take place clandestinely. In this paper, we consider a vehicular tracking scenario and we propose an offline post hoc vehicular trace reconstruction mechanism that can accurately reconstruct vehicular mobility traces of a target entity by fusing the corresponding available visual and radio-frequency surveillance data. The algorithm provides a probabilistic treatment to the problem of incomplete data by means of Bayesian inference. In particular, we realize that it is very likely that a reconstructed route of a target entity will contain gaps (due to missing trace data), so we try to probabilistically fill these gaps. This allows law enforcement agents to conduct off-line tracking while characterizing the quality of available evidence.

Keywords: Tracking, Trace, Bayesian, Scene Reconstruction, Vehicular, Fusion.

1 Introduction

Wireless mobile devices such as mobile phones and laptops can provide useful geolocation information. However, while such data enables novel ways of *tracking* mobile entities, collecting it from multiple sources and logically correlate it often creates new challenges for digital forensics.

Generally, tracking can either be online or offline. Online tracking involves observing the movement of a target entity in real time and is usually reactive (and adaptive) according to the target's behaviors. Offline tracking, on the other hand, entails the extraction of the target's movement traces from raw tracking data to be analyzed. However, offline tracking, in most cases, encounters a problematic issue that we call *tracking gaps*. These gaps represent missing tracking data over particular areas/periods where tracking of the target was not possible, usually due to tracking resources constrains. In this case, it is important to try filling these gaps by probabilistically guessing the routes

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 16–27, 2010.

the target would have most likely taken between the end points of the gaps. For a gap, these end points are called the *Ingress*, which is the point at which tracking of the target was lost marking the beginning of a gap, and the *Egress*, which is when tracking later resumed marking the end of that gap. While such scenarios are certainly important to consider for tracking individuals, we instead discuss vehicular tracking since this is the most common means of transportation. Vehicular tracking data can be collected by explicitly tracking the target vehicle, or by observing data from the existing traffic infrastructure (e.g. CCTV) that is not originally meant to be involved in a tracking process. Either way, we assume that this data was collected passively since this is required by most law enforcement applications. Such data, and beside containing tracking gaps, will certainly exhibit different types of measurement errors, thus we develop a probabilistic method based on basic Bayesian inference to reconstruct the target trace, noting that this method can be adapted for an online mobility prediction too, which is a natural extension to an offline reconstruction, but this is beyond the scope of this paper.

This paper is organized as follows: in section 2 we briefly review some related work. In section 3 we discuss the feasibility of fusing RF and visual traces to minimize the number of tracking gaps in a target's trace. The trace reconstruction algorithm is proposed in section 4, followed by a discussion about its accuracy in section 5. In section 6 a few simulation results are presented, and finally, the paper concludes in section 7.

2 Related Work

Incomplete (or missing) data is a common problem in many applications. Expectation-Maximization and Data Augmentation methods (along with their improved variants) are among the most popular statistical treatments for the missing data problem [1]. Similarly, scene reconstruction (which usually deals with missing data) has been a highly active area of research over the past few decades, especially for forensics and crime investigation purposes. In general, most of the research in this area involves reconstructing scenes from images. For example, in [2] Calbi *et al.* proposed a set of computer-vision-based algorithms that are able to reconstruct a 3D scene of multiple moving objects. However, this system depends on pre-installed camera entities surveilling the area to be reconstructed. Moreover, in [3], Conaire *et al.* discussed how to fuse image-based and RF-based localization to improve the overall accuracy of the process. The authors tested their mechanism in a museum environment where visitors are equipped with devices containing a portable camera. The device can take photos for its current location and compare them with a database of images to identify its current location. This information is then fused with an estimation of signal strength from several wireless networks around the museum; again, signal strength histogram was create for various locations in the museum and stored in a database to be compared with the measurements.

Although most of these algorithms were proposed to track individuals, we are explicitly concerned with vehicular tracking. Examples of such works is presented in [4] by Brakatsoulas *et al.* who discuss the feasibility of reconstructing the movement patterns of objects by observing their GPS tracking information. Their algorithm adopts the so-called *map matching* technique where the tracking information of the objects (which are vehicles in this case) are analyzed and applied to a road-map. This paper adapts

a somewhat similar approach to map-matching but for the purpose of reconstructing a full tracking trace of a particular entity, not quite concerned about accurate localization since it suffices to know that a target vehicle has been in a particular roadway to draw conclusions about how its traces can be reconstructed.

3 Trace Fusion

Vehicular traces are records containing movement information of vehicles over a particular area and during a specific period of time. These traces can be collected by various tools, like GPS and Radar. However, data from other tools like CCTV (Closed Circuit Television), which are not originally tailored for tracking, can be used to improve the existing tracking traces. In this paper, we assume that we have two sets of tracking data, one collected by traditional tracking processes (see [5] for a survey on active tracking, and [9] for a work on passive vehicular tracking) and another retrieved from CCTV cameras that happen to be surveilling the tracking scene – we will call the first data *RF (Radio Frequency) tracks*, and the second *visual tracks*. The aim of this preparatory phase is to fuse RF and visual tracks of a target. Note, however, that this is only an auxiliary phase and is not necessarily required to proceed with the reconstruction algorithm, that is, we assume that both RF and visual tracks exhibit tracking gaps, but when fused, we hope that the number of these gaps is minimized.

Interpreting RF tracking data is straightforward, as a minimum, each record consists of time, vehicle ID and location, and we aim to simplify visual tracks (represented by images taken from CCTV cameras) to be consistent with the RF tracks which in turn will simplify the fusion process. Assuming prior knowledge of the fixed locations of the CCTV cameras and their somewhat narrow recording angular distance, we can estimate the location of a detected target to be the location of the detecting camera. One possible detecting method is by the plate number of the vehicles and detect the target once his plate number is observed, this can be done by capturing live images and compare them to a database of images containing the required plate number. Methods like SURF [6], perform such image-based detection by identifying interest points in the images. The main difference between RF and visual tracks, though, is that RF tracks represent continuous movements of the vehicle for a period of time (represented by a set of chronological tracking records), while the visual tracks represent fixed locations of the target where it was detected.

In this fusion process, we further aim to construct a trace of the target containing gaps only between intersections. However, real tracking information may not satisfy this requirement and can possible loose track of the target half way through roadways. In such cases, the available tracks for these roadways (with incomplete traces) may be: (a) visual only, (b) RF only, or (c) both RF and visual. In all these situations, we run a prediction algorithm to estimate the time it took the target to reach the next intersection and thereby filling the whole roadway. However, situation (a) provides very little information for this algorithm to work, so such roadways are ignored and marked as *semi-incomplete* which basically indicates that the target was observed in this roadway (this information may prove useful in the gap-filling algorithm; see section 4). In situations (b) and (c), and for simplify the description, we assume that there are only

single RF and/or single visual tracks per roadway, noting that the algorithm can easily be extended to consider multiple RF/visual tracks. The prediction algorithm is based on estimating the speed of the target over the period covered by the available tracking information which will then be used to predict the time it will take the target to reach the next intersection given the remaining distance of the corresponding roadway. In situation (c), if the visual track is within the RF track range, it provides no extra information (essentially becoming situation (b)) and the visual tracks can be ignored, but if the RF and visual tracks are apart, we extend the range of the RF to include the visual track. Usually, the longer the available tracking range, the better prediction we can expect. After connecting RF and visual tracks, the result is a tracking range and the predicted time to the next intersection can be calculated as follows:

$$t_i = \frac{(d(R)_1 - d(R)_0)(n - (d(R)_0 + d(R)_1))}{t(R)_1 - t(R)_0} \tag{1}$$

where: $d(R)_0$ and $d(R)_1$ denote the beginning and the end of the tracking range, respectively, and $t(R)_0$ and $t(R)_1$ denote the times at which the tracking range started and ended, respectively. If $d(R)_0 \neq d_{I,i}$ (where $d_{I,i}$ is the beginning of the roadway), then this means that the tracking range didn't start at the beginning of the roadway, so we need to do a *backward* time prediction (connecting the beginning of the range with the beginning of the roadway) as well as a *forward* time prediction (connecting the end of the range with the end of the roadway), which can easily be done in a similar manner. Furthermore, if $d(R)_1 = d_{E,i}$ (where $d_{E,i}$ is the end of the roadway), then only backward prediction is required. In any case, once we know the average speed of the target (which we can easily be calculate from $d(R)_0$, $d(R)_1$, $t(R)_0$ and $t(R)_1$), then given a distance, we can modify equation 1 to compute the time it would take the target to drive that distance, regardless of whether it is forward or backward.

4 Trace Reconstruction

Even after RF-visual fusion, the target's tracking records will still most likely include tracking gaps. In this section, we propose a 2-phase algorithm to probabilistically fill such gaps. To simplify the discussion, we assume that the underlying layout of the tracking scene resembles a Manhattan grid (see figure 1), in phase 1, the two end points (Ingress and Egress) of a gap along with all possible routes between them are identified. Then in phase 2, and based on the target's available tracking traces, the driving behaviors of the target is analyzed and used to fill the gaps by selecting the connecting routes that the target would most likely have taken through these gaps.

4.1 Phase 1: Routes Identification

In this phase, the end points, P_{I,G_i} and P_{E,G_i} (the Ingress and Egress, respectively), of a gap, G_i (where $i = 1, 2, .., n$ for tracking trace with n gaps), along with the possible routes between P_{I,G_i} and P_{E,G_i} are identified assuming that P_{I,G_i} and P_{E,G_i} correspond to intersections. However, it may be computationally expensive (or even infeasible) to identify all the possible routes between P_{I,G_i} and P_{E,G_i} when having a

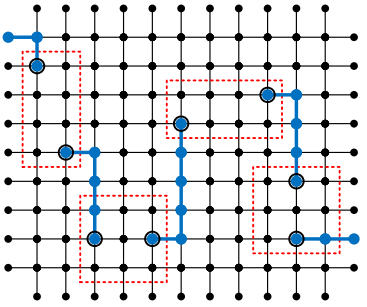


Fig. 1. Sample Scenario

large tracking area, regardless of the size of the gap. Thus, we restrict the area under consideration by setting boundaries around the tracking gap, this bounded area is called *search area* and we expect that this area will cover the most probable routes a target would take through the gap. Figure 1 visually illustrates a sample tracking trace of a target with four tracking gaps. Since we assumed that the tracking area is a Manhattan grid, the search area will be of a rectangular/square shape containing both P_{I,G_i} and P_{E,G_i}. We consider two situations for the locations of the P_{I,G_i} and P_{E,G_i} (we will refer to intersections as vertices and roadways as edges):

- Aligned Ingress/Egress: if P_{I,G_i} and P_{E,G_i} are either aligned horizontally or vertically, they form a straight line, but it is naive to assume that the target used that route to travel from P_{I,G_i} to P_{E,G_i}. Hence, we widen the search area to include the routes through the intersections above and the below (or at the right and left of) P_{I,G_i} and P_{E,G_i}. The search area is then bounded by the rectangle/square whose vertices are: $P_{I,G_i} - 1, P_{I,G_i} + 1, P_{E,G_i} + 1, P_{E,G_i} - 1$ where $+1$ and -1 imply the above and below (or right and left) intersections relative to P_{I,G_i} and P_{I,G_i}, respectively. Finally, all these vertices are marked.
- Diagonal Ingress/Egress: if P_{I,G_i} and P_{E,G_i} are neither aligned horizontally nor vertically, they form a diagonal line and the search area is bounded by the rectangle/square whose vertices are: $P_{I,G_i}, P_{I,G_i} + 1, P_{E,G_i}, P_{E,G_i} - 1$ or $P_{I,G_i} - 1, P_{I,G_i}, P_{E,G_i} + 1, P_{E,G_i}$, depending on the positions of the Ingress and Egress relative to each other. Finally, all these vertices are marked.

Once the four vertices bounding the search area are identified (and marked), they are connected. All the vertices that these connection lines pass through are then marked too indicating that they are part of the bounded search area; we call all marked vertices *search vertices*. We then propose an algorithm, called Bounded Route Counter (BRC), to find all the routes between P_{I,G_i} and P_{E,G_i} that are within the search area. The BRC algorithm is resembles a flooding (or broadcast) algorithm [7]. In the conventional flooding algorithms, the goal is to deliver a message to all nodes within a particular area by configuring nodes to forward every message it receives to all other nodes expect the node it received the message from. Similarly, BRC uses a message-like broadcast mechanism to discover the routes between P_{I,G_i} and P_{E,G_i}. When BRC

algorithm is first executed at P_{I,G_i}, it generates as many messages as there are exit points attached to P_{I,G_i}, each message represents a separate flow. These flows then duplicate or triplicate at every intersection they reach as long as it is a search vertex. This process continues until all flows are terminated. A flow terminates when it reaches: (1) a non-search vertex, (2) the P_{I,G_i} or (3) the P_{E,G_i}. When a flow terminates it raise a special tag with a value of 1, if the flow reached P_{E,G_i}, 0 otherwise. If the termination value was 1, the corresponding flow sends a message back to P_{I,G_i} reporting its traversed path. Algorithm 1 illustrates the BRC algorithm.

It is easy to see that the algorithm will both terminate and find all possible routes from P_{I,G_i} toP_{E,G_i} (that are within the search area). Initially, the algorithm is executed at P_{I,G_i} where it has four possible directions to send the flows through (though, following the flooding algorithm rules, it can't send a flow backward). At least one flow will hit a search vertex and will further propagate since at least one direction out the P_{I,G_i} leads to a search vertex. Also, since a flow can't terminate as long as it is propagating through search vertices and that all vertices will propagate a received flow out all their possible directions, it is guaranteed that all search vertices will be visited and only flows that terminate at the P_{E,G_i} will return to the P_{I,G_i}.

Algorithm 1. Bounded Route Counter Algorithm

```
P_{I,G_i} ← Ingress {identify Ingress}
P_{E,G_i} ← Egress {identify Egress}
Links ← array {which routes connect P_{I,G_i} and P_{E,G_i}}
exitPoints ← no. of exit points attacked to a vertex, usually 4
markSearchVertices(P_{I,G_i}, P_{E,G_i})
flood(P_{I,G_i}) {begin the flooding process}
label Loop
for i = 1 to i = exitPoints − 1 do
    if NonSearchVertexReached = True then
        continue
    else if IngressReached= True then
        continue
    else if EgressReached = true then
        addRoute(Links, i) {add route to the Links array}
        continue
    else if SearchVertexReached = True then
        Tag {tag to becomes a search vertex}
        flood(i)
        goto Loop
    end if
end for
```

We note that due to the visual-RF fusion process, we expect that the size of the search areas is minimized, and so adopting more sophisticated algorithms, such as branch-and-bound, will probably just slightly enhanced the efficiency of the whole route identification process at the cost of unnecessary overall complication.

4.2 Phase 2: Routes Analysis and Selection

At this stage, all routes between P_{I,G_i} and P_{E,G_i} are identified; we will denote the routes as R^i_j, where $i = 1, 2, ..., n$ and $j = 1, 2, ..., f(G_i)$, for the j^{th} route of the i^{th} gap. Also, each R^i_j consists of $f(R^i_j)$ roadways, where $f(R^i_j) = 1, 2, ...$ The function f can be thought of as an *overloading* function which behaves differently depending on its input, that is, f returns the number of routes through a gap when given that gap (e.g. $f(G_i)$), or the number of roadways forming a route when given that route (e.g. $f(R^i_j)$).

In this phase we analyze every route identified in phase 1 individually and estimate the time the target would most likely spend when traveling between P_{I,G_i} and P_{E,G_i} if he use each route, and compare it to the actual time difference between P_{I,G_i} and P_{E,G_i} as obtained from the original incomplete tracking traces. These analyses are based on a basic Bayesian inference by first studying the driving behaviors of the target and then assign probabilities for each possible route through the gap. By investigating the target's driving behavior, we essentially try to devise a mobility model to identify patterns in the target's movement, which is then used in the route selection algorithm. However, prior to the route selection algorithm, we first check whether any of the routes contains *semi-incomplete* roadways. Recall from section 3 that a semi-incomplete roadway is a roadway in which the target was observed at but couldn't be included in the RF-visual fusion process. If a semi-incomplete roadway falls between P_{I,G_i} and P_{E,G_i}, and since we know that the target was indeed in that roadway, we can safely ignore any route not passing through that roadway, which minimizes the number of routes under consideration. Once the possible routes are identified, the algorithm proceeds in 5 steps:

Step 1. After identifying the routes and the roadways each route is composed of, we search the available traces of the target for roadways with similar lengths as those in the routes between P_{I,G_i} and P_{E,G_i}. Then, we calculate the mean (average) of the times the target spent driving those roadways and assign the result to the corresponding roadways of the gap's routes. In this step, we aim to find the mean and variance of the target's driving time for each route through the gap. Suppose we have a route R^i_j that consists of $f(R^i_j)$ roadways, then its mean is:

$$\mu_{R^i_j} = \frac{1}{f(R^i_j)} \sum_{x=1}^{f(R^i_j)} \left(\frac{1}{S(x)} \sum_{y=1}^{S(x)} t_{E,y} - t_{I,y} \right) + \omega_x \tag{2}$$

where $S(x)$ is the number of roadways from the available target's traces with the same length as roadway x, ($x = 1, 2, ..., f(R^i_j)$), and ω is the delay factor which may be different for different roadway, see section 5 for a discussion about how to model this parameter. Since it is easy to extract information about roadways with available tracking information, it is possible to find when the target entered and existed each one of these roadways ($t_{I,y}, t_{E,y}$, respectively, for roadway y). Next, we calculate the corresponding variance of every route as follows:

$$\sigma^2_{R^i_j} = \frac{1}{f(R^i_j)} \sum_{x=1}^{f(R^i_j)} \left(\left(\frac{1}{S(x)} \sum_{y=1}^{S(x)} t_{E,y} - t_{I,y} \right) + \omega_x - \mu_{R^i_j} \right)^2 \tag{3}$$

We then use Bayes' theorem to calculate the probabilities that the target took each route between the Ingress and the Egress of a gap, and select the route with the highest probability as the connecting route. We base our probability calculations on the time difference between the Ingress and the Egress of the gap $t_{G_i} = t_{E,G_i} - t_{I,G_i}$ as obtained from the original trace, that is, for every route, we calculate the probability that the target took that route given the time we obtain by observing the times at P_{I,G_i} and P_{E,G_i}:

$$P(R_j^i|t_{G_i}) = \frac{P(t_{G_i}|R_j^i)P(R_j^i)}{P(t_{G_i})} \tag{4}$$

where: $P(t_{G_i}|R_j^i)$ is the conditional probability that given the target took the route R_j^i, he spent t_{G_i} driving it; which is calculated for every route (with a fixed t_{G_i}). $P(R_j^i)$ is the prior probability that a target will take the route R_j^i, and $P(t_{G_i})$ is the marginal probability. Steps 2 to 4 below show how these probabilities are calculated.

Step 2. Conditional probabilities: the probability that the target spent the time t_{G_i} while driving from $P_{I,i}$ to $P_{E,i}$ through gap G_i given that he took route R_j^i is:

$$P(t_{G_i}|R_j^i) = \frac{1}{\sqrt{2\pi\sigma_{R_j^i}^2}} \exp\left[-\frac{(t_{G_i} - \mu_{R_j^i})^2}{2\sigma_{R_j^i}^2}\right] \tag{5}$$

This probability is calculated using Gaussian probability density function assuming that the underlying process is Gaussian [8], that is, the driving behavior of the target will most likely follow a Gaussian distribution or can be estimated as Gaussian. In fact, it is easy to see that this process is Gaussian because typical driving behavior is to speed at the middle of a roadway and slow down at the beginning/end of that roadway, while driving with an average speed elsewhere.

Step 3. Prior probability: since we don't have enough information to model the target's preferences, the probability that the target selects a particular route (prior probability) through a gap is uniform for all available routes, and can be calculated as follows:

$$P(R_j^i) = \frac{1}{\sum_{x=1}^{f(G_i)}} \tag{6}$$

where $f(G_i)$ is the total number of routes through the gap G_i. Although we assumed that taking any route is equally likely, in practice and in some situations, some routes are more likely to be taken by the target than others. This may be due to traffic flow condition for example, see section 5 for a discussion about such factors. Another way to model this can be done by adopting a mobility prediction algorithm, such as that presented in [9] which predicts the direction a target vehicle would most likely take out an intersection by observing its current lane as it is approaching that intersection. However, carrying out such algorithms is difficult for offline tracking.

Step 4. Marginal probability: the marginal probability $P(t_{G_i})$ is the sum of the conditional (step 2) and prior (step 3) probabilities of all the routes and is calculated as

follows (the marginal probability acts as a normalizing constant in the sense that it makes sure that all the Bayesian probabilities of the routes will sum up to 1):

$$P(t_{G_i}) = \sum_{j=1}^{f(G_i)} P(t_{G_i}|R_j^i)P(R_j^i) \tag{7}$$

Step 5. Finally, and using equation 4, we can now calculate the Bayesian probabilities for each route and select the route with the highest probability as the most likely route the target would have taken through the corresponding gap.

5 Estimation Accuracy

The accuracy of our reconstruction algorithm is influenced by a number of factors, mainly concerning the accuracy of the tracking data collection. Beside the conventional RF measurement errors, it's very likely that the tracking traces are collected by several entities, so unless these entities are tightly synchronized, there will be timing errors among the recorded traces. Traffic-wise, the traffic delay factor ω (which models the various delays sources) influences the accuracy of the algorithm, and is explicitly used in equations 2 and 3. Examples of factors influencing ω are:

Roadways lengths: Routes are most likely composed of several roadways that are usually of different lengths. However, while the accumulation of the lengths of the roadways that form a route is representative to the distance between the Ingress and the Egress through that route, the speed of the movement through this route is affected by the acceleration and deceleration during the journey and around the intersections connecting the route's roadways. Hence, the speed of the target should be estimated for the individual roadway as oppose to the speed over the whole route.

Roadways speed limits: Every roadway restricts the speed of vehicles to a specific speed limit threshold. Knowledge of these limits is useful for estimating the maximum time threshold during which a vehicle can pass the corresponding roadway.

Traffic management type: Traffic delay highly depends on the traffic management type. For example, it is very likely that an intersection managed by stop signs will experience longer delays than another managed by traffic lights. However, care should be taken when considering traffic lights because the delay due to traffic lights depends on the state of the traffic light upon arrival (i.e. vehicles reaching the intersection while the traffic light indicates green will most likely experience much less delay than otherwise).

Points of Interest (POI): Another very effective traffic factor is the existence of points of interests. These points represents locations that are frequently visited by people, like banks and grocery stores, and hence represent locations of common interest among people. Intuitively, the existence of such points along a roadway will very likely increase the traffic density at that roadway and, consequently, the traffic delay. We note that information about POI can be extracted from a few specialized maps (like Google maps) and integrated into the weighting assignment of this phase.

Traffic density and flow: If available, knowledge of vehicular density (number of vehicles per km) and flow (number of vehicles crossing a point per hour) is very useful. Such information can either be statistically estimated from real traces, or probabilistically inferred based on location and time. For example, a particular area may be increasingly crowded during a particular period of time of the day, like an intersection leading to offices which may be crowded only at early morning and late afternoon.

Clearly, obtaining the information above to model ω is extremely difficult, so in section 6, we propose a method to model ω without having to access extra information about the traffic, that is, we estimate the traffic flow of the concerned roadways by referring to the original tracking traces (before extracting the target's trace from it). However, we also note that beside the explicit modeling of ω, we also implicitly accounted for ω (at least partially) when we calculated the average driving time in step 1 which already includes implicit traffic delays.

6 Simulation and Validation

Due to the difficulty of obtaining real vehicular traces to validate our probabilistic trace reconstruction algorithm, we used a vehicular mobility simulator to generate artificial vehicular traces for different scenarios. In this simulation, we used VanetMobiSim simulator [10] to generate vehicular mobility traces based on IDM-LC (IDM with Lane Changes) mobility model [11] which is an extensions to the IDM (Intelligent Driver Motion) mobility model [12]. These traces are then fed into NS-2 simulator [13] to generate the movement of the entities (i.e. vehicles) and a trace database. Arbitrarily appointing one of the simulated entities as a target, we manually (and randomly) created gaps in that target's mobility traces after extracting it from the original trace database. We then executed our trace reconstruction algorithm to probabilistically choose the most likely routes to connect those gaps and compare the selected routes with the routes the target actually took. Since it is difficult to model the delay factor ω without having access to real traffic traces, we modeled ω by observing the node density on the roadways that the gaps' routes are composed of. In particular, we referred to the original mobility traces for all the simulated nodes (before extracting the target's trace) and looked at traces taken for any node that happen to be passing through any of the roadways that are part of the gap's routes, then estimated the average node density of these roadways to assign values for their corresponding ω (clearly, the higher the node density, the larger the value of ω). This is easy to do in practice too since the traces from which the target trace was extracted usually contain other information about other entities that we can use to model the delay factors, which then can be used in equations 2 and 3. The delay factor ω is basically a time delay assigned to individual roadway and is different for different roadways. Figure 2 illustrates our simulation results for scenarios with different node density over a 1000 m^2 area consisting of roadways with different lengths –every simulation was run for 100 seconds. In each scenario, we manually created a gap (with 4 possible routes between the Ingress

and the Egress) and run the algorithm to select the most probable route. The figure shows the probabilities of each of the four possible routes at the manually created gap in each scenario. The results shown in figure 2 indicates that our reconstruction algorithm along with our method of modeling ω by averaging the traffic flow of roadways works very well in scenarios with lighter node densities where the probabilities of the routes vary drastically and it is easy to see which route is the most probable, but as node density increases, the probabilities become closer. Although we argue that this way of modeling ω is, in most cases, efficient since it gives a good estimation of other traffic delay factors affecting the roadways without actually having to model them individually, further modeling may be required in the more cluttered scenarios. However, there are always some routes that can be easily ruled out (like route 4 in all scenarios) usually because they introduce much longer delay compared to the time the target trace was missing over a gap.

We believe that if these algorithms were applied to real traces, the results will be more accurate since most of the simulation-based mobility models (which we used) don't always maintain a tightly consistent driving behavior for every entity, so modeling the driving behavior of the target based on his history trace is slightly less accurate in this case. In real life scenarios, on the other hand, every driver has a unique driving behavior, in fact, recent work [14] even showed that the driving behavior of individuals would make a reasonable biometric measure.

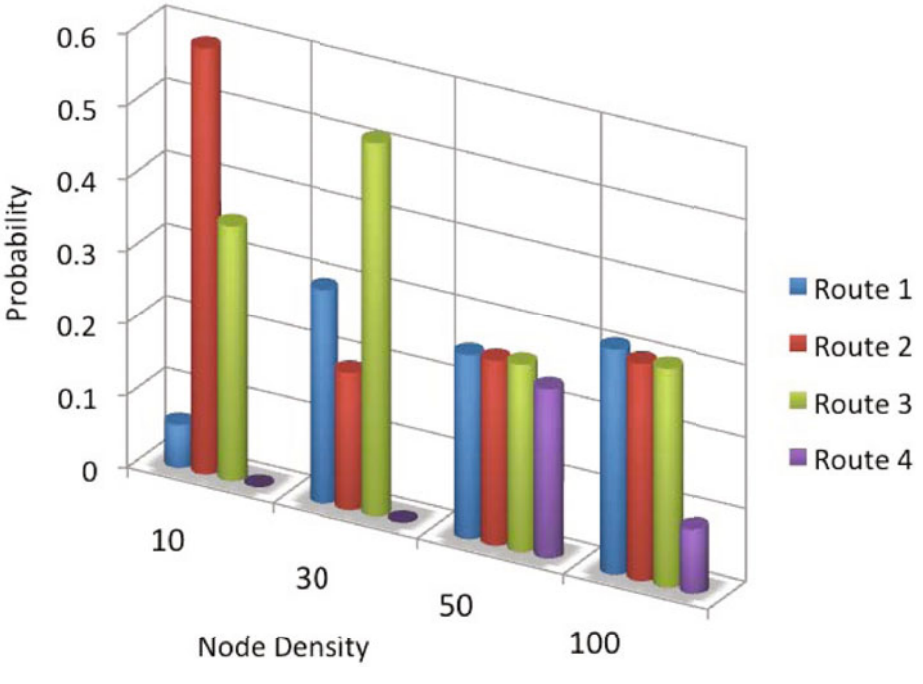


Fig. 2. Simulation Results

7 Conclusion

In this paper, we presented an algorithm for the offline reconstruction of vehicular traces of a target entity from fused RF-visual traces of that entity, but we assume that these traces exhibit occasional missing data, usually due to lack of surveillance resources. Our algorithm reconstructs the target's mobility traces by first identifying the locations of missing data and treats them as gaps. All the possible routes connecting these gaps are then identified. Based on a basic Bayesian inference approach and the driving behaviors of the target (obtained from the available traces), these routes are analyzed and the most probable one is selected. This process is repeated for all the gaps until the full (probabilistic) route of the target is reconstructed. We have validated the algorithm using simulations. Future work will, however, include validation using real vehicular traces, which allows capturing driving behavior more accurately.

References

1. Tan, M., Tian, G.L., Ng, K.W.: Bayesian Missing Data Problems: EM, Data Agumentation and Noniterative Computation. CRC Press, Boca Raton (2009)
2. Calbi, A., Marcenaro, L., Regazzoni, C.: Dynamic Scene Reconstruction for 3D Virtual Guidance. In: Gabrys, B., Howlett, R.J., Jain, L.C. (eds.) KES 2006. LNCS (LNAI), vol. 4252, pp. 179–186. Springer, Heidelberg (2006)
3. Conaire, C.O., Fogarty, K., Brennan, C., O'Connor, N.E.: User Localisation using Visual Sensing and RF Signal Strength. In: ImageSese (2008)
4. Barakatsoulas, S., Pfoser, D., Salas, R., Wenk, C.: On Map-Matching Vehicle Tracking Data. In: VLDB '05: Proceedings of the 31st international conference on Very large data bases, VLDB Endowment , pp. 853–864 (2005)
5. Al-Kuwari, S., Wolthusen, S.: A Survey of Forensic Localization and Tracking Mechanisms in Short-Range and Cellular Networks. In: Goel, S. (ed.) 1st International Conference on Digital Forensics & Cyber crime (ICDF2C), vol. 31, pp. 19–32 (2009)
6. Bay, H., Tuytelaars, T., Van Gool, L.: SURF: Speeded up robust features. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3951, pp. 404–417. Springer, Heidelberg (2006)
7. Tanenbaum, A.: Computer Networks. Pearson Education Ltd., London (2003)
8. Miyajima, C., Nishiwaki, Y., Ozawa, K., Wakita, T., Itou, K., Takeda, K., Itakura, F.: Driver Modeling Based on Driving Behavior and Its Evaluation in Driver Idenfication. Proeedings of the IEEE 95(2), 427–437 (2007)
9. Al-Kuwari, S., Wolthusen, S.: Forensics Tracking and Mobility Prediction in Vehicular Networks. In: Sixth Annual IFIP WG11.9 International Conference on Digital Forensics (2010)
10. Harri, J., Fiore, M., Fethi, F., Bonnet, C.: VanetMobiSim: Generating Realistic Mobility Patterns for VANETs. In: Proc. of the 3rd ACM International Workshop on Vehicular Ad Hoc Networks (VANET'06), Los Angeles, USA (2006)
11. Fiore, M., Harri, J., Filali, F., Bonnet, C.: Vehicular Mobility Simulation for VANETs. In: ANSS '07: Proceedings of the 40th Annual Simulation Symposium, pp. 301–309. IEEE Computer Society, Los Alamitos (2007)
12. Treiber, M., Hennecke, A., Helbing, D.: Congested Traffic States in Empirical Observations and Microscopic Simulations. Physical Review E 62(2), 1805–18024 (2000)
13. Information Sciences Institute (NS-2), http://www.isi.edu/nsnam/ns
14. Wahab, A., Quek, C., Tan, C.K., Takeda, K.: Driving Profile Modeling and Recognition Based on Soft Computing Approach. IEEE Transactions on Neural Networks 20(4), 563–582 (2009)

Throwing a MonkeyWrench into Web Attackers Plans

Armin Büscher[1,2], Michael Meier[1], and Ralf Benzmüller[2]

[1] Technische Universität Dortmund, Computer Science VI, Germany
{armin.buescher,michael.meier}@tu-dortmund.de
[2] G Data Software AG Bochum, Security Labs, Germany
{armin.buescher,ralf.benzmueller}@gdata.de

Abstract. Client-based attacks on internet users with malicious web pages represent a serious and rising threat. Internet Browsers with enabled active content technologies such as JavaScript are vulnerable to so-called drive-by downloads. Drive-by downloads are able to automatically infect a victim's system during a single visit of a crafted web page testing various vulnerabilities and installing e.g. malware files or illegal content without user interaction. In this paper we present MonkeyWrench, a low-interaction web-honeyclient allowing automatic identification of malicious web pages by performing static analysis of the HTML-objects in a web page as well as dynamic analysis of scripts by execution in an emulated browser environment. Using this hybrid approach MonkeyWrench overcomes shortcomings of existing low-interaction web-honeyclients in dealing with obfuscated JavaScript while outperforming high-interaction systems. Further MonkeyWrench is able to identify the exact vulnerability triggered by a malicious page and to extract payloads from within obfuscated scripts which are valuable information to security analysts and researchers. Results of an examination of several hundred thousand web pages demonstrate MonkeyWrench's ability to expose rising threats of the web, and to collect malware and JavaScript exploit samples.

Keywords: Web Security, Drive-by downloads, Active content-based attacks, Honeyclients.

1 Introduction

Recently the browser has gained popularity as an infection vector for the installation of malicious files. In contrast to attacks on vulnerable services the attacker does not have to find and scan remote systems. However the challenge of a successful web attack lies in attracting internet surfing users to visit the manipulated contents. So-called drive-by download attacks are able to automatically infect a victim's system during a single visit of a crafted web page testing various browser vulnerabilities and installing e.g. malware files or illegal content without user interaction. Figure 1 depicts the workflow of a typical web-based attack consisting of the following steps: (1) the attacker sets up a distribution site serving malicious content and malicious binaries (f.e. a trojan horse) that the attacker intends to spread; (2) the attacker compromises a web server and injects code in web pages integrating malicious contents from the distribution site or the attacker sets up a website with contents tempting specific users to visit; (3) the victim visits a manipulated “bait” page; (4) the bait page

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 28–39, 2010.

redirects the victim's browser through one or more steps to contents served by the distribution site; (5) the victim's browser requests the remote contents from the distribution site; (6) the distribution site replies with malicious content that is intended to exploit weaknesses of the victim's browser and/or installed plug-ins.

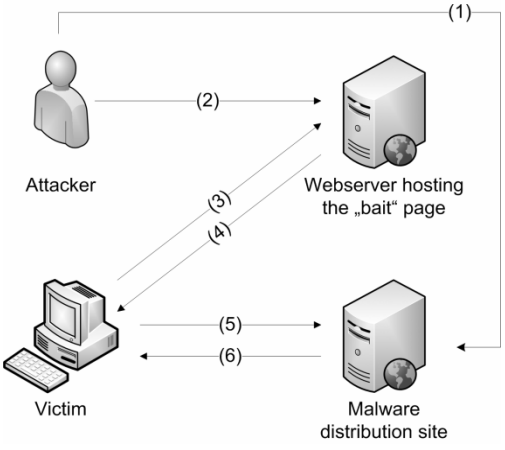


Fig. 1. Workflow of a web-based attack

Traditional security measures like packet filters and NAT do not provide protection against drive-by downloads since the connection is established by the client and allegedly does not pose as a threat. It is a complex and difficult task to establish suitable rules in intrusion detection systems to reliably discover web-based attacks since pattern matching is not applicable to detect obfuscated JavaScript that decrypts itself on the client system during runtime. Many popular websites rely on the clientside execution of script code to provide dynamic contents with enhanced features. Therefore the majority of internet users adopt the default settings of popular browsers to automatically permit JavaScript on every web page. Furthermore a lot of users surf the internet using systems with out-dated and unpatched browsers, operating systems or plug-in-technologies that enable attackers to exploit known vulnerabilities.

Web-honeyclients are used to detect the exploitation of browser vulnerabilities like for drive-by downloads of malware or phishing attacks. Thus the prevalence of web-based attacks can be evaluated and counter measures may be developed. Honeyclients are differentiated into high- and low-interaction systems. Whereas high-interaction systems represent a conventional computer system with real vulnerabilities, low-interaction systems emulate the behavior of a system (cf. [9]). High-interaction web-honeyclients monitor the file system, the system registry and the running processes for changes during a web page visit to heuristically detect malicious behavior. Since the systems are actually vulnerable, the high-interaction honeyclients primarily use virtualized operating systems in order to be easily reset to a predefined state. Resetting the system after each detection obviously limits its availability and performance. Systems following the high-interaction approach reliably detect attacks utilizing obfuscated JavaScript and other active content but need to be protected against malware trying to spread itself and due to the heuristics detection approach offer only a very limited view on the techniques used to obfuscate and carry out web-based attacks. Low-interaction

web-honeyclients check web pages by emulating the behavior of a web browser. Pages are typically checked against a predefined malicious behavior by statically analyzing the HTML-code and performing signature-based detection. Low-interaction web-honeyclients have an advantage in performance over their high-interaction counterparts due to their minor complexity and do not need to be reset after each detection. They however are typically unable to handle obfuscated JavaScript and are therefore prone to most web-based attacks including drive-by downloads.

To overcome the shortcomings of existing low-interaction web-honeyclients, MonkeyWrench follows a hybrid approach that executes and dynamically analyzes the behavior of active content to detect drive-by downloads and provide an analysis of the techniques used to obfuscate attacks and exploit vulnerabilities in a target system. The system has a lower computational overhead than high-interaction web-honeyclients and efficiently solves the problem of obfuscated JavaScript. High-interaction systems have issues detecting malicious sites carrying out attacks against certain combinations of browser version or browser plug-ins. Actually high-interaction architectures have to provide a vulnerable system for each of these combinations whereas MonkeyWrench is able to emulate multiple combinations by repeatedly executing JavaScript in adjusted browser environments when it detects scripts that request properties used to fingerprint the client system. Unlike high-interaction systems that only detect an anomalous behavior of the client system, MonkeyWrench is able to log the exact vulnerabilities that were attacked by a website and extract respective payloads.

Primary contributions of this paper are as follows: An overview on web-based attacks incorporating JavaScript is given in Section 2. We propose a hybrid analysis approach for detecting malicious websites trying to strike a balance between performance and detection rate and describe architecture and implementation of our MonkeyWrench system in Section 3. The system's detection and analysis abilities are demonstrated by evaluation results on the system's performance and results of an examination of several hundred thousand websites as well as an in-depth analysis of a malicious website in Section 4. Related work is discussed in Section 5 and we conclude in Section 6.

2 Web-Based Attacks Using JavaScript

JavaScript enables web designers to create web pages with dynamic content reacting to the user's interactions without reloading the page. Since the script code is downloaded and executed on a client system, attackers are able to go for local vulnerabilities. In addition JavaScript can be used to prepare web-based attacks (e.g. using heap spraying) and to obfuscate the exploit code to prevent static signature-based detection. The following browser vulnerabilities with their "Common Vulnerabilities and Exposures" CVE-ID [2] are subsequently referred to as V1-V6:

V1. Real Networks Real Player CVE-2008-1309: A buffer overflow in Real Player's ActiveX control allows the execution of arbitrary code.
V2. Yahoo! Music Jukebox CVE-2008-0623: A stack-based buffer overflow in the ActiveX control "YMP Datagrid" allows the execution of arbitrary code.
V3. Microsoft Access Snapshot Viewer CVE-2008-2463: A function of the ActiveX plug-in allows the download of an arbitrary file. The exploit may lead to the automatic execution of the downloaded file in a Startup folder.

V4. Microsoft Windows Media Encoder CVE-2008-3008: A stack-based overflow in the ActiveX profile manager component of Windows Media Encoder 9 allows the execution of arbitrary code.
V5. Ourgame GLWorld CVE-2008-0647: Multiple stack-based buffer overflows in the ActiveX control of the chat application Ourgame GLWorld allow remote attackers to execute arbitrary code.
V6. Microsoft Internet Explorer XML Parsing CVE-2008-4844: A vulnerability in Microsoft Internet Explorer allows remote attackers to execute arbitrary code via a crafted XML document.

Many of the commonly exploited vulnerabilities attack ActiveX controls, portable COM-objects used in Microsoft operating systems offering an interface to services complementing the browser's functionality. In this context ActiveX shall be deemed to be a security risk since calls to the controls are executed with the current user's rights. If an installed ActiveX control has a vulnerability like a buffer overflow, the client is exposed to web-based attacks utilizing active content like JavaScript or VBScript. JavaScript sample #1 was found during the experiments presented in Section 4 and aims at exploiting a vulnerability in Real Player's ActiveX-plug-in. The sample shows how attackers utilize JavaScript to customize their attacks to the victim's system at runtime. First of all a Real Player ActiveX-object is initialized (line 1). Afterwards several memory addresses required to successfully exploit the vulnerability are computed using the Real Player's version and the system language of the browser (line 2-17). Then the payload of the attack is assembled using these addresses, a padding and shellcode (line 18). Finally the vulnerable function of the ActiveX control is called with the prepared arguments (line 19). Note that the sample was shortened and essential parts of the script have been omitted to avoid publishing a working exploit in this paper. Section 3.4 takes up the exploitation of this vulnerability again to describe how the analysis modules of MonkeyWrench detect attacks.

JavaScript sample #1 (exploitation of *V1*, omitted details marked with *):

```
 1 real = new ActiveXObject("IERPCtl.IERPCtl.1");
 2 realversion = real.PlayerProperty("PRODUCTVERSION");
 3 Padding = "";
 4 JmpOver = unescape("%75%06%74%04");
 5 if(navigator.userLanguage.toLowerCase() == "en-us")
 6   ret = unescape("%4f%71%a4%60");
 7 else return;
 8 if(realversion.indexOf("6.0.14.") != -1) {
 9   for(i=0;i<10;i++)
10     Padding = Padding + JmpOver;
11   Padding = Padding + ret;
12 }
13 else return;
14 for(i=0;i<32*148;i++)
15   Padding += ******;
16 AdjESP = ******;
17 Shellcode = ******************;
18 PayLoad = Padding + AdjESP + Shellcode;
19 real.import("C:\\WINDOWS\\Media\\ding.wav",PayLoad,"",0,0);
```

3 MonkeyWrench

MonkeyWrench follows a hybrid approach to check web pages by statically analyzing the HTML-objects of a page and running JavaScript in an emulated browser environment for dynamic analysis. Therefore dynamic content can be examined and dynamic redirects can be followed without causing the overhead of a high-interaction web-honeyclient. As opposed to conventional honeyclients MonkeyWrench is able to analyze and identify the techniques used in web-based attacks by checking and logging all calls from JavaScripts to the emulated browser environment.

The MonkeyWrench system consists of two main components that are unified in a console-application written in C++: the client and the analysis engine. As an additional tool WrenchServer was developed as a graphical database user-interface used to visualize the results of an examination offering analysis reports and a source code view of malicious pages and scripts.

3.1 Client

The client represents MonkeyWrench's main program and contains the functionality to locally read or download a web page or a script and examine it using a static analysis and the included analysis engine for dynamic analysis of active content. Figure 2 depicts the workflow of MonkeyWrench's client working off a list of candidates.

First of all the candidate list crafted by the candidate creation component or manually entered by the user is loaded into memory. Afterwards the candidates are either checked one by one or assigned to threads by a taskpool if multi-threading is activated. The candidate's web pages are retrieved via the library libcurl [3] following the Hypertext Transfer Protocol (HTTP) in online mode or read from the filesystem when used in offline mode to analyze network dumps. MonkeyWrench uses the useragent string of Microsoft's Internet Explorer 7 for HTTP requests to avoid simple fingerprinting techniques in online mode. The web pages are subsequently preprocessed and the document tree is being traversed using the libTidy library. JavaScripts are forwarded to the analysis engine where they are executed in an emulated browser environment to enforce behavior based detection of malicious scripts.

Static and dynamic analysis results are managed as events within the client. Events occurring during examination of a candidate are stored for evaluation into a result database using the SQLite-database interface. These events represent interactions of the contents of a page or a script with the emulated browser. During static analysis the web page triggers so-called PageEvents, e.g. induced by an invisible inline frame. JSEvents are the effects of the execution of a script generated in the analysis engine to log calls to methods and properties of browser objects and ActiveX controls in the emulated browser. Every event inside of MonkeyWrench is classified into one of the following five security levels: *erroneous* - event was triggered by an erroneous call or object; *harmless* - event was classified harmless; *questionable* - event could be used to prepare an attack; *suspicious* - event indicates a possible malicious interaction; *dangerous* - event was triggered by interaction used to harm the emulated system. All

events raised in MonkeyWrench by the HTML-parser and the analysis engine are evaluated to determine the security level of the respective website, applying a set of rules. In addition to explicit attacks on the emulated vulnerability modules, MonkeyWrench detects suspicious activities like certain variants of code obfuscation, invisible iFrames and heap spraying techniques.

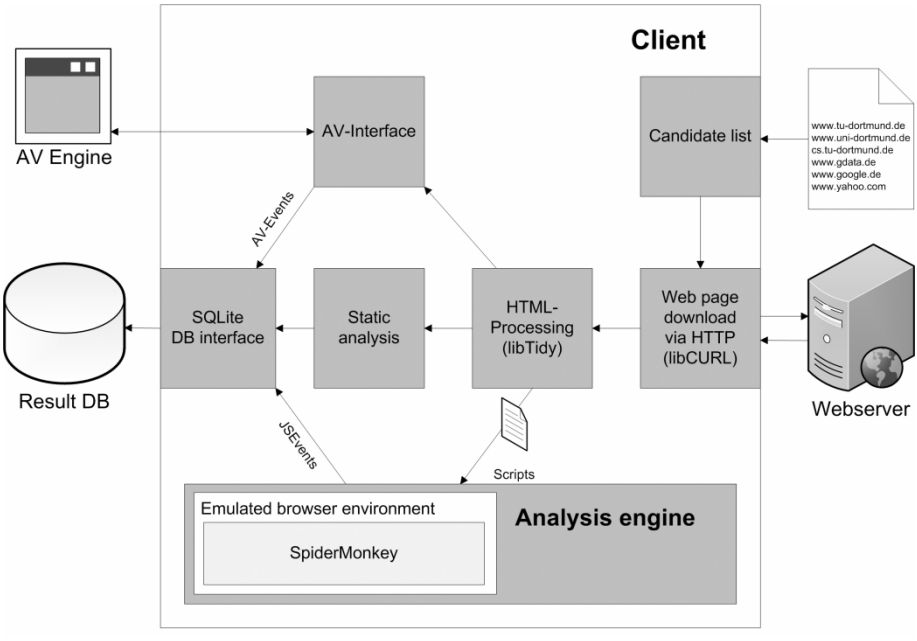


Fig. 2. Workflow of the MonkeyWrench client checking web pages

Static and dynamic analysis results are managed as events within the client. Events occurring during examination of a candidate are stored for evaluation into a result database using the SQLite-database interface. These events represent interactions of the contents of a page or a script with the emulated browser. During static analysis the web page triggers so-called PageEvents, e.g. induced by an invisible inline frame. JSEvents are the effects of the execution of a script generated in the analysis engine to log calls to methods and properties of browser objects and ActiveX controls in the emulated browser. Every event inside of MonkeyWrench is classified into one of the following five security levels: *erroneous* - event was triggered by an erroneous call or object; *harmless* - event was classified harmless; *questionable* - event could be used to prepare an attack; *suspicious* - event indicates a possible malicious interaction; *dangerous* - event was triggered by interaction used to harm the emulated system. All events raised in MonkeyWrench by the HTML-parser and the analysis engine are evaluated to determine the security level of the respective website, applying a set of rules. In addition to explicit attacks on the emulated vulnerability modules, MonkeyWrench detects suspicious activities like certain variants of code obfuscation, invisible iFrames and heap spraying techniques.

3.2 Dynamic Analysis Engine

The task of the analysis engine within MonkeyWrench is to detect active content executing web-based attacks and their preparation, while the scripts are executed in an emulated browser environment. The execution of active content is currently limited to JavaScript using the Mozilla's script engine SpiderMonkey [5]. Thanks to its open-source policy SpiderMonkey could be modified to suit the requirements of analyzing potentially malicious code.

The environment of a web page inside the browser is the Document Object Model (DOM), representing the interface between content and web browser. By reproducing the DOM, MonkeyWrench creates an emulated browser environment to control the calls of a script and enforce the behavior based detection in the client. Every call is checked heuristically for buffer overflows and the occurrence of shellcode to detect unknown and not yet emulated vulnerabilities. Every interaction with the emulated browser triggers an event with a certain security level that is forwarded to Monkey-Wrench's client. A drawback of this approach is that malicious scripts could try to detect MonkeyWrench by calling unimplemented methods to check the return values.

JavaScript sample #2 is part of a malware webkit found on a website during the experiments presented in Section 4. Its task is to deploy different attacks depending on the client's internet browser and installed plug-ins. In the first three lines the script tests the value of *navigator.userAgent* to decide whether or not the browser is Microsoft's Internet Explorer (MS IE) 7. In case that the returned string contains "msie 7" the script loads an iFrame containing an attack against the vulnerability V6. Then the script tries to instantiate the ActiveX-object of V5. Afterwards an iFrame with an attack against this plug-in's weakness is loaded if the browser successfully returns the object. Unlike high-interaction systems MonkeyWrench is able to detect the request for *navigator.useragent* and repeatedly execute the script. Thereby the system is able to react to simple script-side fingerprinting by offering different response values. The prototype of MonkeyWrench currently implements the useragent strings of the following combinations of operating system and internet browser: MS IE 7 on Windows XP, MS IE 6 on Windows XP, MS IE 8 on Windows Vista, Mozilla Firefox 3 on Windows Vista, Mozilla Firefox 2 on Windows XP, Google Chrome on Windows XP, Apple Safari on Windows XP and Opera on Windows XP.

Certainly other JavaScript techniques can be used to fingerprint the client's browser and system and therefore be used to effectively hide malicious activities from MonkeyWrench. One technique to detect MonkeyWrench is to execute certain methods revealing differences in the implementation of the JavaScript engine and the emulated browser environment.

The MonkeyWrench prototype emulates and dependably identifies several ActiveX-vulnerabilities including V1-V6. Furthermore MonkeyWrench can be simply upgraded with additional vulnerability modules. The ActiveX vulnerability modules of MonkeyWrench are triggered by a call to *new ActiveXObject()* or an <object>-tag including the class-id embedded in a web page. MonkeyWrench instantiates the respective module or a general module used to check all method invocations and property requests against arguments containing large strings or shellcode. As an example the class deployed by the vulnerability module "Real Player" (cf. JavaScript sample #1 in Section 2) contains the following methods/properties:

- The Property *Console* vulnerable to buffer overflow induced by setting.
- The Method *PlayerProperty* that returns a string containing the client's Real Player version. The module acts as the vulnerable version "6.0.14.544".
- The method *Import* vulnerable to a buffer overflow induced by a long string passed as the second argument of a call to the method.

During execution of a script the vulnerability modules analyze calls to implemented and unimplemented methods and properties. Alarm events are raised whenever a buffer overflow attempt or an argument containing shellcode is found.

JavaScript sample #2 (part of a malware webkit, shortened):

```
if (navigator.userAgent.toLowerCase()
    .indexOf("msie 7")==-1)
document.write("<iframe src=14.htm><\/iframe>");
try {
  var f;
  var gw=new ActiveXObject("GLIEDown.IEDown.1"); }
catch(f){}
finally {
  if(f!="[object Error]") {
     document.write("<iframe src=newlz.htm><\/iframe>");
  }  }
```

4 Evaluation

This section describes experiments performed to evaluate MonkeyWrench's performance and detection effectiveness in comparison to a high-interaction honeyclient. Furthermore we describe the insights on prevalent web-attacks that we gained during our analysis of web pages and we present an in-depth analysis of one of the malicious web pages. The experiments described in this section were performed from May 25th to June 6th 2009 on a 2.4 GHz Intel Core-2-Quad test system with 8 GB of RAM running Debian Linux "Lenny" connected to the internet with 2 MBit/s DSL. For comparative evaluation the high-interaction honeyclient CaptureHPC [1] version 2.51 was installed on the test system running a Windows XP SP2 client with Internet Explorer 6 using the default configuration.

4.1 Multi-threading Performance

As mentioned in Section 3, MonkeyWrench is able to check multiple pages concurrently using multithreading. The diagram in Fig. 3 shows how multithreading affects the average length of an experimental web page check on our test system during an examination of the top 1000 domains of the Alexa top list worldwide [11].

Since the DSL connection presumably acts as the restrictive factor in this experiment a reevaluation utilizing a better internet connection is considered. During the examination one site that tried to instantiate a spyware plug-in was found to be malicious by MonkeyWrench.

CaptureHPC finished the list of Alexa's top 1000 domains in 212 minutes, which equals an average check length of 12.72 seconds per page. No malicious pages were found and the virtual machine image was automatically set back 26 times during the process. CaptureHPC was using the default configuration of 20 concurrent Internet Explorer processes with a visitation time of 10 seconds after each page was loaded.

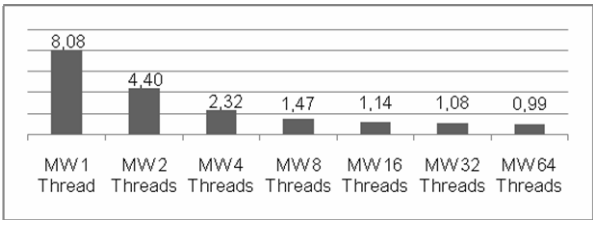


Fig. 3. Average length of a web page check in seconds during examination of Alexa Top 1000 domains world-wide using the MonkeyWrench prototype (average value of 3 runs)

4.2 Web Attacks

During the examination a total of 567.157 web pages were checked. The candidate list was created by querying Google's search engine with promising search terms and URLs that were reported by users. MonkeyWrench found 31.817 malicious web pages equaling a rate of about 5.6% and 19.317 inline frames invisible to the user pointing to malware distribution pages. During the examination the system saved approximately 3 GB of HTML, JavaScript (obfuscated and deobfuscated) and binaries including 2.114 unique (differing MD5 values) malicious executable samples.

All files generated by MonkeyWrench were scanned utilizing the G Data Linux antivirus engine. The scanner marked a total of 43.175 files as malicious. The bulk of the antivirus detections were triggered by files that were deobfuscated by the MonkeyWrench system. Therefore a HTTP scanner as utilized by many common antivirus solutions would not have detected these attacks since the attacks are dynamically decrypted in the browser. A noticeable amount of detections were triggered by signatures not targeting web-based exploit code but inline frames pointing to known (blacklisted) malware distribution domains.

Several large-scale attacks were identified using the result database of MonkeyWrench. Thereby several thousand infected pages were linked to the malware distribution servers used. The largest attack found by MonkeyWrench was the infection of 887 web pages using SQL injection. The actual attack was carried out by a malware server having a DNS name with the top level domain of China using 33 different domains. A total of 598 malicious pages were trying to lure the user into installing a fake media codec being a trojan horse. An attack against a recent Adobe Acrobat Reader vulnerability (CVE-2009-0927 [2]) infected 391 domains hosting adult content. Another infection, called Gumblar/Martuz used websites that were taken over by attackers through stolen FTP credentials and vulnerable web applications. MonkeyWrench detected 181 infections of Gumblar/Martuz.

The majority of malicious pages tried to exploit more than one vulnerability. Figure 4 depicts the distribution of the attacks detected by MonkeyWrench's ActiveX

vulnerability modules. The numerous exploitation of the Microsoft Access Snapshot Viewer vulnerability may be explained by its ease of use: The attacker just has to use a specific regular functionality of the ActiveX control to download an arbitrary remote file to the local filesystem and to overwrite system files.

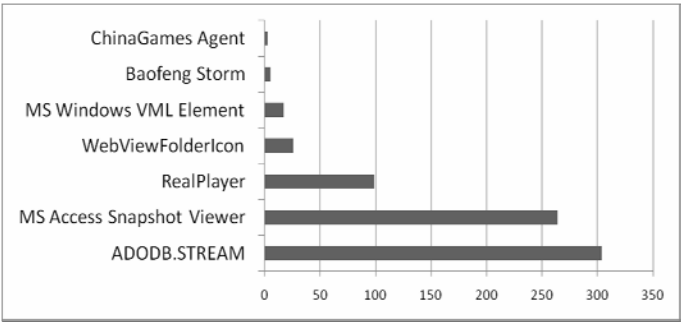


Fig. 4. Detected exploitation of vulnerabilities

A fraction of 900 web pages were reexamined to compare the results and in particular the detection rates of MonkeyWrench and CaptureHPC. Of the 900 pages checked MonkeyWrench found two pages to be malicious while CaptureHPC marked all pages as benign. The results of MonkeyWrench revealed that the malicious websites both attacked vulnerability V1. Therefore the virtual machine image used in CaptureHPC was equipped with a vulnerable version of “Real Player”. In a subsequent run CaptureHPC detected the malicious pages previously found by MonkeyWrench but had no further findings.

4.3 In-Depth Analysis of Web-Based Attacks

In the following we analyze the techniques of one particular malicious page found during our examination and exemplify MonkeyWrench's ability to collect samples of partly unknown malware and JavaScript exploit code. The page offers an overview of the server status in the online role-playing game “World of Warcraft” to users. The market leader in online role-playing games “World of Warcraft” had over 11 million subscribers at the end of 2008 with a sales volume of about one billion dollars. It is uncertain whether the website's server was compromised by an attacker or deliberately prepared by the operator to lure users into visiting a malicious page. Embedded inline frames pointing to a malicious distribution site were repeatedly found during examination of the page. The inline frame redirected to content hosted by “pagead2.googlesyndicastion.com”. This domain resembles the URL of Google's advertisement application AdSense “googlesyndication.com”. Presumably the attacker tries to obfuscate the injected inline frame so that it is ignored by web admins during the manual inspection of the page's source code. The malicious content hosted by the distribution site tries to exploit several browser- and plug-in-vulnerabilities namely V1, V3 and V6. A JavaScript first checks the name and version of the user's browser and accordingly reloads adjusted variants of the Internet Explorer exploit.

Afterwards the system is tested for installed plug-ins and the corresponding exploits are loaded into the browser. During the examination of the page, MonkeyWrench collected a malware sample that ought to be installed by the distribution site. The G Data anti-virus-scanner used in the test system did not identify the sample as malware. Therefore the sample was uploaded to the web service Virustotal.com and scanned using anti-virus-scanners of 39 vendors. Only three engines identified the sample as malware: Prevx1: Malicious Software; Rising: Trojan.PSW.Win32.WoWar.ayg; SecureWeb-Gateway: Win32.LooksLike.Rootkit.gen. Prevx1 and SecureWeb-Gateway seemed to identify the sample using generic signatures whereas Rising could identify the sample more precisely. A description on the behavior of "Trojan-.PSW.Win32.WoWar.ayg" could not be found, thus it was executed in the sandbox application CWSandbox [12] to find out how it would affect a user's system. After installation the malware would copy itself to the system folder and register as a system service to automatically run at startup. In a manual comparison of the malware's behavior and characteristics of known malware samples, the sample could be assigned to the malware family "Trojan-GameThief.Win32.OnLineGames". The malware is a polymorphic trojan horse thus having a different MD5-checksum than other samples. Trojan horses of this family are used to record the user's keyboard input and monitor running processes. Once the user of an infected system starts "World of Warcraft" or other online games, the malware logs the account information and sends it to a remote server via HTTP. Therefore the malware used by the attacker is directly connected to its distribution since almost only customers interested in "World of Warcraft" are visiting the compromised website.

5 Related Work

Provos and Holz [9] provide a broad introduction to the subject of honeypots. There are several ongoing projects engaging in the development of web-honeyclients. Kathy Wang released an open-source honeyclient in 2005 whose development is nowadays coordinated by the MITRE Honeyclient Project [4]. It consisted of two Perl-scripts managing an instance of Microsoft's Internet Explorer and monitoring the system while web pages are checked. Provos et al. [8] conducted a comprehensive study of malware on the web by using honeyclients to check the pages of Google's web repository. Microsoft uses Strider HoneyMonkeys [10], high-interaction web-honeyclients based on virtualized Windows operating systems, to detect the abuse of previously unknown vulnerabilities. CaptureHPC [1,7] is a high-interaction web-honeyclient released as open-source software by Seifert et al., those who earlier published their work on the HoneyC system [6] following a low-interaction approach. While effective, existing honeyclients are limited either in run-time performance due to their high-interaction approach or in detection performance due to their inability to dynamically analyze JavaScripts. In contrast to that, our honeyclient MonkeyWrench dynamically analyzes JavaScript in an emulated browser environment while avoiding run-time performance overheads by realizing a low-interaction approach.

6 Conclusion and Future Work

The evaluations in this paper demonstrate the abilities of MonkeyWrench as an efficient tool to examine the techniques of attackers using malicious web pages and to collect previously unknown malware and JavaScript exploit samples. Besides detecting web-based attacks based on events triggered by the vulnerability modules, MonkeyWrench uses advanced techniques like shellcode and heap spraying detection to spot previously unknown attacks. Compared to the high-interaction honeyclient CaptureHPC, MonkeyWrench showed a considerable advantage in performance during our experiments. MonkeyWrench's emulated browser environment allows a researcher to identify and display exactly which vulnerabilities are attacked by a malicious website and how the attack was prepared. Certainly the approach has some drawbacks naturally caused by the emulation of a system like attackers who could avoid being detected performing either server- or client-side fingerprinting.

The in-depth examination of websites proves that the attackers adjust their attacks to capitalize on booming trends within the internet community. Users are lured into visiting web pages that contain promising content though attacking the browser to perform drive-by downloads of customized Trojan horses which typically steal the user's account information and password that is afterwards resold by the attacker.

The web-honeyclient MonkeyWrench represents a prototype that features some aspects that need further development to achieve the approach's full potential. Thus the execution of active content besides JavaScript like VBScript or Adobe Flash including Actionscript is considered.

References

1. The Honeynet Project CaptureHPC, `https://projects.honeynet.org/capture-hpc/`
2. Common Vulnerabilities and Exposures (CVE), `http://cve.mitre.org/`
3. libcurl, `http://curl.haxx.se/libcurl/`
4. MITRE Honeyclient Project, `http://www.honeyclient.org/trac`
5. Mozilla SpiderMonkey (JavaScript-C) Engine, `http://www.mozilla.org/js/spidermonkey/`
6. Seifert, C., Welch, I., Komisarczuk, P.: HoneyC - The Low-Interaction Client Honeypot. In: Proc. of the 2007 NZCSRCS. Waikato University, Hamilton (April 2007)
7. Seifert, C., Steenson, R., Holz, T., Yuan, B., Davis, M.A.: Know Your Enemy: Malicious Web Servers (August 2007), `http://www.honeynet.org/papers/mws/`
8. Provos, N., Mavrommatis, P., Rajab, M., Monrose, F.: All Your iFrames Point to Us. In: Proc. of the 17th USENIX Security Symposium (July 2008)
9. Provos, N., Holz, T.: Virtual Honeypots: From Botnet Tracking to Intrusion Detection. Addison-Wesley Professional, Reading (2007)
10. Wang, Y., Beck, D., Jiang, X., Roussev, R., Verbowski, C., Chen, S., King, S.: Automated Web Patrol with Strider HoneyMonkeys. In: Proc. of NDSS 2006 (2006)
11. Alexa the Web Information Company - Top Sites, `http://www.alexa.com/topsites`
12. CWSandbox – Automated Malware Analysis, `http://www.cwsandbox.org/`

Security in OpenSocial-Instrumented Social Networking Services

Matthias Häsel[1] and Luigi Lo Iacono[2,⋆]

[1] XING AG, Hamburg, Germany
matthias.haesel@xing.com
[2] Europäische Fachhochschule (EUFH), Brühl, Germany
l.lo_iacono@eufh.de

Abstract. Securing social networking services is challenging and becomes even more complex when third-party applications are able to access user data. Still, adequate security and privacy solutions are imperative in order to build and maintain trust in such extensible social platforms. This paper discusses security issues in the context of OpenSocial-instrumented social networking services. It shows that the OpenSocial specification is far from being comprehensive in respect to security. Resulting weaknesses and shortcomings are emphasized and discussed. Finally, the paper attempts to fill these gaps by proposing extensions to the OpenSocial specification and recommendations for social networks that implement OpenSocial.

1 Introduction

Social networking services (SNS) such as Facebook, mixi or XING aim at building online communities of people who share interests and/or activities. They are technically accomplished through networking software that maps a social graph [1]. Most SNS are Web-based and build on proprietary solutions and data formats. An emerging trend is that SNS can be enriched by third-party applications that access their social graph. Interoperability is therefore an issue if these applications should reach more than a single SNS only. OpenSocial is a set of programming interfaces for developing applications that are interoperable within the context of different SNS [2].

When third-party applications are able to access user data, adequate security and privacy solutions are imperative. This paper provides an analysis of the security mechanisms integrated into OpenSocial and shows that protection is focused on data exchange between SNS and third-party applications only with protection mechanisms related to the integrity and authenticity of requests. The paper wants to raise awareness for security issues in an OpenSocial context. It gives recommendations on how to avoid flaws in implementing OpenSocial and integrating applications into an SNS by making use of standard security mechanisms which are not included in the OpenSocial specification. Finally, the paper

⋆ This work was performed while Luigi Lo Iacono was with NEC Laboratories Europe, NEC Europe Ltd.

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 40–52, 2010.

considers more advanced security techniques such as making the access to sensitive social data transparent to the user and integrating OpenSocial applications into a platform's access control system.

2 OpenSocial-Instrumented Social Networking Services

OpenSocial provides a means to extend or enrich SNS with third-party applications [2]. Using standard Web technologies, developers can create applications that run on SNS that have implemented OpenSocial. Until it was made public in November 2007, OpenSocial was driven primarily by Google, but is now managed by the non-profit OpenSocial Foundation. The Foundation consists of representatives of all major enterprises active in the social networking domain (except Facebook, which has developed a proprietary platform)[1] and specifies the APIs required for enhancing an SNS with third-party applications. The list of SNS which support OpenSocial (referred to as "containers" in OpenSocial jargon), can be accessed from the OpenSocial Website.[2] Prominent examples include iGoogle, LinkedIn, mixi, MySpace, orkut, Yahoo!, and XING.

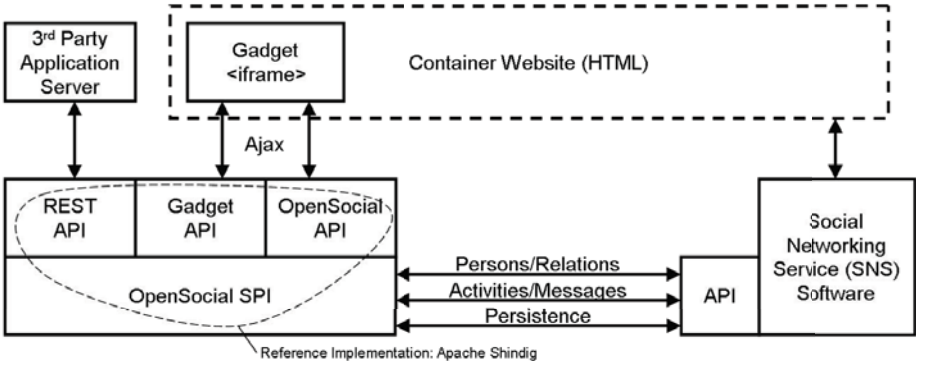


Fig. 1. OpenSocial Reference Architecture

The OpenSocial reference architecture is shown in Figure 1 and gives an overview of the involved technologies and components, as well as the relations and interactions between these components. OpenSocial applications are commonly based on the Gadget architecture[3] originally developed by Google, which has been expanded upon by interfaces enabling the access to the social data within the context of any given container. Gadgets are XML documents containing HTML and JavaScript code along with metadata. The XML specification of a Gadget is rendered by the container and integrated into its own website. Communication between the Gadget and the container operates in such instances

[1] http://developers.facebook.com/
[2] http://www.opensocial.org/
[3] http://code.google.com/intl/en/apis/gadgets/index.html

via standardized Ajax requests [3], which are defined in the `opensocial.*` and `gadgets.*` namespaces of the JavaScript-based OpenSocial [4] and Gadget [5] API respectively.

2.1 OpenSocial Compliance and Basic Services

Containers must implement all of the components shown in Figure 1 to be compliant with OpenSocial. Apache Shindig[4] is a reference implementation of the entire OpenSocial stack that operators of social network sites can refer back to. Shindig is open source and developed in both Java and PHP. The interconnection between the existing social networking software and Shindig takes place via the so-called OpenSocial Service Provider Interface (SPI), whose classes are extended in such a way that they access the networking software.

OpenSocial applications and the SNS software exchange three kinds of data which correspond to the three basic services that OpenSocial-instrumented platforms offer to their applications:

1. **People/Relationships:** The OpenSocial API enables direct access to the social graph of the container in the form of individual user objects. Two user instances that are made directly available to an application are the *viewer* and the *owner*. The viewer object refers to the current user, while the owner object is associated with the user in whose profile context the application is executed. Moreover, social applications have access to the targeted connections within the social network to be able to support the exchange of information and the interaction between users.
2. **Activities/Notifications:** Activities represent interactions between the user and the Gadget. Containers usually display a user's activities to all friends of that user within their update feeds. With Notifications, applications can make use of a container's messaging system by passing one-to-one messages to the container that will be sent on behalf of the current user.
3. **Persistence:** Data is often created during the course of user interaction that has to be saved on a persistent basis. For this purpose OpenSocial includes a persistence layer, allowing developers to store simple name/value pairs for each application or for each application and user. The container is responsible for the actual implementation of this persistence layer, which remains hidden from the social application developer.

Shindig makes available these services in its implementation of the OpenSocial JavaScript API. It also implements the Gadget JavaScript API which is used for security, communication and the user interface, and includes a Gadget rendering server that transforms the XML specification of an application into JavaScript and HTML code and makes this available via HTTP. Finally, with the OpenSocial gateway server component, Shindig provides an implementation of an additional server-side, RESTful API.

[4] `http://incubator.apache.org/shindig/`

2.2 OpenSocial Application Types

Three main types of OpenSocial applications can be destinguished (see Figure 2): Social mashups, social applications, and external applications.

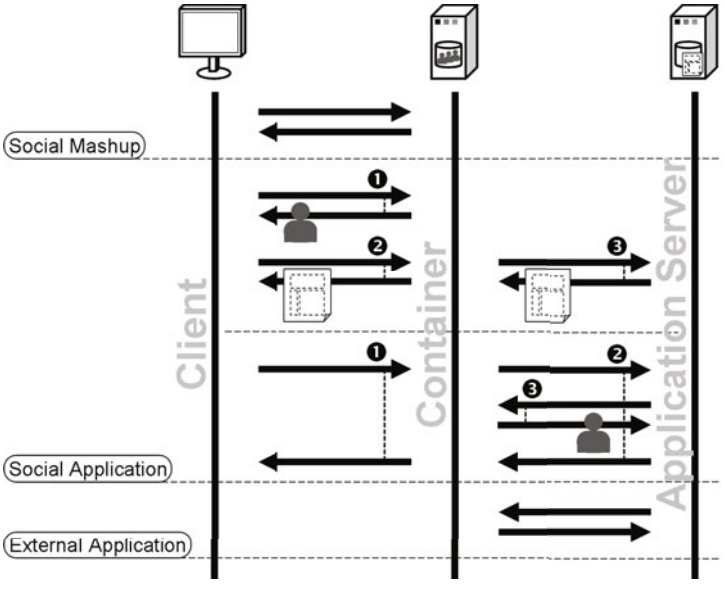


Fig. 2. OpenSocial Application Types and Associated Communication Patterns

A *social mashup* is a lightweight OpenSocial Gadget which runs inside a container. Since such applications do not rely on external third-party servers, they are extremely scalable, and the social data does not leave the protection sphere of the container. However, social mashups are limited in terms of data storage and/or processing capabilities and do not support interaction scenarios without user intervention. An example is an enhanced visualization mechanism to give users a view of all direct connections of the own social graph. A social mashup is typically created using client-side technologies only, such as HTML, JavaScript, and CSS.

A *social application*, while also being a Gadget, relies on an external third-party server for processing, storing and/or rendering data. Social applications use server-side technologies such as PHP, Python, Java, Perl, .NET, or Ruby on Rails. They provide advanced functionality and support interaction scenarios without user intervention. This comes, however, with the cost of being less scalable, and may require the processing and storage of social data by third parties outside of the protection and control sphere of the container. As an example, consider an application that allows the user to create polls, letting their friends within the container vote for different alternatives. To enable the user to choose which of their contacts should participate in the poll, the application

would retrieve the owner and his/her friends via the OpenSocial API (step 1 in the upper part of Figure 2). In the background, the user data is requested from the container's networking software via the appropriate SPI call. The application would then send the poll data and a list of the participants' user IDs to its server using a `gadgets.io.makeRequest()` call to the Gadget API (step 2), with the container proxying the data to the application server (step 3). Subsequently, the application would send out a notification to each invited friend via the OpenSocial API, which could be translated by the container into site messages containing a link to the poll. Participants clicking on that link would be identified by requesting the owner object (again, step 1), and would then be asked to vote. The application would pass the vote to its application server using another `makeRequest()` call (steps 2 and 3). Note that in this realisation, the application server does not process personal data. For the application's logic to work, it is sufficient to associate in the application server the container's user identifiers with the polls and votes created.

An alternative version of the polls example is depicted in lower part of Figure 2. As a first step, instead of utilizing the JavaScript API, the Gadget sets off a proxied `makeRequest()` call, transferring the owner's user identifier to the application server (steps 1 and 2). The application server requests the container's RESTful API (step 3) which responds with the owner and/or a list of their friends. The people data is combined with the application data, producing fragments of HTML and JavaScript that form the response to the container's request started in step 2 and the gadget's request started in step 1, respectively. Note that in this scenario, the application server processes (and may store) personal data retrieved via the RESTful API. In fact, the polls application does not require personal data to be transfered to the application server. However, there are use cases that cannot be realized with client-side APIs only. As an example, consider that polls would be limited to a certain period of time, and the application would be required to send a message (with a link to the results) to all participants after expiry. As the creator would not be interacting with the application at the time of expiry, the message needs to be triggered by the application server, rendering a client-side API useless.

Finally, an *external application* such as a website or mobile application runs outside a container, but still consumes social data through the RESTful API. Users can grant access to their personal data without needing to add the application on the SNS Website. Note that external applications are not necessarily based on the Gadget architecture. An example is an application that feeds or synchronizes the contacts database of a mobile phone with the data available from the social graph. This application type grants the most flexibility—almost all languages and platforms can take advantage of the social data—but also includes the highest risks of unauthorized access.

2.3 OpenSocial Application Lifecycle

From a developer's perspective (see left side of Figure 3), deploying a social mashup or application is equivalent to passing its XML specification to a

container. Most containers require that developers submit their application to be reviewed before it is made available in their directory. Containers have different processes for granting developers access to their sandbox environments and reviewing applications. On Orkut, for example, developers can sign up for a developer sandbox and submit their application online, while on XING developers have to hand in a product concept before they are granted access to the sandbox.

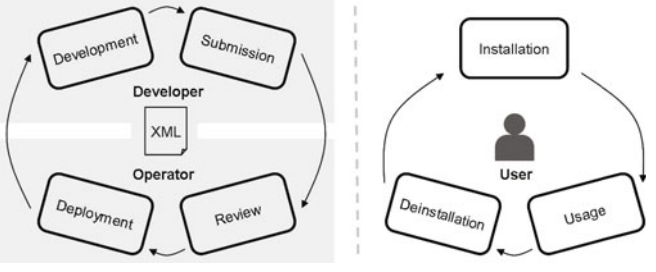


Fig. 3. OpenSocial Application Lifecycle

From a user's perspective (see right side of Figure 3), first-time usage of an OpenSocial application usually involves some kind of installation process whereby the user explicitly permits the application to access the data in their personal user profile. Events such as installation or deinstallation are desirable yet difficult for developers to track. The OpenSocial specification therefore allows developers to specify URLs that the container will POST event data to when these events are triggered. Besides installation and deinstallation, such events include rate limiting, directory listing changes, and blacklist/whitelist notifications. By tracking the data sent to the specified URL, application providers can accurately track the number of installs, remove database entries upon uninstall, and get automatic notifications if their application exceeds a quota or is marked as spam by the container.

3 Analysis of the OpenSocial Security Specifications

Security is a crucial aspect in SNS due to the sensitive and personal nature of the social data exchanged via these platforms. In the context of OpenSocial-instrumented SNS, security becomes even more important because third-party applications (possibly) access the social data. This section will introduce and analyse the built-in security mechanisms in the OpenSocial specification.

The focus of this analysis is on OpenSocial-specific issues. More general security problems related to Rich Internet Applications (RIA) and Web applications in general such as Cross-Site Scripting (XSS) or issues related to the execution of client-side JavaScript code are not included. Such aspects are not unique to OpenSocial and subject of other initiatives such as the Open Web Application Security Project (OWASP) [6].

3.1 Focus of the OpenSocial Specification Regarding Security

Taking the application types and their communication patterns introduced in Section 2 into account, it becomes clear that OpenSocial primarily introduces the communication path between the container and the third-party application server. The communication link between the container and the client is the SNS proprietary channel. Security considerations within the OpenSocial specification are therefore not targeted to the SNS proprietary link as it is seen as a container-specific implementation detail and assumed to be protected by appropriate policies and means. In the case of a social mashup or application e.g., the content of the Gadget XML—which contains HTML and JavaScript code—is rendered into the container's Website. To prevent DOM manipulations, the Gadget is usually embedded into an `<iframe>` served from an alternate domain.

The container is involved in all communications and in some cases it is mediating from the client-side application to the application server. Thus, the container plays a decisive role in terms of security and privacy especially in the presence of external application servers and should therefore carefully inspect all communications and strictly enforce its policies. This is manifested in the OpenSocial specification which recommends that the container should always validate the transferred parameters before passing them to the application server.

3.2 Message Integrity and Authentication

The communication between the container and the application server will most commonly include at least the IDs of the users currently interacting with an OpenSocial application on the container (see Section 2). Since `makeRequest()` calls are just JavaScript transmitted via HTTP GET or POST, it is possible for any user to create `makeRequest()` calls with whatever arguments they wish [7].

To protect IDs from manipulations by malicious users and to ensure that parameters truly originate from the container, OpenSocial contains a method to communicate IDs to the application server in a verifiable manner by using the OAuth API authorization protocol [8]. More specifically, requests are signed based upon OAuth's parameter signing mechanism. Through the method of parameter signing, it is possible for an application to request the container to send IDs along with a digital signature that allows third-party servers to verify that the parameters passed are legitimate. Note, however, that the response flow from the application server back to the client-side application is not protected by any OpenSocial-specific security mechanisms.

When a request is signed by the container, it adds additional data before forwarding it to the remote application server. This data contains information such as the IDs of the application making the request, the owner and viewer, and information that the application server can use to verify that the information added was not tampered with since the container sent the request. Two signature algorithms are specified: the secret-key based HMAC-SHA1 [9] and the public-key based RSA-SHA1 [9] method. To generate the signature, a so-called Signature Base String is generated first, which is a consistent reproducible concatenation of the request elements into a single string. The parameters contained

in the POST or GET request (`oauth_signature` excluded) are ordered and concatenated into a normalized string which finally forms the input to the digital signature algorithm. An example for the required steps to generate the signature base string and the signature is shown in Figure 4.

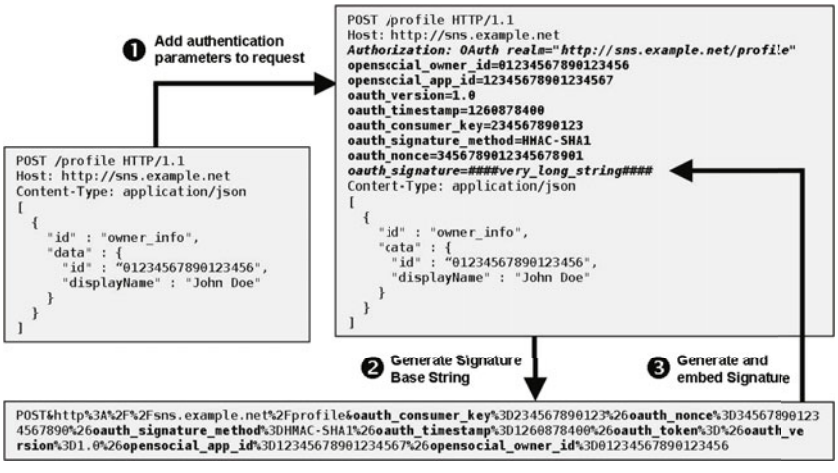


Fig. 4. Signature Generation based on the OAuth Parameter Signing Mechanism

Note that the OAuth parameter signing algorithm is applied to the HTTP parameters, but not to the application data (here JSON-encoded) included in the POST as depicted in Figure 4. Assume that the owner ID in the application data is altered by the user and that the container sets the owner ID to the HTTP parameters according to some context information established for the session with the user. If the container in this scenario does not cross-check whether the same owner ID appears in the application data as the one stored in the HTTP parameters, the parameter signing might get meaningless. If this happens, the owner ID is sent twice to the application server (once signed in the HTTP parameters and once unsigned in the application data). In this case the developer needs to compare these IDs carefully in his verification code. If this is not performed and the application server code verifies the signed parameters and executes the business logic based on the application data without cross-checking the consistency of IDs, the forgery of IDs and henceforth the unauthorized retrieval of social data is still feasible. Such an attack is very similar to the XML Signature Wrapping attack which was first described by Fournet at the DIMACS workshop in 2005 [10] and first published by McIntosh and Austel at SWS workshop in 2005 [11]. This threat renders a large set of applications that rely on the use of XML Signature [12] as a security enforcement technology critically insecure in cases in which the validation procedure does not carefully perform other security-related checks in addition to the pure verification of the digital

signature. The real-world impact of this threat was first shown by Gruschka and Lo Iacono at the ICWS conference in 2009 [13] when they described a similar vulnerability in the Amazon EC2 Cloud services.

3.3 Message Confidentiality

In order to protect the data flow against unauthorized access during the transmission and to maintain it confidential in the presence of passive attacks performed by eavesdroppers, communication needs to be encrypted. As in the case of the integration and authentication of requests, container-specific mechanisms are deployed for the communication between the client-side application and the container. For the communication between the container and the application server the OpenSocial specification does not state how to protect messages against eavesdropping, not even in terms of guidelines or recommendations. It is up to the container to define, implement and enforce an appropriate security policy.

3.4 Identity Management and Access Control

To ensure that the requestor identity cannot be spoofed in a client initiated data flow, OpenSocial containers usually expect a short lived security token that is made up by the owner ID, viewer ID, and application ID, among others. As far as a rendered application is concerned, the token should be entirely opaque; it should only be interpretable by the container's Gadget renderer and APIs. However, the OpenSocial specification itself does not state any details on how this token should be encoded. Neither does it give a recommendation on the token lifetime or its encryption.

Similarly, security tokens can be applied in an application server initiated data flow. In this scenario, the client retrieves the token via the JavaScript API and sends it to the application server via a proxied `makeRequest()` call. The application server using the security token can then make calls to the container's RESTful API, passing the security token as a URL parameter or within the HTTP request header. Again, OpenSocial does not provide any recommendations on identity management in a RESTful scenario. E.g., containers could use OAuth to validate whether a request has been initiated by a certain application server. Note that in such as scenario, the application server is able to arbitrarily set the identity of the user for whom the data is requested.

4 Recommended Security Improvements and Extensions

This section takes up the identified security issues of OpenSocial and provides suggestions on how to improve the current situation. The following proposals include both enhancements to the OpenSocial specification and recommendations to SNS that implement OpenSocial.

4.1 Application Type Selection

In Section 2.2 and 3.1 it became clear that social mashups have advantages in respect to security and privacy in comparison to the two other application types, since no social data is transmitted to an external server, but the personal data remains inside the protection sphere of the container. Thus, it is recommended that the container as well as the user should carefully examine the underlying communication pattern of each application before deciding to integrate or make use of it, respectively. For the container, this means that an acceptable procedure needs to be in place which controls and governs the integration of applications, i.e., the container needs to ensure during the review phase that an OpenSocial applications does not access more user data than required in order to protect its users against unauthorized data gathering.

4.2 Data Minimisation and Pseudonymisation

An important aspect SNS operators and users should check before deploying or installing an application respectively, is the kind of social data that is going to be used by the application outside the container's premises (if any). It needs to be controlled and ensured that the released data items are the minimum required to perform the task. Minimising the data amount exchanged with external entities is a measure to reduce the risk of unauthorized data gathering and data aggregation. This should be included in the specifications as a recommendation to make SNS operators aware of this risk.

To further increase privacy protection, the following additional aspect must be considered by the OpenSocial specification and the container. In most cases, applications require some form of ID in order to perform their tasks (see the polls example in Section 2.2). The OpenSocial specification does not define how this ID should look like. It is therefore possible that a container simply uses its internal user IDs, opening the door for coalition or multi-application attacks in which a single entity (by using a set of applications or a set of entities forming a coalition) will be able to correlate the obtained data using these IDs. To prevent such kind of attacks and to protect the users' privacy more effectively, the specification should recommend not to use internal IDs since these are personal identifiable information, but to replace them with application-specific pseudonyms instead.

4.3 Application Integration and Code Signing

As introduced in Section 3.1, `<iframe>` elements are a common means to integrate social mashups and applications into the container's Website without allowing client side scripts to perform attacks. Since this is not explicitly stated in the specification, an according recommendation should be added to raise the awareness and point SNS operators to additional resources on this topic.

Another issue is that the JavaScript code is often not stored at the container, but loaded from the Web. The JavaScript code within the Gadget XML specification is commonly reduced to a URL referencing the actual source. This renders

the review process of the SNS operator meaningless if the application code can easily be altered after it has been reviewed and added to the container. The adoption of code signing technology are required here, so that the source code can be verified by the container and in case of a dispute, the developer can not deny having released the code providing the necessary evidence for liability cases.

4.4 Communication Security

As discussed before, the specified means to protect the communication in an OpenSocial setting are limited to parameter signing of HTTP requests. No means to protect the HTTP responses are included, which might open doors for new attacks.

Moreover, OAuth targets security service integrity and authentication, but does not provide a guideline how to protect the confidentiality of user data. Since communication between the client-side application and the container is protected by container-specific means, OpenSocial does not interfere here. Still, for communication between the container and the application server, the specification should provide a guideline how to encrypt data. This is not only required to raise the awareness in confidentiality issues among SNS operators and application developers, but also in order to reduce the variety of different approaches that will be taken by distinct containers, which is contradicting one of the goals of OpenSocial, to create a write-once-run-everywhere (or at least learn-once-write-anywhere) environment since application developers need to adapt to container-specific demands.

The OpenSocial specification must be enhanced to additionally sign responses to enable the container to verify integrity and authenticity. Further means are out of scope of the OpenSocial specification, but still required in order to reach a certain level of security as well as compatibility among containers. Thus, the following recommendations should be carefully considered by SNS operators:

- Use HTTPS for Gadget rendering, Gadget JavScript API, OpenSocial JavaScript API, and RESTful API.
- Make HTTPS and signed messages (including event messages) mandatory for application providers' APIs (especially for social and external applications).
- Compare IDs in the signed parameters with the ones used in the unsigned application data and advise application providers to so as well.
- Make sure that security tokens are implemented in a safe way.

Some of these recommendations are out of the control sphere of the container. Still, containers have means to check and enforce such rules by analyzing the Gadget's XML specification.

4.5 Access Control and Delegation

Access to social data by external entities via the RESTful API is an aspect to be carefully considered. Unfortunately, the OpenSocial specification does not

give much guidance here. Authentication of external entities such as application servers or mobile applications is handled by mechanisms established by the container and initiated during the application deployment process. Further aspects should be defined within the specification to form a basis for effective and cross-platform access control systems. This includes mechanisms to check whether a user granted access to his/her personal data to a specific application. The specification should recommend that in any case, a container should make personal data only available to external entities of such users who have explicitly provided their consent (opt-in).

For more fine-grained access policies that limit the access to particular data items, appropriate mechanisms still need to be defined and specified. An initial approach for a more fine-grain access to user data by OpenSocial applications has been developed by StudiVZ.[5] The basic idea is that the user can pass "configurable" user profiles (so-called ID cards) to the application and by this means control the disclosure of their profile data (also a means to minimize the exposed data as stated in Section 4.2). Although this approach gives the user a maximum of control, it increases complexity for the user and may decrease the value provided by an application or might even effect its functioning due to the lack of relevant data. Practice will show how this approach will be accepted by users and application providers.

Delegation of access rights to external entities is another point which requires a more indepth analysis and appropriate specification. In case of external applications, standard three-legged OAuth mechanisms can be used [8]. In cases where an external entity needs to access social data without a user interaction, the definition of delegation tokens which can be used in conjunction with the two-legged OAuth protocol is still an open issue.

4.6 Transparency Enhancements

Crucial questions regarding privacy are whether, why and how an application processes and/or stores what kinds of personal data. Unfortunately, a procedure which empowers the user to make an informed decision whether to install an application is not in the scope of the OpenSocial specification. As a result, applications make warning statements (if at all) in an proprietary manner, requiring the user to carefully read and understand their privacy policies. In the case that the data leaving the control sphere of the container is made explicit to the user at the time of installation, it is often not possible for the user to reassess the privacy policy during application lifetime. The extension of the Gadget XML specification to include such information would provide a standardized way of describing whether user data will be processed or stored by a third party. The container would be able to extract this information and present it in a container-specific and consistent manner, providing decision support to the user.

[5] http://www.vzlog.de/2009/10/neue-details-zu-opensocial-bei-studivz/ (German)

5 Conclusions

Securing SNS is challenging and becomes even more complex when third-party applications are able to access a user's data. This paper analyzed and discussed security implications in the context of OpenSocial-instrumented SNS. It showed that the OpenSocial specification is far from being comprehensive in respect to security. The paper introduced issues and depicted possible vulnerabilities resulting from these shortcomings. It presented different approaches to fill these gaps and proposed additions to the OpenSocial specification as well as a set of recommendations for containers that implement OpenSocial.

Although some of the proposals are still in an early stage and certainly require further research and development efforts, the paper contributes findings to raise awareness for existing security issues and provides suggestions and recommendations for resolving these issues. The results and suggestions will be brought to the attention of the OpenSocial Foundation with the goal to enhance upcoming versions of the specification and encourage the creation of guidelines for non-normative aspects.

References

1. Boyd, D., Ellison, N.: Social network sites: History, and scholarship. Journal of Computer-Mediated Communication 13(1), 210–230 (2007)
2. Häsel, M.: Opensocial: An enabler for social applications on the web. Communications of the ACM nn(n), nn–nn (2010)
3. Garrett, J.: Ajax: A new approach to web applications. Technical report, Adaptive Path Inc. (2005)
4. OpenSocial and Gadgets Specification Group: Opensocial specification v0.9. Technical report, OpenSocial Foundation (April 2009)
5. OpenSocial and Gadgets Specification Group: Opensocial gadgets api specification v0.9. Technical report, OpenSocial Foundation (April 2009)
6. Wiesmann, A., van der Stock, A., Curphey, M., Stirbei, R. (eds.): A Guide to Building Secure Web Applications and Web Services. The Open Web Application Security Project (2005)
7. Arrington, M.: First opensocial application hacked within 45 minutes, http://www.techcrunch.com/2007/11/02/first-opensocial-application-hacked-within-45-minutes/ (last accessed November 27, 2009)
8. Hammer-Lahav, E. (ed.): OAuth Core 1.0 Revision A (2009)
9. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press, Boca Raton (2001)
10. Fournet, C.: Verification tools for web services security. In: DIMACS Workshop on Security of Web Services and E-Commerce (2005)
11. McIntosh, M., Austel, P.: XML signature element wrapping attacks and countermeasures. In: SWS '05: Proceedings of the 2005 Workshop on Secure Web Services, pp. 20–27. ACM Press, New York (2005)
12. Bartel, M., Boyer, J., Fox, B., LaMacchia, B., Simon, E.: XML-Signature Syntax and Processing. W3C Recommendation (2002)
13. Gruschka, N., Lo Iacono, L.: Vulnerable cloud: Soap message security validation revisited. In: Proceedings of the IEEE International Conference on Web Services, ICWS (2009)

Security for XML Data Binding

Nils Gruschka[1] and Luigi Lo Iacono[2,⋆]

[1] NEC Laboratories Europe, Heidelberg, Germany
nils.gruschka@neclab.eu
[2] Europäische Fachhochschule (EUFH), Brühl, Germany
l.lo_iacono@eufh.de

Abstract. This paper introduces a complementary extension to XML data binding enabling the (selective) protection of structured objects and members. By this contribution, an object can be transformed into a *secured object* which contains encrypted and/or signed parts according to an assigned security policy. The serialization of secured objects results in XML data which is protected by standard XML security means. Thus, this approach introduces a data-oriented security mechanism which seamlessly integrates into XML data binding and therefore enables cross-platform (de)serialization of secured objects without the need of programming against a specific XML security API. Distinct entities in a distributed processing environment then operate transparently either on plain or secured instances of a class.

Keywords: XML, Data Binding, Data Protection, Security, Secured Objects.

1 Introduction

The extended markup language (XML) [3] provides a basic syntax and a set of encoding rules that can be used to share data between different kinds of computers, different applications, and different organizations. Its platform and language independent declarative description of data makes XML indispensable especially in system integration scenarios in which heterogeneous components and systems need to be coupled.

Different mechanisms exist to process XML data. The simple API for XML (SAX) [16] provides a mechanism to read data from an XML document. The parser operates in a streaming manner generating events which invoke user-defined and registered callback methods for processing these events. SAX has the advantage to be efficient in terms of resource-consumption and parsing time, but has a complex programming model which allows the read-only access to XML data. The document object model (DOM) [8] offers a tree-oriented API for random access to the XML data. Although DOM provides a much more intuitive programming model in comparison to SAX and allows a read as well as

⋆ This work was performed while Luigi Lo Iacono was with NEC Laboratories Europe, NEC Europe Ltd.

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 53–63, 2010.

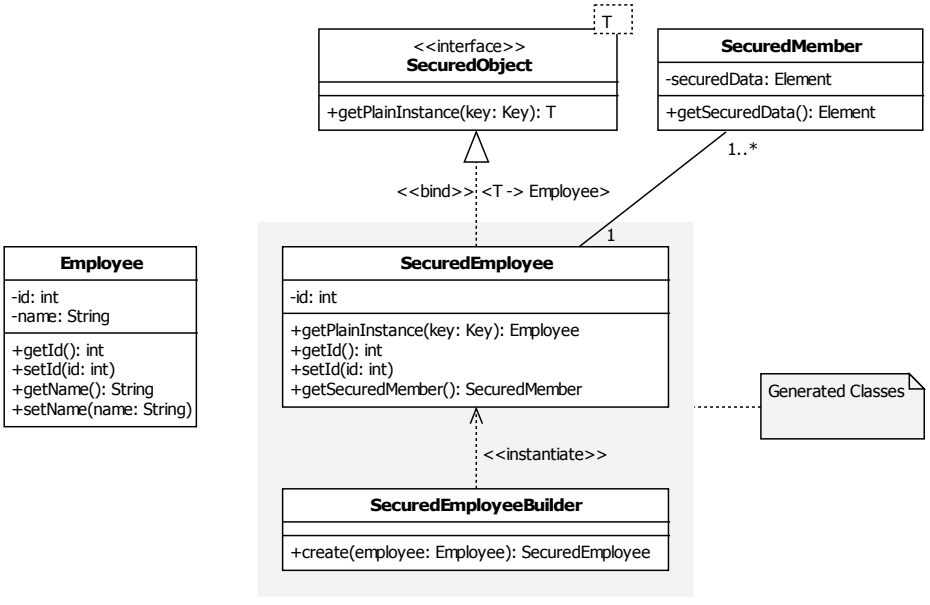


Fig. 1. Pattern of the secured object approach (including an example)

a write access to XML data, the trade-off comes with the required resources to build an in-memory tree representation of the XML data. Additionally, further approaches exist trying to combine the stream- and tree-based schemes in order to exploit the benefits of both [10,19].

All the above mentioned approaches have in common, that they define some kind of API with an associated programming model to retrieve the data from the direct representation of the XML document itself. With XML data binding [13] instead the information inside the XML document is mapped to an object of a specific programming language, enabling the conversion between an XML representation and an in-memory representation. This allows applications to access the data in the XML document from the object rather than using a specific API operating on the raw XML document. An XML data binding engine accomplishes this task by automatically creating a mapping between XML schema [17] types of the XML document and the programming language classes to be represented in memory.

XML data binding has become an essential building block to implement communications in distributed systems. Contemporary Web Service frameworks e.g. rely internally on XML data binding mechanisms to bridge between the service (or client) implementation and the Web Service messaging framework which are commonly based upon XML-based SOAP messages [2].

Of course, security issues are crucial relevant in distributed sytems using open networks. The adoption of transport-based security such as SSL/TLS [4] is a straight-forward solution, but does not fulfill requirements for more fine-grained

security mechanisms applied to individual elements contained in the structured objects. Even the message-oriented security mechanisms defined by WS-Security [14] for protecting SOAP messages only partly fulfill this requirement. While WS-Security allows securing selected parts of the message and the protection mechanisms "survive" multiple SOAP intermediaries, even here the transferred data is completely unprotected inside the ultimate receiver.

For protecting data beyond that point security mechanisms at the data level are needed. Integrating data-oriented security into XML data binding provides a solution for this issue giving developers the full convenience of working with data structures in their prefered programming language instead of dealing with according API and requiring the according knowledge to operate on the raw XML and applying XML security to it.

As a usage scenario consider an online shop offering a Web Service interface to the user. The order request messages contain in addition to the user's address and the ordered goods also the user's credit card number. All of this data should be kept confidential during transport from the user to the service. This task can e.g. be performed by WS-Security means which are even able to encrypt selective parts of the message. This can be useful if intermediaries between the client and the service must have access to parts of the message. But in any case, the message is completely decrypted inside the service. However, it is not necessary for the service to learn the credit card number. Thus, it is desirable that (just) the credit card number remains encrypted inside the order document. For billing purposes the shop can send the encrypted number to the credit card company, which is the only entity able to decrypt it.

An important property of XML data binding is the ability to (de)serialize objects across programs, languages, and platforms without the need of programming XML generation or processing code. Thus, requirements for a security solution for XML data binding include the capability to protect selective parts of structured objects while keeping the property of being platform and language independent. Additionally, from a programming perspective, a security system for XML data binding should not require programming of code to protect the objects or the serialized secured XML documents.

2 Related Work

To the knowledge of the authors, the existing approaches to secure software objects while being serialized to a a certain data sink (such as a file system, database or communication channel) are characterized by an "all-or-nothing" approach. Commonly, cipher stream classes are provided by XML data binding frameworks which can be plugged into the processing chain. The encipherment is performed before the serialized XML document is passed to its target (file system or communication channel). The decipherment is performed exactly in the opposite direction. First, the data is read in, then decrypted to an XML fragment and finally deserialized to the corresponding object hierarchy. JBoss Remoting [11] is an example for such a system. It provides a single API for

network-based invocations that uses pluggable transports and data marshallers, including an `EncryptingMarshaller` class for encrypting.

Other available approaches are even more general focusing either on the objects themselves or raw I/O streams. Examples again taken from the Java domain include the `SealedObject` [15] and `CipherOutputStream` [18] respectively. These approaches are not specific to XML data binding and can therefore not fulfill the granularity requirements at the object or XML document level.

Using XML security offers of course the required flexibility and selectiveness in applying security means to—also parts of—the XML document. Making use of XML security directly, however, breaks the processing model of XML data binding, since developers need to cope again with raw XML processing and XML security APIs which the data binding is aiming to hide. Moreover, the direct use of XML security requires a profund knowledge of the underlying concepts and specifications in order to ba able to implement effective security systems. An XML data binding integrated approach can henceforth lower the burden for developers and generate more reliable code which e.g. is not vulnerable to XML Signature Wrapping attacks [12].

Summarizing, no technology currently exists, which allows protecting sensitive data contained in software objects while being serialized to XML and at the same time preserves all benefits provided by XML in general and XML data binding in particular.

3 Security for XML Data Binding

To overcome these limitations of current approaches and to seamlessly integrate a (selective) protection of structured objects in XML data binding without requiring the programming of code and without the loss of platform and language neutrality, a new approach is introduced in this paper.

3.1 General Approach

The basic idea is illustrated in Figure 2. First, the field (or fields) to be secured are serialized to an XML tree fragment. On this tree fragment the intended securing means are applied using standard XML security means. The result is stored as XML element in the corresponding secured object. This secured object can be serialized using XML data binding methods and send to other peers. Depending on the scenario those peers operate either on the secured object or the plain object.

This approach leads to a number of interesting characteristics. First of all, such a secured object can be serialized and deserialized using standard data binding methods. This enables a non-security enabled peer to transparently operate on the secured object. Further, the transport message is an interoperable XML document conforming to the XML security standards. Additionally, the serialized message as well as the secured object contain all the necessary meta-data (algorithms, certificates etc.) for processing the secured parts.

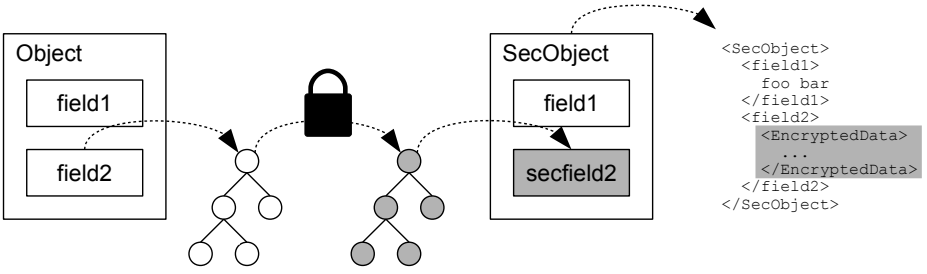


Fig. 2. Illustration of general approach

To define the parts to be protected an enhancement of the mapping specification between the object hierarchy and the XML document by security expressions is introduced. Developers can henceforth define object members to be protected by declaring them in the mapping specifications. A corresponding tool chain generates the secured counterpart objects according to this defition. The newly developed approach uses the XML security techniques XML Signature [1] and XML Encryption [9] to protect the sensitive object parts. This leads to a generic approach to manage the protected parts as secured XML elements inside specific secured mirror classes. Thus, together with the developed code generation mechanisms, the effort for a developer to introduce code for protecting the data in XML serialization in their own code is reduced to zero.

The two main artifacts introduced with this approach are the `SecuredObject` interface and the `SecuredMember` class (see Figure 1). The `SecuredObject` interface simply insures that all secured classes—which are generated by the framework—implement an operation to revert the security measures. The `SecuredMember` class contains the secured parts serialized as DOM element protected by XML security means. Depending on whether the parts to be protected are all within the same subtree or in distinct and disconnected subtrees, the secured class contains one or multiple associations with the `SecuredMember` class.

The proposed approach starts from the class, which defines the XML root element. For this class, a corresponding protected class is generated in case the mapping specification includes security declarations. As naming convention, the class name of the generated class will be constructed by prepending the term "`Secured`" in front of the original class name. This is also illustrated in Figure 1. The class which serves as XML root element is the `Employee` class. For this class and the contained members, the developer can specify what needs to be protected and how. In the same way as the developer specifies what members should become part of the serialized XML and whether as element or attribute, the developer is able to declare whether the members should be encrypted, signed, encrypted-before-signed or signed-before-encrypted. In this example, we assume the `name` member is to be encrypted. One can see in the example, that the `SecuredEmployee` does not contain the name, but instead a `securedMember` hiding it (in encrypted form).

As already stated, most of the code required to handle protected instances of classes is generated. A specific command line tool takes the name of the class representing the XML root element as starting point. It then generates two classes: the mirror class which contains the specified security mechanisms along with the enforcement and a builder class to create secured instances out of unprotected ones. Again, a naming convention is used to construct the class name of the builder. The word "`Builder`" is appended to the end of the corresponding `SecuredObject` class. In Figure 1 the class to build instances of `SecuredEmployee` is the `SecuredEmployeeBuilder`.

The reverse operation, i.e. generating an unprotected instance out of a secured one, is performed by calling the `getPlainInstance()` operation. This method is defined in the `SecuredObject` interface and needs to be implemented by all secured classes. Since the code of such classes is generated and due to the usage of XML security standards this is done automatically without the interaction or coding by a developer.

3.2 Inheritance and Composition

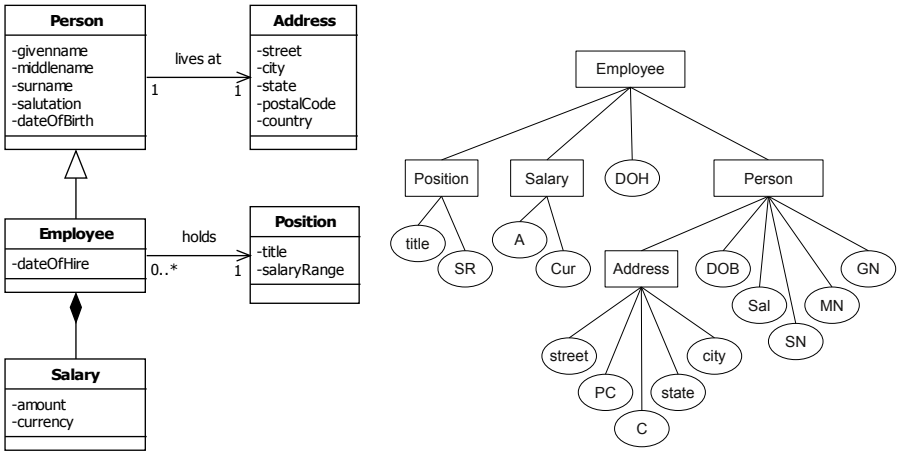
In cases in which a secured member is not a simple data type but a complex data structure or object hierarchy, all included data must be secured and is treated as defined for the class or its members respectively. Assume—as an extension of the previous example—that the whole `Employee` class presented before obtains some data items by inheritance and others by some form of association (see Figure 3a). By specifying that the `Employee` class must be encrypted, all members contained in it and especially the `Person`, `Position` and `Salary` class (either simple or complex) will be encrypted. Thus, security protection rules diffuse inside complex types. If this is too coarse grain (e.g. if the postal code must stay accessible), more selective policies can be defined for the contained complex object structures, by labeling all members but the `postalCode` member of the `Address` class to be encrypted.

This flexibility is realized by serializing the data structure to an in-memory DOM representation. Figure 3b shows the graphical DOM representation of an instance of the data model given in Figure 3a. On this tree XML Encryption and XML Signature means are applied and thus their capabilities regarding granularity can be fully exploited, meaning for example that security mechanisms can be applied to single nodes up to (distinct) sub-trees or even the complete DOM tree.

Finally, the developed design of the presented approach allows security expressions to be nested, resulting in an inside-out (i.e. starting with the adoption at the lowest level of the object hierarchy) application of the cryptographic primitives.

4 Prototype Implementation

The proposed approach has been prototypically implementated based on the Java Architecture for XML Binding (JAXB) [13] framework. Figure 4 gives an



(a) Structured object hierarchy (b) DOM representation

Fig. 3. Example data structure

overview of the JAXB concept. On the left-hand one can see, that classes are mapped to XML Schema elements. Depending on the scenario, the developer either creates a Java class and the JAXB framework generates the appropriate XML Schema or vice versa. This part is performed during compile-time. At runtime the framework converts the appropriate Java objects to XML documents and vice versa.

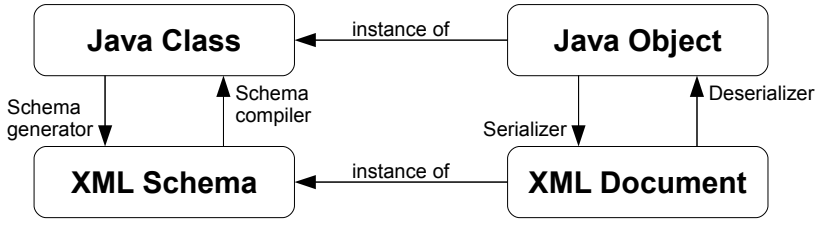


Fig. 4. JAXB concept

```
@XmlRootElement
public class Employee {
   private int id;

   @Encrypt
   private String name;
...
```

Listing 1.1. Security-related annotations

Since JAXB uses annotations to specify the mapping between the Java class and its XML representation, the proposed approach extends this scheme by security-related annotations for the expression of security policies. The prototype framework defines the `@Encrypt`, `@Sign`, `@EncryptBeforeSign` and `@SignBeforeEncrypt` annotations for this purpose. Note, that the annotations do not require to be configured with cryptographic parameters. Such are specified while creating instances of the secured class and can be defined dynamically during runtime as will be explained later in this section.

Listing 1.1 shows how the `Employee` class (used as an example in Figure 1) is enriched with these annotations, declaring to encrypt the employee's name. In order to generate the secured class, the `sogen` command line tool scans the Java class which defines the XML root element for these annotations in post-order tree traversal. For the example given in section 3, the `SecuredEmployee` class is generated for securing `Employee` instances. Listing 1.2 shows how to serialize a protected object using the generated classes.

```
// Create an instance of SecuredEmployee
Employee employee = new Employee(123456, "Luigi");
SecuredEmployeeBuilder seb = new
   SecuredEmployeeBuilder(cryptoSuite, key);
SecuredEmployee securedEmployee =
   seb.create(employee);

// Serialize the SecuredEmployee instance
// to the console
JAXBContext context =
   JAXBContext.newInstance(SecuredEmployee.class);
Marshaller marshaller =
   context.createMarshaller();
marshaller.setProperty(
   Marshaller.JAXB_FORMATTED_OUTPUT, true);
marshaller.marshal(securedEmployee, System.out);
```

Listing 1.2. Securing object instances before serialization

`SecuredEmployee` instances are created out of `Employee` instances by the `SecuredEmployeeBuilder` using a specified cryptographic suite and keying material. The code example given in Listing 1.2 also shows, that the usage of the JAXB framework is not changed by our security extension. The only difference is, that instead of providing a clear text instance of an object, its secured counterpart instance is passed to the `marshal()` operation. Since the protected object instance contains secured members as DOM elements, these elements are serialized as is. Listing 1.3 shows an example of a serialized `SecuredEmployee` resulting from the security policy contained in Listing 1.1. One can see that it contains an employee identifier in plaintext and one `SecuredMember` containing ciphertext using standard and interoperable XML security mechanisms.

```
<securedEmployee>
  <id>123456</id>
  <securedMember>
    <xenc:EncryptedData xmlns:xenc=".../xmlenc#"
      Type=".../xmlenc#Element">
      <xenc:EncryptionMethod
        Algorithm="...#aes128-cbc"/>
      <xenc:CipherData>
        <xenc:CipherValue>
          ClZU7W/aj4ElpRDeofETz8FsPI8T2T2dl9vedc0+
        </xenc:CipherValue>
      </xenc:CipherData>
    </xenc:EncryptedData>
  </securedMember>
</securedEmployee>
```

Listing 1.3. XML serialization of a secured object

The simple procedure for retrieving an unprotected object instance from its protected counterpart is shown in Listing 1.4. Since the generated `SecuredEmployee` class implements the `SecuredObject` interface, it contains the `getPlainInstance()` method. This operation perform the necessary transformations and checks (e.g. signature verification), inverts the cryptographic operations and returns an instance of the `Employee` class.

```
// Create an instance of Employee from a
// SecureEmployee instance
Employee employee =
   securedEmployee.getPlainInstance(key);
```

Listing 1.4. Reversal of a secured object

Note, that due to the explicit design and the usage of XML security to store the protected members inside the `SecuredEmployee` class, the meta data included inside the XML element of the `SecuredMember` class contains all required information to perform the cryptographic transformations and checks. Thus, just the keying material for secret key operations has to be provided .

5 Discussions and Conclusion

Despite all the criticisms related its verbosity, XML has established itself in a broad variety of applications and systems. XML data binding helps in eliminating XML processing drawbacks by offering a convenient method of handling XML data from within applications. Together with its ability to (de)serialize objects across programs, languages, and platforms it is a powerful tool for implementing heterogeneous distributed systems, e.g. as the core of SOAP-based Web Service frameworks.

For objects containing sensitive information security issues become relevant. The presented approach for securing objects introduces a mechanism that integrates seamlessly with XML data binding and allows for the (selective) protection of an object's structure and members without the need of using additional APIs and knowledge to work on the raw XML document. The concept of managing the protected members as secured XML elements using the XML security standards enables flexible data-oriented protection and furthermore keeps the properties of being interoperable and program, language and platform independent.

Other XML based systems can benefit from the data-oriented security approach introduced by this paper. Consider SOAP frameworks or XML databases as an example. When coupling the approach of secured XML data binding with HTTP [7] and the REST [6] concepts, light-weighted secured Web Services can even be realized. Further research is required to investigate this idea as a simple yet powerful replacement of SOAP as an alternative technical foundation for service-oriented architectures (SOA) [5]. In particular, the data-oriented vs. message-oriented security capabilities are an interesting study target, especially regarding the need for increased security within the endpoints themselves and not only during the transfer.

A proof of concept implementation in Java based on JAXB has been discussed, showing the ease-of-use and potential of this approach. During the evaluation of the presented prototype, it became clear, that additional policy expressions can further improve the usability. In cases, in which the number of members which are to remain unprotected is much larger than the number of members which must be protected, expressions to declare the exclusion from the protection inherited from the upper object hierarchy layer decreases the overall number of expressions required. Currently missing features are the creation of Java classes out of an XML schema definition and related aspects such as security policy expressions for XML schema.

Finally, it is planned to release these developments as an open source project in order to provide a platform to support future research and development work as well as to raise the awareness for the presented approach in standardization forums.

References

1. Bartel, M., Boyer, J., Fox, B., LaMacchia, B., Simon, E.: XML-Signature Syntax and Processing. W3C Recommendation (2002)
2. Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Nielsen, H.F., Thatte, S., Winer, D.: Simple Object Access Protocol (SOAP) 1.1. W3C Note (2000)
3. Bray, T., Paoli, J., Sperberg-McQueen, C.M., Male, E., Yergeau, F.: Extensible Markup Language (XML) 1.0. In: W3C Recommendation, 4th edn. (2006)
4. Dierks, T., Rescorla, E.: The Transport Layer Security (TLS) Protocol Version 1.1. IETF request for comments, RFC 4346 (2006)
5. Erl, T.: Service-Oriented Architecture – Concepts, Technology, and Design. Prentice-Hall, Englewood Cliffs (2005)

6. Fielding, R.T.: Architectural Styles and the Design of Network-based Software Architectures. Dissertation. University of California, Irvine (2000)
7. Fielding, R.T., Gettys, J., Mogul, J.C., Frystyk, H., Masinter, L., Leach, P., Berners-Lee, T.: Hypertext Transfer Protocol – HTTP/1.1. IETF request for comments, RFC 2616 (1999)
8. Hors, A.L., Hégaret, P.L., Wood, L., Nicol, G., Robie, J., Champion, M., Byrne, S.: Document Object Model (DOM) Level 3 Core Specification. W3C Recommendation (2004)
9. Imamura, T., Dillaway, B., Simon, E.: XML Encryption Syntax and Processing. W3C Recommendation (2002)
10. Java Web Services Performance Team. Streaming APIs for XML Parsers. Technical report, Sun Microsystems (2005)
11. JBoss Community. JBoss Remoting, http://jboss.org/jbossremoting/
12. McIntosh, M., Austel, P.: XML signature element wrapping attacks and countermeasures. In: SWS '05: Proceedings of the 2005 workshop on Secure web services, pp. 20–27. ACM Press, New York (2005)
13. McLaughlin, B.: Java and XML Data Binding. O'Reilly, Sebastopol (2002)
14. Nadalin, A., Kaler, C., Monzillo, R., Hallam-Baker, P.: Web Services Security: SOAP Message Security 1.1, WS-Security 2004 (2006)
15. Sundsted, T.: Signed and sealed objects deliver secure serialized content. JavaWorld (2000)
16. The SAX Project. Simple API for XML – SAX 2.0.1 (2002), http://www.saxproject.org/
17. van der Vlist, E.: XML Schema. O'Reilly, Sebastopol (2002)
18. Ward, A.: Encrypting the Java serialized object. Journal of Object Technologies (2006)
19. Zhang, J.: Simplify XML processing with VTD-XML. JavaWorld (2006)

Watermark Detection for Video Bookmarking Using Mobile Phone Camera

Peter Meerwald* and Andreas Uhl

Dept. of Computer Sciences, University of Salzburg,
Jakob-Haringer-Str. 2, A-5020 Salzburg, Austria
{pmeerw,uhl}@cosy.sbg.ac.at
http://www.wavelab.at

Abstract. In this paper we investigate a watermarking application for bookmarking of video content using a mobile phone's camera. A content identifier and time-stamp information are embedded in individual video frames and decoded from a single frame captured from a display device, allowing to remember ('bookmark') scenes in the video. We propose a simple watermarking scheme and blind image registration to combat the inherent geometric distortion due to digital/analog conversion. The work-in-progress shows promising results over previous approaches.

Keywords: Watermarking, image registration, geometric distortion.

1 Introduction

Watermarking has been proposed as a technology to embed an imperceptible, yet detectable signal in digital multimedia content such as images or video [1]. Since the watermark information is embedded in the video data itself and not a particular file format, the embedded information is retained even if the content undergoes transformation such as re-encoding or presentation on a monitor and capturing with a camera. Pramila et al. [2] survey the challenges in bridging the analog/digital gap using camera-based watermark extraction. Transmission of watermark information over the print/scan channel has been studied for applications including document authentication and copyright protection [3,4,5].

Nakamura et al. [6,7] present two watermark detection schemes for camera-equipped cellular phones: in [6], the authors aim to decode information such as an imperceptible content identifier analogous to a visible bar code in printed material while in [7], a content id is decoded from a sequence of video frames. Both methods rely on extraction of the target content region in the captured image and image sequence before applying projective correction to combat geometric distortion which inevitably results from freehand shooting. The side trace algorithm (STA) [8] employed requires a smooth background or an artificial border marker in order to identify the target region.

* Supported by Austrian Science Fund project FWF-P19159-N13.

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 64–74, 2010.

Stach et al. investigate the use of web cameras for watermark detection [9], most of the challenges identified (demosaicking, lens distortion, focus issues, white balance and gain compensation, compression) also apply to mobile phone cameras.

Watermarking robust to complex geometric distortion is a requirement for the digital cinema application [10,11,12] where the watermark may reveal the particular cinema in which in camcorder copy ('screener') was made: The early work [11] requires the original movie for synchronization, the later proposal [12] spread the watermark information over large temporal regions and thus cannot pinpoint individual frames; both approaches do not satisfy real-time detection requirement. Lee et al. [10] propose to use the the same watermark for information embedding and for geometric distortion estimation via local auto-correlation function; however, they require high-resolution video and a controlled capture environment.

In this work we investigate a watermarking application where a content identifier and time-stamp information are embedded in individual video frames and decoded from a single frame captured from a display device. This allows to remember ('bookmark') scenes in the video by means of decoding the embedded time-stamp and content id information. We discuss the application requirements and trade-offs in Section 2 and propose a simple watermarking scheme in Section 3. The blind image registration procedure to combat the inherent geometric distortion is presented in Section 4. Performance evaluation and experimental results are provided in Section 5, followed by concluding remarks in Section 6.

2 Application Scenario

In the envisioned application, a content identifier and time-stamp information are embedded in individual video frames. By taking pictures of video scenes displayed on a monitor with a mobile phone's camera, bookmarks can be conveniently stored as the embedded time-stamp information in the captured pictures links back to the particular scenes of the video content.

Hirakawa et al. [13] study the use of watermarking in conjunction with mobile devices and put forward marketing as a promising field of application instead of the well-researched copyright protection scenario. Nakamura et al. [6,7] propose a so-called Related Service Introduction System (RSIS) which enables auxiliary program information, online shopping, etc., based on the content identifier transmitted with the content itself.

For the video bookmark application, watermarking can be used to imperceptibly and unobtrusively store the time-stamp information in the video and thus cross the media boundary. Instead of watermarking, perceptual hashing could be used as a passive alternative, i.e. without having to modify the video content to embed the watermark information. Both technologies have their merits: (i) Watermark decoding can be performed stand-alone by the mobile device. This allows to give feedback to the user immediately, e.g. when capture conditions do not allow reliable detection. The embedded information can be extracted by the mobile device or the captured pictures can be uploaded to a server that does the processing. Clearly, there is a bandwidth versus processing trade-off. Perceptual

hashing requires a query to a database server to turn the perceptual hash into meaningful information. (ii) In case of active techniques, content preparation can be time-consuming due to the application of the watermark; furthermore, there is always at least a slight quality degradation due to watermark embedding. Perceptual hashing only requires to compute the hashes, without altering the content. (iii) Perceptual hashing cannot distinguish the same content coming from two different sources. On the other hand, by embedding different watermarks, one can link different services to the same content distributed via DVD or TV broadcast for example [7].

For the given video bookmarking application, 56 bits of watermark capacity is necessary to transmit 32 bits content id and 24 bits time-stamp information which permits temporal addressing of more than three days of video at 60 frames/second. A high video quality should be maintained; in this work we aim for 43 dB PSNR; this is considerable higher than the PSNR achieved by previous single frame watermark detection schemes (37 dB PSNR [6]), but lower than the quality obtained by methods utilizing multiple frames for detection (48 dB PSNR [10]; 49 dB PSNR [7]). Efficient processing is necessary for real-time application but we do not attempt to implement the algorithms on a mobile platform at this time.

3 Watermark Embedding and Detection

The watermark payload vector $\mathbf{b}$ is embedded in the luminance component of the video frames of size $N \times M$ pixels. $\mathbf{b}$ consists of $B = 56$ bits, a concatenation of the 32 bit content id and 24 bits time-stamp information identifying the temporal frame position. Payload bits are denoted $b[i] \in \{-1, 1\}$ with $1 \leq i \leq B$.

We perform a two-level discrete wavelet transform (DWT) with biorthogonal 7/9 filters and concatenate the resulting detail subband coefficients of the HL_2, LH_2, and HH_2 subband into the host signal vector denoted by $\mathbf{x}$. Next, we permute the coefficients of $\mathbf{x}$ and partition the resulting vector into B non-overlapping blocks of equal size, the coefficients of each block are denoted $x_i[k]$ with $1 \leq i \leq B$, $1 \leq k \leq S$ and $S = {}^{3 \cdot N \cdot M}/_{4 \cdot B}$. The permutation guarantees that the watermark information contributing to one payload bit is spread out equally over the host signal.

Embedding of the payload in the detail subbands using additive, spread-spectrum watermarking can be written

$$x'_i[k] = x_i[k] + \alpha \cdot b[i] \cdot w_i[k] \tag{1}$$

where α is the embedding strength. The spreading sequence $\mathbf{w}$ is generated pseudo-randomly with symbols $w[k] \in \{-1, 1\}$ of equal probability. No perceptual shaping is performed in this work, we simply set $\alpha = 3$.

3.1 Watermark Detection and Decoding

Watermark detection is performed *blind*, i.e. without reference to the original host signal. For efficient blind watermark detection, accurate modeling of the

host signal is required. We assume a Cauchy distribution of the DWT details subband coefficients and chose the Rao-Cauchy (RC) detector [14] whose detection statistic for the received signal $\mathbf{y}$ of length L is given by

$$\rho = \frac{8\hat{\gamma}^2}{L}\left[\sum_{t=1}^{L}\frac{y[t]\cdot w[t]}{\hat{\gamma}^2 + y[t]^2}\right]^2 \tag{2}$$

where γ is an estimate of the Cauchy distribution's scale parameter. To decide between the null- ($\mathcal{H}_0$, no or other watermark present) and alternative hypothesis ($\mathcal{H}_1$, watermark $\mathbf{w}$ present), the detection statistic ρ is compared against a threshold, $\rho \gtrless_{\mathcal{H}_0}^{\mathcal{H}_1} T$. The detection statistic ρ follows a Chi-Square distribution with one degree of freedom (χ_1^2) under $\mathcal{H}_0$ and, under $\mathcal{H}_1$, a non-central Chi-Square distribution with non-centrality parameter λ ($\chi_{1,\lambda}^2$) [15]. The detection threshold T can be established in the Neyman-Pearson sense for a desired probability of false-alarm using the relation $P_f = \mathbb{P}\{\rho > T|\mathcal{H}_0\} = Q_{\chi_1^2}(T)$ and the identity $Q_{\chi_1^2}(x) = 2Q(\sqrt{x})$ where the function $Q(\cdot)$ expresses the right-tail probability of the Gaussian distribution [16]. It follows that $T = \left[Q^{-1}(P_f/2)\right]^2$. For estimation of the Cauchy scale parameter γ, several fast, approximate methods are available, e.g. [17].

The RC detector has the advantage that the detection threshold does not depend on the received signal and can be precomputed to meet the desired false-alarm probability. Further, knowledge of the watermark strength is not necessary for detection [15]. These properties are important in the investigated application scenario as extensive experiments for threshold determination would be time-consuming to perform and the watermark power is in most instances significantly impaired due to the signal's analog/digital transition.

In order to retrieve the embedded payload, the target video content must be located and extracted from a captured image. Then the geometric distortion has to be estimated and reversed in a projective correction step. We discuss this blind registration process in the next section. For now, assume that we have a candidate received frame the same size as the original frame which is subjected to the DWT and used to construct the received signal vector $\mathbf{y}$ the same ways $\mathbf{x}$ was derived earlier.

First, we try to establish the presence or absence of any watermark information on the full host signal vector $\mathbf{y}$. Only if successful, the payload is decoded from the individual blocks $\mathbf{y}_i$. On absence, different parameters can be tried in the frame registration process described in Section 4 or in the shift compensation process, see Fig. 1.

For watermark detection, we remember that the watermark sequence $\mathbf{w}$ is known, but not the payload $\mathbf{b}$. Due to the square operation, the RC detection statistic for each block (Eq. 2) does not depend on the embedded payload bit. Assuming that the detector response of each block $\mathbf{y}_i$ is independent, we can exploit the reproductivity property of the Chi-Square distribution, namely that the sum of Chi-Square random variables is again Chi-Square distributed. We have B random variables $\rho_1, \ldots, \rho_B$, which all follow Chi-Square distributions

with one degree of freedom, thus the sum follows a Chi-Square distribution with B degrees of freedom, $\sum_{i=1}^{B} \rho_i \sim \chi^2_B$, and the combined threshold can be determined using the inverse of the Chi-Square cumulative distribution function with B degrees of freedom.

For payload decoding, we simply compute the sign of the modified detection statistic (without the square operation) for each received signal block

$$b[i] = \operatorname{sgn}\left(\sum_{k=1}^{S} \frac{y_i[k] \cdot w_i[k]}{\hat{\gamma_i}^2 + y_i[k]^2}\right) \tag{3}$$

which is equivalent to a bit-by-bit hard-decision decoder. The proposed decoder uses binary, antipodal constellation of codewords and space-division multiplexing.

In the proposed approach, no pilot or template watermark is used to estimate distortion, hence the entire watermark energy can be spent on encoding the payload. Compared to very recent work [10], a more sophisticated but still computationally efficient host signal model and corresponding detector is employed. Further, the spread-spectrum scheme is implicitly robust against certain valumetric distortion, in particular contrast change, which is an important advantage over quantization-based schemes [1].

4 Blind Image Registration

As a result of camera capture, the rectangular video frame is transformed into a quadrangle due to perspective projection [10]. For correlation-based detectors, the received signal and the reference watermark must be properly aligned (synchronized) for successful detection and decoding. The strategy we adopt in this work is to estimate and invert distortions in the detector, without reference to the original signal. Note that in our application scenario, no malicious attacker aims to thwart the decoding process by malicious transformation; on the contrary, the user may be asked to improve capture conditions or aid in identification of the target video frame.

The overall process is depicted in Fig. 1. Given a captured image, the first task is to locate the candidate target frame. We assume that the entire video frame has been captured and consumes a significant, large area in the image. Fig. 2 illustrates the processing steps for locating candidate target frames.

In the second task, a candidate target frame is then subjected to a corrective projection in order to invert the distortion. Given the coordinates of the target frame quadrangle, this transform can be easily computed, e.g. using ImageMagick's[1] *distort perspective* feature. The output of the corrective projection is a registered video frame, the same size as the original frame.

Finally, the registered, received frame is fed to the geometric shift compensation unit and the watermark detector. Shift compensation simply applies a small number of integer pixel shifts in each dimension before applying the DWT; this

[1] ImageMagick is available at `http://www.imagemagick.org`

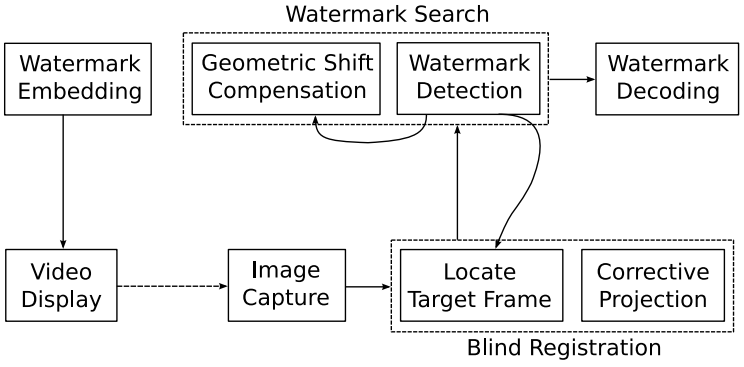


Fig. 1. Overview of video bookmarking system's components

step is necessary since the (decimated) DWT is not shift-invariant and produces drastically different results on shifted version of the same input.

4.1 Target Frame Localization

The goal is to produce a list of coordinate tuples (p'_1, p'_2, p'_3, p'_4) identifying the corner points of suspect video frames. To this end we use the contrast between the target video frame and the background to identify edges and, subsequently, corners of the target frame. As a pre-processing step, the captured image is Gaussian filtered (7×7 kernel) to suppress image noise. Then, Canny edge detection is performed to produce a gradient magnitudes image which serves as the input for a probabilistic Hough transform. The Hough transform computes a list of the dominant line segments from which approximately horizontally or vertically oriented lines are selected. We assume that geometry of the original video frame in only mildly distorted, i.e. border lines are rotated less than 10 degrees. The endpoints of the selected line segments are used as the center location for a local (33×33 pixels) template matching operation in the gradient magnitude image with the aim to pinpoint the corner coordinates. The templates are small, idealized corner images in four orientations. Note that the line segments already provide orientation information (left or right corner for a horizontal line, top or bottom corner for a vertical line) which is used in the template matching step; the template matching step removes the remaining ambiguity and allows to associate the point of peak normed correlation with one of the target frame coordinate, $p'_1, \ldots, p'_4$. If the maximum normed correlation computed by template matching is below a threshold (< 0.7), the line endpoint is discarded. In Fig. 4, a gradient magnitudes image is shown with line segments superimposed. Small circles denote line endpoints, large circles indicate potential frame corners.

Compared to a previous approach (STA, [8]), the proposed method does not require an explicit border marker, nor a uniform background for target frame localization. Nevertheless, blind localization of the target video frame is probably

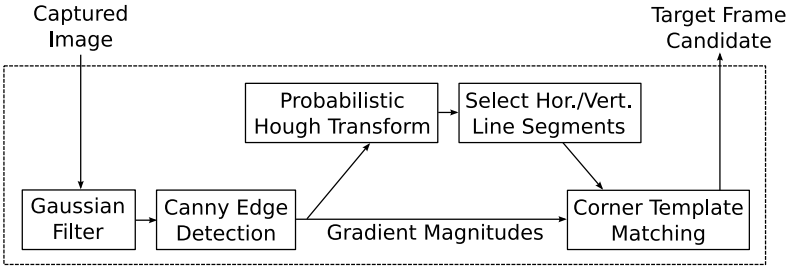


Fig. 2. Target frame location process within the blind image registration component

the Archilles' heel of the scheme; given the nature of the application, it might be acceptable to ask the user for assistance in a semi-automated way.

5 Experimental Results

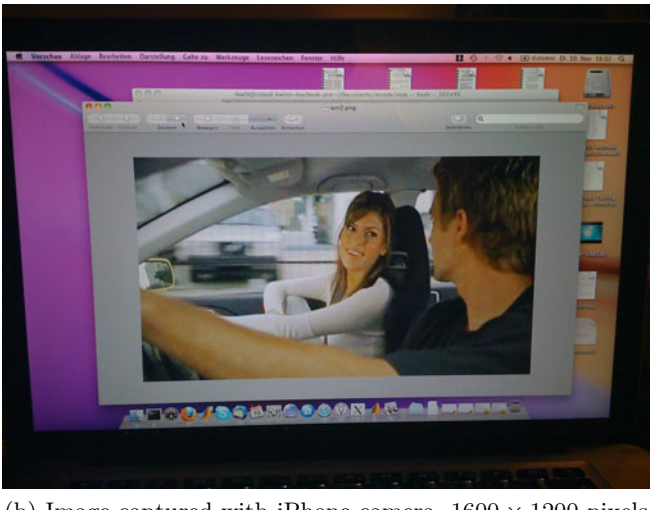
In order to assess the performance of the proposed watermarking system, we select two 480×272 pixel frames of the *2 Fast and 2 Furious* trailer video sequence. The watermark is embedded only in the luminance component to encode 56 bits of payload; the embedding strength $\alpha = 3$ results in an average PSNR of ~ 43 dB. In Fig. 3a we show the watermarked frame #2. The image captured using the camera of an Apple iPhone 3GS (1600×1200 pixels) is presented in Fig 3b.

A critical step is the localization of the target video frame in the captured image. The chosen picture, Fig. 3b, provides an interesting example as several quadrangles can be identified. In Fig. 4 we show the gradient magnitude image produced by the Canny edge detector (cf. Fig. 2). Detected horizontal and vertical line segments have been superimposed (in *red*). Small *(blue)* circles denote line segment endpoints while larger *(green)* circles identify the corner points of candidate quadrangular target frames and result from a positive corner template matching step. It can be seen that the corners of the target video frame are among the potential corners identified.

In Fig. 3c we show the correctly identified 480×272 target video frame extracted using the coordination from the localization step above, after applying corrective projection using ImageMagick. As becoming evident from the light-gray stripe at the top of the extracted frame, the identified top left corner has not precisely located the corresponding quadrangle's coordinate. This imperfection is mitigated in the watermark search step in the spatial (integer pixel offsets $0, \ldots, 3$) and wavelet domain (integer coefficient offsets $-1, \ldots, 1$).

In Table 1 we present the watermark detection results (ρ) and the achieved bit error rate (BER) for the decoded payload on two video frames. The images have been obtained with different mobile phone camera models (and one digital camera for comparison) covering a wide range of resolutions and picture qualities. No specific instructions were given regarding the presentation of the video frames; all captured images resulted from free-hand shooting.

(a) Watermarked 480×272 pixels video frame (43 dB PSNR).



(b) Image captured with iPhone camera, 1600×1200 pixels.

(c) Extracted 480×272 image after corrective projection.

Fig. 3. Example watermarked video frame

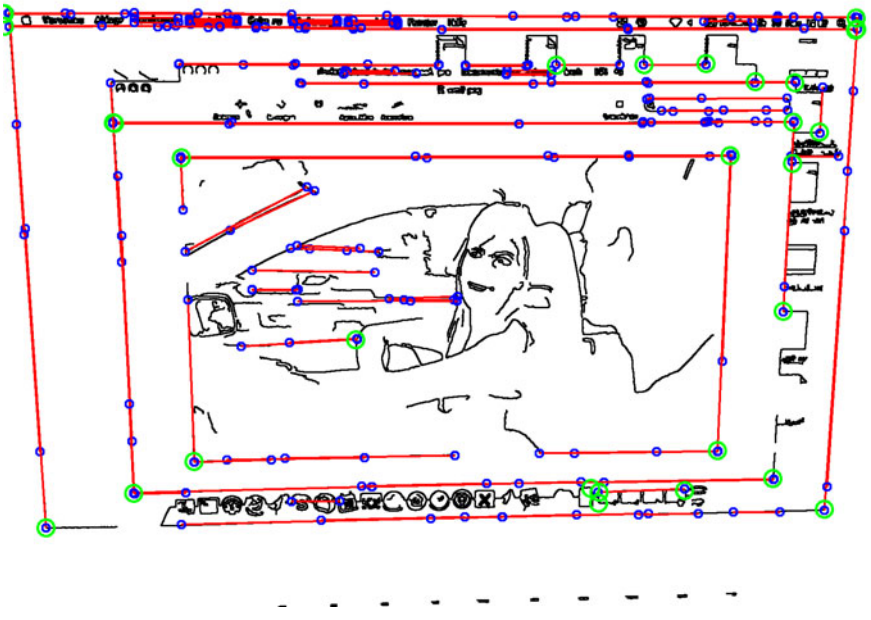


Fig. 4. Gradient magnitude image with line segments and candidate target frame corners superimposed

Table 1. Detection (ρ) and decoding results (BER) for two watermarked video frames (480×272) captured with different cameras; payload 56 bits, embedding with 43 dB PSNR

Phone / Camera Model and Resolution	Frame #1		Frame #2	
	ρ	BER	ρ	BER
Apple iPhone 3GS (1600×1200)	349.25	0.00%	901.07	0.00%
Motorola Razr v3 (640×480)	202.37	3.57%	204.10	5.36%
Nokia 6300i (1600×1200)	710.80	0.00%	1211.41	0.00%
Nokia 6303 (2048×1536)	*361.94	*1.79%	*213.96	*5.36%
Nokia E51 (640×480)	431.71	0.00%	1524.15	0.00%
Nokia N97 (2592×1944)	909.52	0.00%	1972.83	0.00%
Qtek 2020i (640×480)	320.28	0.00%	524.51	0.00%
Samsung SGH-X550 (640×480)	*204.77	*1.79%	220.37	1.56%
Samsung SGH-F480 (2560×1920)	452.11	0.00%	2483.59	0.00%
Sony Ericsson W200 (640×480)	482.55	0.00%	505.30	0.00%
Sony Ericsson W302 (1600×1200)	*162.27	*5.36%	667.95	0.00%
Sony Ericsson K550i (1632×1224)	*312.67	*0.00%	395.61	0.00%
Canon IXUS 70 (3072×2304)	586.16	0.00%	972.85	0.00%

For a false-alarm probability of $P_f = 10^{-9}$, the detection threshold $T \approx 144.30$ can be determined (assuming $\rho \sim \chi^2_{56}$ under $\mathcal{H}_0$). We observe that the detection statistic is above the detection threshold in all cases. In most cases, the 56 bit payload can be perfectly recovered. Even with camera resolution as low as 640×480 times, watermark recovery is possible. On the other hand, a high-resolution sensor does not guarantee error-free decoding in unfavorable capture conditions (lighting, small target frame size) as is the case for the Nokia 6303 images. All captured images have been JPEG compressed by the camera with default quality settings producing image files between over 500 KByte in case of the iPhone, or as small as 40 KByte in case of older Samsung and Sony models. Results marked with * have been obtained by manually specifying one of the coordinates of the target video frame, i.e. the automatic localization has failed.

All source code is available at `http://www.wavelab.at/sources`. Watermark embedding and detection/decoding has been implemented in Python using the `numpy`/`scipy` libraries[2], the target frame localization code builds on the OpenCV library[3]. The code is not optimized for performance. Watermark detection including all shift compensation steps takes approximately half a second on a Intel Core2 2.33 GHz. Blind registration consumes less than half a second to compute the target frame coordinates and approximately 700 ms for each perspective correction attempt.

6 Conclusion

We presented a watermarking application for bookmarking of video scenes by decoding time-stamp and content identifier information from captures frames using mobile phone cameras. We motivated our design decisions by comparing with recent related work. The unique requirement of the application to provide watermark information with high temporal resolution in the video content, and the monitor/camera capture transmission channel have not been studied before.

Universal Pictures Germany GmbH acknowledged that permission is not required to use screen shots taken of the *2 Fast 2 Furious* trailer for academic purpose. The author would like to thank all contributors who provided their mobile phones' pictures.

References

1. Cox, I.J., Miller, M.L., Bloom, J.A., Fridrich, J., Kalker, T.: Digital Watermarking and Steganography. Morgan Kaufmann, San Francisco (2007)
2. Pramila, A., Keskinarkaus, A., Seppänen, T.: Camera based watermark extraction - problems and examples. In: Proceedings of the Finnish Signal Processing Symposium 2007, Oulu, Finland (August 2007)
3. He, D., Sun, Q.: Practical print-scan resilient watermarking scheme. In: Proceedings of the IEEE International Conference on Image Processing, ICIP '05, Genova, Italy, vol. 1, pp. 257–260. IEEE, Los Alamitos (September 2005)

[2] See `http://www.numpy.org` and `http://www.scipy.org`
[3] Available at `http://opencv.willowgarage.com`

4. Solanki, K., Madhow, U., Manjunath, B.S., Chandrasekaran, S., El-Khalil, I.: 'print and scan' resilient data hiding in images. IEEE Transactions on Information Forensics and Security 1(4), 464–478 (2006)
5. Kim, W.G., Lee, S.H., Seo, Y.S.: Image fingerprinting scheme for print-and-capture mode. In: Zhuang, Y.-t., Yang, S.-Q., Rui, Y., He, Q. (eds.) PCM 2006. LNCS, vol. 4261, pp. 106–113. Springer, Heidelberg (2006)
6. Nakamura, T., Katayama, A., Yamamuro, M., Sonehara, N.: Fast watermark detection scheme for camera-equipped cellular phone. In: Proceedings of the 3rd International Conference on Mobile and ubiquitous Multimedia, College Park, MD, USA, pp. 101–108. ACM, New York (October 2004)
7. Nakamura, T., Yamamoto, S., Kitahara, R., Katayama, A., Yasuno, T., Sonehara, N.: A fast, robust watermark detection scheme for videos captured on camera phones. In: Proceedings of the IEEE International Conference on Multimedia and Expo, ICME '07, July 2007, pp. 316–319 (2007)
8. Katayama, A., Nakamura, T., Yamamuro, M., Sonehara, N.: New high-speed frame detection method: Side trace algorithm (STA) for i-appli on cellular phones to detect watermarks. In: Proceedings of the 3rd International Conference on Mobile and Ubiquitous Multimedia, College Park, MD, USA, pp. 109–116. ACM, New York (October 2004)
9. Stach, J., Brundage, T.J., Kirk, T., Bradley, B.A., Brunk, H.: Use of web cameras for watermark detection. In: Delp, E.J., Wong, P.W. (eds.) Proceedings of SPIE, Security and Watermarking of Multimedia Contents IV, vol. 4675, pp. 611–620. SPIE, San Jose (January 2002)
10. Lee, M.J., Kim, K.S., Suh, Y.H., Lee, H.K.: Improved watermark detection robust to camcorder capture based on quadrangle estimation. In: Proceedings of the IEEE Internationel Conference on Image Processing, ICIP '09, Cairo, Egypt, pp. 101–104. IEEE, Los Alamitos (November 2009)
11. Delannay, D., Delaigle, J.F., Macq, B.M., Barlaud, M.: Compensation of geometrical deformations for watermark extraction in the digital cinema applications. In: Proceedings of SPIE, Security and Watermarking of Multimedia Contents III, San Jose, CA, USA, January 2001, vol. 4314, pp. 149–157 (2001)
12. Lubin, J., Bloom, J., Hui, C.: Robust, content-dependent, high-fidelity watermark for tracking in digital cinema. In: Proceedings of SPIE, Security and watermarking of multimedia contents V, vol. 5020, pp. 536–545. SPIE, San Jose (January 2003)
13. Hirakawa, M., Iijima, J.: A study on digital watermarking usage in the mobile marketing field: Cases in Japan. In: Proceedings of the 2nd International Symposium on Logistics and Industrial Informatics, LINDI '09, Linz, Austria, September 2009, pp. 154–159 (2009)
14. Kwitt, R., Meerwald, P., Uhl, A.: A lightweight Rao-Cauchy detector for additive watermarking in the DWT-domain. In: Proceedings of the ACM Multimedia and Security Workshop (MMSEC '08), Oxford, UK, pp. 33–41. ACM, New York (September 2008)
15. Kay, S.: Fundamentals of Statistical Signal Processing: Detection Theory, vol. 2. Prentice-Hall, Englewood Cliffs (1998)
16. Nikolaidis, A., Pitas, I.: Asymptotically optimal detection for additive watermarking in the DCT and DWT domains. IEEE Transactions on Image Processing 12(5), 563–571 (2003)
17. Tsihrintzis, G., Nikias, C.: Fast estimation of the parameters of alpha–stable impulsive interference. IEEE Transactions on Signal Processing 44(6), 1492–1503 (1996)

Watermark-Based Authentication and Key Exchange in Teleconferencing Systems

Ulrich Rührmair[1], Stefan Katzenbeisser[2],
Martin Steinebach[3], and Sascha Zmudzinski[3]

[1] Technische Universität München, Department of Computer Science
[2] Technische Universität Darmstadt, Computer Science Department
[3] Fraunhofer Institute for Secure Information Technology SIT, Darmstadt

Abstract. In this paper we propose an architecture which combines watermarking with traditional cryptographic key agreement protocols to establish an authenticated or encrypted channel in teleconferencing systems. Technically the proposed method embeds messages of the key agreement protocol within an audio or video stream and is based on the assumption that the human communication partners can recognize each other easily; the watermark establishes a close coupling between the cryptographic key exchange messages and the media stream. We argue that the security of the scheme is based on a yet unexplored security property of digital watermarks; furthermore we present preliminary research results that suggest that this property holds in standard watermarking schemes.

1 Introduction

After the introduction of public-key cryptography, the problem of secure communication between two parties who have never exchanged secret keys seemed to be solved in practice. Ideally, communication partners access a trusted register (similar to a telephone book) in order to download each others public keys, which should be maintained by a trusted certification authority. Subsequently, the parties can use this key in order to encrypt and authenticate messages. Over three decades later, it has turned out that realizing this vision through a worldwide Public Key Infrastructure (PKI) is by no means simple. In fact, all attempts to establish a large-scale PKIs have failed due to organizational problems and lack of trust in certification authorities [2,3].

It is thus pressing to consider alternative ways of authentication and key exchange, which do not utilize a PKI, but still allow to establish an encrypted and authenticated channel between two parties. In this paper, we consider a teleconferencing scenario, where two parties use digital telephony or a video conferencing system to establish a connection. Often, both communication partners know each other well enough to recognize their voices and/or each other on the transmitted audio or video stream. We propose to utilize this human knowledge and mutual recognition ability to authenticate a channel; key transfer is realized by using a watermark, which is directly embedded in this channel. More precisely, we embed messages of a key agreement protocol inseparably into the transmitted audio or video signal by means of a robust watermark. The analog voice or video signals, which can mutually be recognized as genuine by the communication partners, provide an authenticated channel that inhibits real-time masquerading attacks.

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 75–80, 2010.

A natural way of establishing an encrypted phone or video conference utilizing the approach depicted above would work as follows: In the first phase, conversation is only transmitted in plain, but carries embedded information allowing secure key exchange. The messages required for key exchange are embedded by a watermark in the content. The two communication partners identify each other by recognizing the transmitted audio or video stream, and thereby implicitly authenticate the embedded key exchange messages. Once the key exchange has been completed, the system can encrypt all further communication between Alice and Bob in a second phase. The method seems suitable for establishing encrypted telephone or video communication in a highly convenient, ad hoc fashion, and would readily be usable with existing technology such as *skype* or similar services.

The approach is in principle similar to the Cryptophone architecture [1], where the communication equipment used by both communication partners sends Diffie-Hellman key exchange messages to each other. To detect a potential man-in-the-middle attack, the phones generate digests of the sent messages, which are displayed to the users. Both communication partners read the digest displayed on the phone over the encrypted line and the partner verifies if the hash was read correctly. If any discrepancy is detected, potentially a man-in-the-middle attack occurred. In comparison to the existing Cryptophone architecture, our approach is much more convenient as it does not require reading hash values aloud but ensures authenticity in the background of the communication. The approach also differs from [4], where a key is directly extracted from significant components of the speakers voices.

The rest of the paper is organized as follows. In Section 2 we propose a protocol that realizes the above mentioned authentication method in audio or video signals. Subsequently, Section 3 discusses the security of the proposed protocol; in particular we analyze which security property the underlying watermarking scheme needs to have to make the overall scheme secure.

2 Authentication Protocol

In this section we propose a protocol for teleconferencing systems that authenticates messages of key agreement protocols by help of the human user. As an example we describe the protocol in conjunction with the Diffie-Hellman key exchange. The protocol utilizes a watermarking scheme which only needs to be robust against varying channel conditions (the recipient will abort the protocol if no watermark is detectable). Instead, we use a different security assumption, which is detailed in the next section.

1. Both parties share the public system parameters: a random prime p, a generator g of $\mathbb{Z}_p^*$ and a global watermarking key K_W. These parameters can e.g. be distributed along with the communication software.
2. Alice chooses suitably and uniformly at random an exponent a.
3. Alice and Bob begin their telephone or video call.
4. Alice embeds in her part of the conversation during the first seconds a watermark by use of the watermarking key K_W. This watermark contains the payload $H(p,\, g,\, g^a \ mod\ p)$, where H denotes any cryptographic hash function. Furthermore, Alice sends the tuple $(p,\, g,\, g^a \ mod\ p)$ alongside the communication.

5. Bob verifies whether Alice's transmission was watermarked with key K_W, that the watermark payload is a hash of the sent key exchange message, and that he is speaking to Alice by recognizing her image/voice. Furthermore, Bob also uses a detector software to scan for artifacts of signal manipulations, similar to tools used in forensic investigations (see Section 3). If any test fails, or if he did not receive a watermarked content, he aborts the protocol.
6. Bob chooses an exponent b and embeds a watermark with payload $H(g^b)$ in the data stream (simultaneously to Alice). Furthermore, he sends g^b to Alice.
7. Analogously to Bob, Alice verifies whether Bob's transmission was watermarked with key K_W, that the watermark payload is the hash of the key exchange message and that she is speaking to Bob by recognizing his image/voice. She performs the same tests as Bob to detect any tampering with the audio or video stream. If any test fails, or if she did not receive watermarked content, she aborts the protocol.
8. Alice and Bob set their joint secret key to be $K = g^{ab}$.
9. Alice and Bob henceforth encrypt their communication with the key K. Optionally, Alice and Bob may send each other confirmation messages that are authenticated with the key K to verify that the key exchange was successful.

The described mechanism for authenticating messages via watermarks and voice and/or image signals does not necessarily need to be used in connection with the Diffie-Hellman key exchange protocol. Any other key exchange protocol that operates in one challenge-response round is also possible; in this case, the public Diffie-Hellman messages are replaced with the messages of the key exchange protocol.

Instead of authenticating the messages of a key exchange protocol, one can also directly exchange and authenticate the public keys of the communication partners via the described mechanism in the audio and/or video streams. As usual, these transferred public keys can subsequently be used for encryption and authentication of further speech or video signals.

As a further variant, one could imagine that communication partners record little voice and/or video samples, embed their public key in the sample via a watermarking scheme, and post these sequences on their web-pages. Others can download these samples, watch them to judge whether they are genuine, and recover the embedded public key. These sequences could also be recorded newly in short distances, for example on a daily basis, with recorded person pronouncing the current date. This would not overstretch the security properties of the watermarks: They merely had to remain secure for one day in that example.

3 Security

Since the underlying key exchange mechanism is known to be secure, attacks mainly stem from the watermarking layer. It is crucial to note that, due to the fixed symmetric watermarking key K_W, we have to assume that this key is available to an adversary as well. In principle we can distinguish between passive and active attackers; the former record the exchanged communication and try to obtain the key K, while the latter replace messages in order to impersonate one communication partner.

Passive attacks. The system inherits its security against passive attacks from the use of the Diffie-Hellman protocol. That is, an eavesdropper will not be able to restore K, unless he can break the cryptographic part of the protocol. However, since Diffie-Hellman key exchange provides no security against active attackers, security against active attackers must be considered separately.

Active attacks: Replaying unmarked sequences. In this type of attack, an adversary tries to obtain unmarked video or audio sequences of the communication partner (such as fragments of speech posted on web sites or captured by analog recording devices), selects its own Diffie-Hellman message and embeds the message into the signal by using its knowledge of the watermarking key K_W. In a more sophisticated system, an adversary may even assemble speech fragments of one communication partner in order to generate a new coherent speech signal. Note that the same attack is possible against the Cryptophone architecture (record the voice of the communication partner when he pronounces the individual characters of the message digest and use speech synthesis to create a speech signal for a new digest). This attack cannot be prevented by technical means, but must be tackled by the communication partners themselves. For example, both communication partners can try to individualize the initial (unencrypted) part of the communication, e.g. by saying the time, the date, or by having Alice and Bob pronounce randomly chosen, unusual words from a dictionary. Alternatively, Alice and Bob might ask each other questions that only the correct communication partner can answer, or take similar measures to assure that the communication is not replayed or modified. Note that prevention of this attack in our scenario seems to be easier than in the Cryptophone architecture, since it is considerably easier to automatically synthesize speech pronouncing a small set of message digests rather than complex text.

Active attacks: Masquerading Attacks. Another threat consists in an adversary who records one protocol run and replays it at a later time with a different watermark. If successful, such an attack would allow to alter the watermark payload (to point to the key of the attacker), while the audio file is still authenticated by the recipient. This attack is particularly problematic, since many phone calls resemble each other in the first part (i.e., the parties state their names, greet each other, etc.). To exploit this, Eve records one typical initial conversation of Alice and re-embeds her own watermark into the data stream (potentially after performing signal processing operations aiming at removing the first watermark). Subsequently she can mount a masquerading attack by replaying the newly watermarked sequence.

Protection against this attack requires a novel security property of the employed digital watermark: a communication partner must be able to decide whether the watermark was embedded in a previously unmarked media object or in one that already carried a watermark that was embedded with the same key; first results can be found in [6]. For the security of the scheme, both parties thus need to perform an analysis of the audio or video stream to detect such manipulation attempts. In contrast to the re-marking problem (where several watermarks are embedded with different keys), the particular problem has not extensively been discussed in the literature yet.

By applying methods from media forensics, it is possible to distinguish (within some error bounds) whether a media file has been marked before with the same key, since every watermarking process irreversibly changes the statistical characteristics of a media

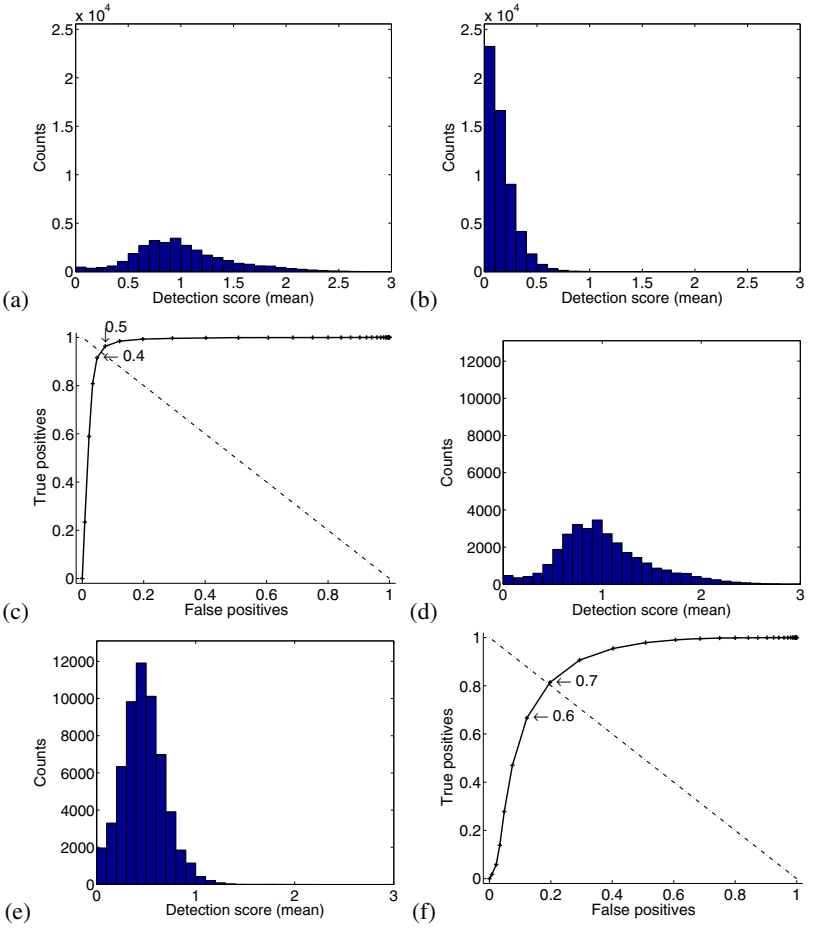


Fig. 1. Robustness against re-marking with the same key

object. To see whether this is possible using a standard watermarking scheme we performed some initial experiments. In particular, we used an audio watermarking system, which employs the Patchwork embedding method in the Fourier domain and is used in commercial products [5,6]. We tested different (mono) audio material of approximately 8.5 hours length, sampled at a rate of 44.1 kHz, into which we embedded a watermark sequence; subsequently we attacked the content in order to make a watermark visible where all payload bits are flipped. If a second watermark is embedded at the same strength as the first one (3dB in our experiment), the histograms of the detection statistics within single (Figure 1(a)) and doubly marked content (Figure 1(b)) are clearly distinguishable by a simple classifier which compares the value of the detection response by a threshold. The ROC curve of this classifier is shown in Figure 1(c); this plot shows that both cases can be distinguished with an EER of approximately 7%. If an attacker embeds the second watermark with a larger strength (6dB), the two cases

can still be distinguished (Figures 1(d) and 1(e) show the histograms of single and doubly marked content, while Figures 1(f) depicts the ROC curve), albeit at larger error rates (EER of approximately 20%). In addition, closer analysis shows that the adversary has to accept significant losses in sound quality, which might signal the attack to the communication partner. These preliminary results indicate that by using appropriate forensic analysis of the watermark detection responses, it is in principle possible to automatically determine whether a signal underwent malicious re-marking attacks under the same key. Further research is ongoing to determine the susceptibility of this classifier to other signal processing attacks.

4 Conclusions

We described a protocol that allows key establishment in digital telephony or video conferencing applications. In particular, we bind messages of a key exchange protocol as watermark to the audio or video stream and rely on the communication partners to recognize their voices. Due to the coupling of the messages and the digital data stream, the source of the messages is authenticated. We showed that the security of the approach crucially depends on a novel security property of watermarks, which has not been extensively discussed in the literature: one should be able to distinguish an object where a watermark was embedded in a previously unmarked media from one that already carried a watermark with the same key. Initial experiments showed that this question can in principle be answered (using a standard watermarking scheme) by analyzing the statistical properties of the detection response. Future work thus includes the definition of forensic methods that yield lower error rates.

References

1. http://www.cryptophone.de
2. Ellison, C., Schneier, B.: Ten Risks of PKI: What You're not Being Told about Public Key Infrastructure. Computer Security Journal XVI(1) (2000)
3. Ellison, C., Schneier, B.: Risks of PKI: Secure Email. Communications of the ACM 43(1), 160 (2000)
4. Enayah, M., Samsudin, A.: Securing Telecommunication based on Speaker Voice as the Public Key. IJCSNS International Journal of Computer Science and Network Security 7(3), 201–209 (2007)
5. Steinebach, M., Zmudzinski, S.: Evaluation of robustness and transparency of multiple audio watermark embedding. In: Proceedings of the SPIE, Security, Steganography, and Watermarking of Multimedia Contents X (2008)
6. Steinebach, M., Zmudzinski, S., Katzenbeisser, S., Rührmair, U.: Audio watermarking forensics: detecting malicious re-embedding. To appear in Proceedings of SPIE, Media Forensics and Security XII, vol. 7541 (2010)

Efficient Format-Compliant Encryption of Regular Languages: Block-Based Cycle-Walking

Thomas Stütz and Andreas Uhl

University of Salzburg,
Department of Computer Sciences
{tstuetz,uhl}@cosy.sbg.ac.at

Abstract. In this work an efficient format-compliant encryption approach for regular languages is proposed, block-based cycle-walking. The approach can transparently trade-off security for efficiency and can be adjusted to the desired level of security. The relationship between the encryption of regular languages and the encryption of JPEG2000 bitstreams is established. Block-based cycle-walking for the regular language of JPEG2000 bitstreams is compared to the extensive previous work from the multimedia security community and to the previous work from the cryptographic community, which is analyzed and evaluated.

1 Introduction

Format-compliant encryption has been intensely researched especially in the multimedia (security) community [11] and recently the cryptographic community contributed to the topic as well [1]. In this work, we define format-compliant encryption as ciphers where both plaintext and ciphertext are in the same format, i.e., meet all the syntactical and semantical requirements of the format definition.

Format-compliant encryption is the key technology to enable security in environments in which security has not been initially incorporated in the design, e.g., PGP can be considered the most widely deployed format-compliant encryption system for the mail format. Other examples are the format-compliant encryption of social security numbers, credit card numbers, 512-byte disk sectors, database entries of a certain type, and multimedia formats, such as JPEG, JPEG2000 and H.264.

For multimedia formats, format-compliant encryption can offer significant application advantages. Modern compression standards, such as JPEG2000 and H.264:SVC, generate a scalable representation of the visual data, that can be efficiently adapted. The preservation of scalability in the encrypted domain allows to perform adaption in the encrypted domain. The preservation of scalability is of fundamental importance for the secure streaming of multimedia content [6]. There have been numerous proposals for scalability-preserving encryption of JPEG2000 (see [3] for a survey), which has also been standardized within JPEG2000 (JPSEC) [8]. Most of the proposed JPEG2000 encryption schemes encrypt only the coded coefficient data (called bitstream in JPEG2000), while

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 81–92, 2010.

preserving header data. These schemes require encryption schemes which preserve the format of the JPEG2000 bitstream. Our approach is to use the formal description of the JPEG2000 bitstream, i.e, of the regular language describing it, in order to develop a simple yet efficient bitstream-compliant encryption scheme.

Thus in this work we focus on efficient encryption of regular languages, and specifically we show that JPEG2000 bitstreams are a regular language and thus the cryptographically secure format-compliant encryption algorithms (in the sense of the strong notion of message privacy, i.e., MP-security), cycle-walking and RtE (Rank-then-Encipher) [1], can be adapted. We answer the question whether RtE and cycle-walking are computationally efficient enough for practical application. We also propose a new notion of security, MQ security (message quality), which allows to differentiate between weaker and stronger encryption schemes, which are not MP-secure. Block-based cycle-walking is presented, a novel approach. It is the first efficient JPEG2000 encryption approach that does not rely on the availability of an external IV (initialization vector). An extensive analysis and evaluation of its security and complexity is conducted and it is compared to the state of the art.

2 Regular Languages

Regular languages have an important role in computer science and applications, specifically in the definition of data formats. In the scope of this work it is most interesting that for regular languages given as DFA (deterministic finite automaton) efficiently computable (linear complexity) ciphers with proven security have been proposed. Regular languages can be described in different ways, e.g., by a DFA accepting the language, or by a regular expression generating the language. A DFA can be formalized as a tuple $(Q, \Sigma, \delta, q_0, F)$; Q is the set of states, Σ is an alphabet, $\delta : Q \times \Sigma \rightarrow Q$ is the transition function, q_0 is the start state and F is the set of accept states, following the conventions of [9].

In [7] it is stated that the JPEG2000 bitstream must not contain sequences above 0xff8f and must not end with 0xff. In the following we show that these requirements are equivalent to a regular language, i.e., that all bitstrings obeying the above mentioned syntax requirements can be produced by a regular expression, and are accepted by a DFA.

JPEG2000 Bitstreams Are A Regular Language. The regular language of valid JPEG2000 bitstreams B is accepted by a DFA, with only three states, $Q = \{q_1, q_2, q_3\}$, $\Sigma = \{0\text{x}00, \ldots, 0\text{xff}\}$, the start state $q_0 = q_1$ and a single accept state $F = \{q_1\}$. The DFA is illustrated in figure 1, from which the transition function δ can be easily derived. The regular expression describing the language of JPEG2000 bitstreams is

$$(0\text{xff}(0\text{x}00 - 0\text{x}8\text{f}) \cup (0\text{x}00 - 0\text{xfe}))^*$$

where "()" groups elements to a new regular expression, "$(x - y)$" denotes all the elements between x and y and $*$ denotes 0 and more repetitions of a regular

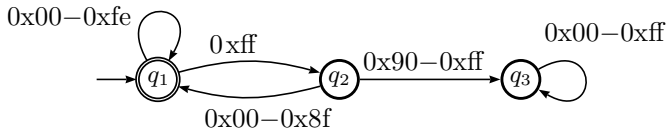


Fig. 1. DFA for B

expression. The equivalence between the syntax requirements for JPEG2000 bitstream and the regular language B can be verified briefly: no string ending with 0xff can be generated as 0xff always yields a non-accept state and if a symbol in excess of 0x8f follows, the DFA goes in a terminal reject state, and thus no sequence in excess 0xff8f is accepted.

3 Format-Compliant Encryption

In [1] two secure generic format-compliant encryption schemes, especially for regular languages, are discussed and analyzed.

Cycle-Walking. Cycle-Walking encrypts format-compliant messages of a certain length (n bits) and employs an n-bit block cipher for encryption. The plaintext is repeatedly encrypted until the ciphertext is format-compliant. For decryption the ciphertext is decrypted until it is format-compliant. As one decryption step exactly inverts one encryption step, the first format-compliant decryption has to be the plaintext. The scheme is as secure as the underlying n-bit block cipher [2] (see section 4 for a discussion on variable length block ciphers).

Rank-then-Encipher. If the format-compliant strings are sparse (in the n-bit domain of the block cipher) the computational complexity of cycle-walking is huge, as it is very improbable that a format-compliant string is produced in the pseudo-random encryption process. In order to reduce the sparsity, ranking of the language L can be employed. Ranking assigns each string of L with length n an unique number, its index, and the indices are consecutive natural numbers in the range of 1 to $|L_n|$, the number of strings of L with length n. The index can then be encrypted with the cycle-walking technique and a variable-length block cipher, which reduces the expected number of necessary iterations i for cycle-walking ($\mathrm{E}(i) \leq 2$).

In [1] it is pointed out that regular languages can be efficiently ranked and algorithms are given.

JPEG2000 Bitstream Encryption. For the JPEG2000 bitstream numerous algorithms have been proposed, an overview is given in [3]. A variation of the cycle-walking technique has been proposed for JPEG2000 bitstreams[13]. Apart from cycle-walking, all proposed length-preserving schemes preserve to a certain extent plaintext bits and are thus not secure with respect to the strong

cryptographic security notion of MP-security (see section 5.1). An **exemplary efficient algorithm** [12] preserves two-byte-sequences starting with 0xff and encrypts the remaining bytes in the range 0x00−0xfe. In figure 1, this corresponds to encrypting only values in the loop of state q_1, a simple explanation why format-compliance is preserved. While the format-compliance of this algorithm is easily shown, for more complex algorithms this is quite a tedious task and several flawed algorithms have been proposed. Also the amount of preserved plaintext data is tedious to assess for more complex algorithms.

4 Block-Based Cycle-Walking

Block-based cycle walking is a simple, efficient algorithm for the format-compliant encryption of regular languages. Security can be transparently traded off against efficiency, by choosing the appropriate block size. The approach depends on the specifics of the language.

The encryption algorithm splits an arbitrary length plaintext string into blocks. Each block consists of symbols, which determine a sequence of states of the describing DFA. These blocks are processed in order. Each block is encrypted repeatedly until it is format-compliant and the last state of the state sequence of the ciphertext is the same as of the state sequence of the plaintext. In the determination of format compliance the previous block's end state has to be considered, i.e., the DFA starts in the previous block's end state.

The decryption algorithm splits the ciphertext into blocks. Each block is repeatedly decrypted until it is format-compliant and its end state is the same of the ciphertext block (which of course is the same as of the plaintext block). In the determination of format compliance the previous block's end state has to be considered as well. Thus for every block concise conditions for the determination of the necessary iterations are available and the cycle-walking technique can employed on each block.

The scheme elegantly reduces the complexity of cycle-walking of the entire string to the complexity of cycle-walking small blocks. The security breach is limited to the preservation of a single state per block, which has different implications on the security for different regular languages.

In the following we discuss the approach for JPEG2000 bitstreams in detail.

Algorithm Description. The format-compliant encryption of a JPEG2000 bitstream is given in listing 1.1, parameters are a string of bytes, which has to be format-compliant, the key and the block length l. The bitstream is split into blocks and each block is encrypted with the function fc_encblock which additionally gets the last byte of the previous block which is used in the preservation of format-compliance. The function get_block(f, l) returns the next block of length l, except for the last two blocks (in order to cope with the minimal block size of the underlying cipher). Internally in get_block, the bitstream is split into blocks of length l the last block can have arbitrary length $\leq$ l. Each block is returned except for the last two blocks, for which the behaviour depends on

Listing 1.1. Format-compliant encryption

```
fc_bitstream fcencrypt( fc_bitstream f,
    key k, block_length l)
{
    plb = 0;
    t = 0;

    while( more_blocks( f, l ) )
    {
        block = get_block( f, l );
        lb = last_byte_of( block );
        fc_encblock( block, plb, k, t++ );
        plb = lb;
    }
    return f;
}
```

Listing 1.2. Format-compliant encryption of a block

```
fc_block fcenc_blk( fc_block b,
    last_byte_of_previous_block plb,
    key k, tweak t )
{
    lb = last_byte_of( b );
    while ( !is_valid( b, plb, lb ))
    {
        enc_vblock( b, k, t );
    }
    return b;
}
```

the minimal block length m the underlying cipher can handle: if the length of the last block is below m, the preceding block is returned with the last block appended. The function more_blocks implements this behaviour.

The encryption of a block is given in listing 1.2. Blocks are simply encrypted until they are valid, i.e., containing no sequence in excess of 0xff8f, the first byte must be below 0x8f if the previous byte is a 0xff byte, and for the last cipher byte must preserve whether the byte was 0xff (see listing 1.3). The function enc_vblock calls an variable length block cipher with minimal length m.

The format-compliant decryption works exactly the same, except that the decryption function of the variable length cipher is called.

Requirements. The algorithm relies on the availability of an arbitrary length cipher. An arbitrary length tweakable block-cipher would be optimal. Having a secure block cipher with a fixed block size, e.g., AES with 128 bit, secure block-ciphers with a larger block size can be constructed [1]. Currently there are no secure constructions for smaller block size ciphers known [2]. Thus a BBCW block length smaller than the block size of the employed block cipher is not directly supported, solutions are padding or method 1 of [2]. If the last block is smaller than the cipher block size it can be encrypted jointly with the preceding block, as block ciphers with a larger block size can be constructed.

5 Security Analysis and Evaluation

In order to analyze and compare the security of block-based cycle walking, we first discuss sensible notions of security for format-compliant encryption.

Listing 1.3. Checking the validity of a block

```
bool is_valid( fc_block b,
    last_byte_of_previous_block plb,
    last_byte_of_current_block lb )
{
    bsc = contains_seq_in_excess_of_0xff8f( b );
    bprevblk = (plb != 0xff) ||
            (first_byte_of( b ) <= 0x8f);
    if (lb == 0xff)
        blb = last_byte_of( b ) == 0xff;
    else
        blb = last_byte_of( b ) != 0xff;

    return  bsc && bprevblk && blb;
}
```

5.1 Security Notions

A security notion defines the precise meaning of the term security. We adopt the framework of [1], but refrain from most technical details here. In [1] the security notions, MP security and MR security are formalized as code-based games, the notion of MQ security is introduced in this work.

MP security. For MP security (message privacy) an adversary must not be able to compute any property of plaintext from the ciphertext. E.g., an encryption scheme preserving the digit sum of the plaintext is not secure in the sense of MP security.

MR security. For MR security (message recovery) an adversary must not be able to compute the plaintext from the ciphertext. E.g., an encryption scheme preserving the digit sum of the plaintext can be perfectly secure in the sense of MR security.

We think that for the majority of applications MR security is too weak, while MP security is too strong. A good example is image encryption: an MR-secure encryption scheme may still give an adversary access to a visually indistinguishable high-quality approximation of the plaintext, while a perfect recovery of the plaintext is infeasible (consider LSB encryption). MP security is, however, not the security goal as e.g. the preservation of checksums in the encrypted domain could even be considered a beneficial feature (error correction could be done on the ciphertext!). A similar rationale is necessary for format-compliant encryption: provably MP-secure encryption is often practically infeasible (see section 6), but we still want to distinguish "really insecure" schemes, that encrypt only a small fraction of the plaintext, from "rather secure ones". We want to define a security notion that is capable to distinguish between a scheme which only encrypts a block of the plaintext (as MR-secure as the block cipher, e.g., AES) and a scheme which only preserves few bits, commonly to ensure format-compliance.

MQ security. In MQ security (message quality), security is defined by the inability of an adversary to compute an approximation of the plaintext with higher "quality" than targeted. The quality needs to be measured, for the encryption of regular languages we propose the nHd (normalized Hamming distance) of the

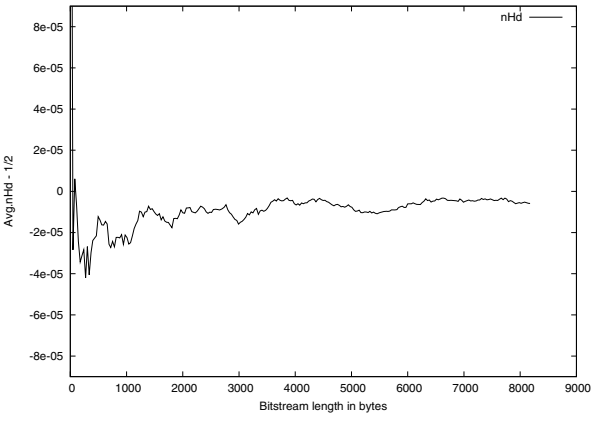


Fig. 2. Average nHd-1/2 of 10.000 random string pairs of B

binary representation of the strings of the regular language as security metric. Thus schemes that only encrypt a small fraction are insecure (nHd ≈ 0). Other security metrics, modeling the actual security requirements of an application, can be employed in the MQ security notion (e.g., the MSE for images). MQ-secure schemes should yield a nHd of plain- and ciphertext that is close to $\mathrm{E}(\mathrm{nHd})_L$, the expected nHd of two randomly-chosen, equal-length strings of the language L. An adversary is successful in breaking a scheme in MQ security if she computes an approximation that is closer to the original as defined in the security definition, i.e., if she can compute a reconstruction Y with an nHd to X that differs from the expected (the security level can be adjusted by ϵ): $|\mathrm{nHd}(X, Y) - \mathrm{E}(\mathrm{nHd})_L| < \epsilon$.

MQ-security with the nHd as metric is rather similar to defining a certain encryption ratio as secure. The encryption ratio, however, is hard to assess if we consider the encryption process a black box (the encryption ratio of some schemes depends on the plaintext and the key), which randomly preserves some of the plaintext data.

In the case of arbitrary bitstrings ($A = \{0, 1\}^*$) and a uniform distribution of the plaintexts, $\mathrm{E}(\mathrm{nHd})_A$, the expected nHd of two equal-length random strings is $1/2$, for other languages L the $\mathrm{E}(\mathrm{nHd})_L$ can differ. In the following we investigate the $\mathrm{E}(\mathrm{nHd})_B$ for the regular language of JPEG2000 bitstreams B. On average the nHd of two randomly chosen strings of B (uniform distribution), which is an estimate of $\mathrm{E}(\mathrm{nHd})_B$, is very close to $1/2$, as can be seen in figure 2.

Relations between notions. MP security implies MQ security, which implies MR security for sensible choices of d and ϵ.

$$\mathrm{MP} \Rightarrow \mathrm{MQ}_{\mathrm{nHd},\epsilon>0} \Rightarrow \mathrm{MR}$$

The implications work only in one direction. Thus an MQ-secure scheme needs not to be MP-secure.

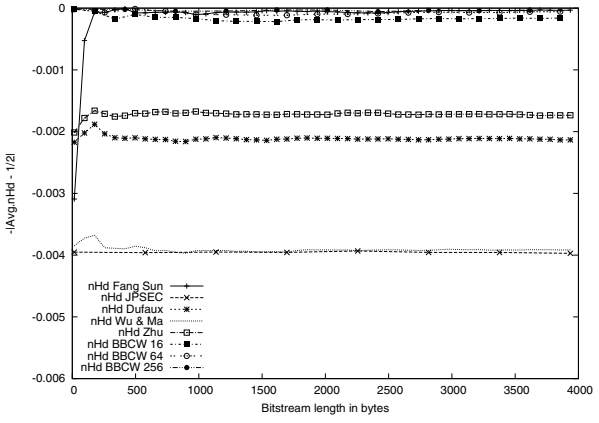


Fig. 3. Average normalized Hamming distance for different encryption approaches

Analysis. Of the discussed format-compliant encryption algorithms, only cycle-walking [2] and RtE + cycle-walking achieve MP-security. Block-based cycle-walking only preserves data if the last byte of a block equals 0xff. Thus in the worst case every block's last byte 0xff byte is preserved, on average every 1/256 block's last byte is a 0xff byte, while the best case is that no byte is preserved. Thus the desired level of MQ security can simply be achieved by choosing the appropriate block size. For the exemplary efficient algorithm (see section 3) the worst case is that every 2^{nd} byte is an 0xff byte and thus the entire string is preserved (MR insecure!), in average every 256 byte equals 0xff and in the best case no 0xff byte is present.

Evaluation. The security of block-based cycle-walking is experimentally compared to previously proposed JPEG2000 bitstream encryption algorithms. Note that the experimental evaluation of security only gives an upper-bound for MQ-security, i.e., schemes evaluated as insecure are insecure, schemes evaluated as "secure" may have hidden flaws. A simple example is an "encryption" scheme which inverts the first half of the bits and preserves the other, the average nHd would be exactly 1/2, while an adversary can easily compute a reconstruction with nHd zero, by inverting the first half again (though MQ-insecure). The average nHd of 10.000 plaintext ciphertext pairs for each length is illustrated in figure 3. Block-based cycle-walking with block size b is labeled "BBCW x", the exemplary efficient encryption algorithm (see section 3) is labelled "Wu Ma". The other algorithms follow the nomenclature in [3].

Of all evaluated schemes (each scheme discussed in [3] has been evaluated) only the approach of [4] achieves a comparable MQ-security level very close to 1/2, the expected nHd of two random strings of B, the regular language of JPEG2000 bitstreams. For a very large string length $n = 2^{20}$ results for selected algorithms are given in table 1.

Table 1. Average nHd for 1MB and 100 trials

Fang	JPSEC	Kiya	Kiya10	Dufaux	WuBlk	Zhu	bbcw16	bbcw128	bbcw256	AES
.50009	.49606	.23444	.02344	.49787	.49611	.49830	.50001	.49999	.50000	.50001

6 Complexity Analysis and Evaluation

The runtime performance of block-based cycle-walking (in dependency of the block size), other JPEG2000 bitstream encryption algorithms, cycle-walking and RtE is compared, analyzed and evaluated. An interesting question is whether RtE performs better than cycle-walking, the theoretical analysis is in favour (if the format-compliant strings are sparse), but the increased complexity of multi-precision arithmetic is not considered in this analysis.

6.1 Analysis

The runtime complexity of cycle-walking for JPEG2000 bitstreams has been shown to increase exponentially with the string length n ($\mathcal{O}(2^n)$) [10]. The space complexity grows linearly, i.e., is in $\mathcal{O}(n)$.

The ranking of a string of a regular language consists of two steps: build the table T and rank (requires T). The table needs to be built only once for several ranking operations of same length strings. Building the table has complexity $\mathcal{O}(n|Q||\Sigma|)$, and the table is of size $\mathcal{O}(n|Q||\Sigma|\log|B_n|)$, where $|B_n|$ is the number of strings with length n. The runtime complexity of ranking is $\mathcal{O}(|\Sigma|n)$ and the space complexity is constant, but T is required. The arithmetic, however, has to be performed on numbers which correlate in size with $|B_n|$.

The runtime of block-based cycle-walking is linear in the length of string and for the block size b the complexity is the same as that of cycle-walking, which is efficient for short JPEG2000 bitstreams (below 0.001s for $b < 1500$ bytes, see figure 5(a)). The space complexity is constant, as the bitstream can be read block by block.

Most of the JPEG2000 bitstream encryption approaches discussed in [3] have a runtime linear in n and constant space complexity, i.e., are capable of stream processing.

6.2 Evaluation

The experiments have been conducted on a Intel Core2 6700@2.66GHz, using custom C implementations which employ GMP 4 (gnu multi-precision library). The source code of the implementations and the experiments will be available at www.wavelab.at. As secure encryption primitive AES has been employed, the variable length cipher is mimicked by AES in ECB mode with ciphertext stealing, which may result in too optimistic results for cycle-walking (but cycle walking is infeasible even with this efficient implementation).

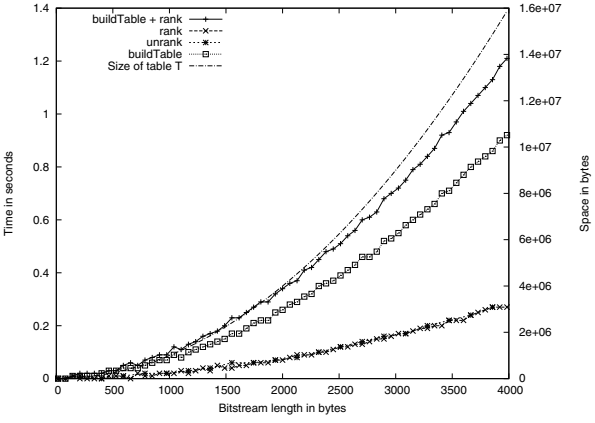
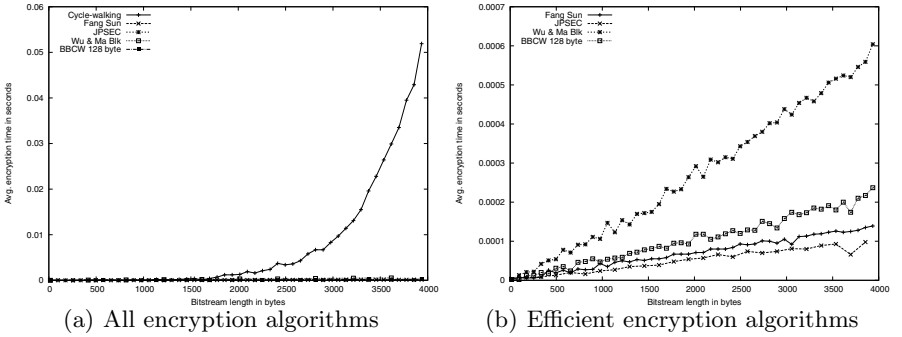


Fig. 4. Space and time of RtE



(a) All encryption algorithms (b) Efficient encryption algorithms

Fig. 5.

Cycle-walking, RtE. The actual runtime complexity of ranking is not linear, the multi-precision arithmetic takes its toll (see figure 4). The size of the table T (the number of entries of T is linear in n) is illustrated in figure 4. The size of its entries correlates with $\log |B_n|$ and $|B_n|$ increases exponentially with n [10]. Plain cycle-walking is way more efficient than RtE (compare figure 4 and figure 5(a)). However, for larger string length n the complexity of cycle-walking increases dramatically as well.

Efficient JPEG2000 Bitstream Encryption. The results have been obtained by averaging 10.000 independent trials. The runtime performance of cycle-walking and efficient JPEG2000 encryption algorithms is illustrated in figure 5(a). The average encryption time of cycle-walking increases exponentially with increasing string length, which greatly reduces the applicability of cycle-walking for JPEG2000 encryption [10]. In figure 5(b) the runtime performance of efficient JPEG2000 bitstream encryption approaches is compared in detail. The performance of block-based cycle-walking with different block sizes has been evaluated

Table 2. Average time (seconds) for 1MB and 100 trials

Fang	JPSEC	Kiya	Kiya10	Dufaux	WuBlk	Zhu	bbcw16	bbcw128	bbcw256	AES
.0359	.0263	.0266	.0068	.0354	.0364	.0340	.1030	.0573	.0657	.0235

for block sizes from 16 byte to 256 byte. The performance increases with the block size up to 128 byte, than the effect reverses (these effects are highly implementation and hardware dependent). For a very long string of 1MB, i.e., $n = 2^{20}$ results are given in table 2. In summary block-based cycle-walking is well-performing.

7 Conclusion

An efficient format-compliant encryption approach for regular languages has been proposed, namely block-based cycle walking. The general approach has been implemented for the regular language of JPEG2000 bitstreams. The regular language of JPEG2000 bitstreams is dense enough that ranking is inefficient and sparse enough that cycle-walking is inefficient. Thus MP-secure encryption algorithms are practically infeasible. For JPEG2000-bitstreams, block-based cycle-walking greatly reduces the complexity compared to cycle-walking, from exponential time and linear space to linear time and constant space (in plaintext length). Compared to previous JPEG2000 encryption schemes it is simple and elegant (its reversibility is easily shown) and offers increased security.

References

1. Bellare, M., Ristenpart, T.: Format-preserving encryption. In: Proceedings of Selected Areas in Cryptography, SAC '09, Calgary, Canada, August 2009, Springer, Heidelberg (2009)
2. Black, J., Rogaway, P.: Ciphers with arbitrary finite domains. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 185–203. Springer, Heidelberg (2002)
3. Engel, D., Stütz, T., Uhl, A.: A survey on JPEG2000 encryption. Multimedia Systems 15(4), 243–270 (2009)
4. Fang, J., Sun, J.: Compliant encryption scheme for JPEG2000 image code streams. Journal of Electronic Imaging 15(4) (2006)
5. Goldreich, O.: The Foundations of Cryptography. Cambridge University Press, Cambridge (2001)
6. Hellwagner, H., Kuschnig, R., Stütz, T., Uhl, A.: Efficient in-network adaptation of encrypted H.264/SVC content. Elsevier Journal on Signal Processing: Image Communication 24(9), 740–758 (2009)
7. ISO/IEC 15444-1. Information technology – JPEG2000 image coding system, Part 1: Core coding system (December 2000)
8. ISO/IEC 15444-8. Information technology – JPEG2000 image coding system, Part 8: Secure JPEG2000 (April 2007)
9. Sipser, M.: Introduction to the Theory of Computation. International Thomson Publishing (1996)

10. Stütz, T., Uhl, A.: On format-compliant iterative encryption of JPEG2000. In: Proceedings of the Eighth IEEE International Symposium on Multimedia (ISM'06), San Diego, CA, USA, December 2006, pp. 985–990. IEEE Computer Society Press, Los Alamitos (2006)
11. Uhl, A., Pommer, A.: Image and Video Encryption. In: From Digital Rights Management to Secured Personal Communication. Advances in Information Security, vol. 15. Springer, Heidelberg (2005)
12. Wu, H., Ma, D.: Efficient and secure encryption schemes for JPEG2000. In: Proceedings of the 2004 International Conference on Acoustics, Speech and Signal Processing (ICASSP 2004), pp. 869–872 (May 2004)
13. Wu, Y., Deng, R.H.: Compliant encryption of JPEG2000 codestreams. In: Proceedings of the IEEE International Conference on Image Processing (ICIP'04), Singapure, October 2004. IEEE Signal Processing Society, Los Alamitos (2004)

Statistical Detection of Malicious PE-Executables for Fast Offline Analysis

Ronny Merkel, Tobias Hoppe, Christian Kraetzer, and Jana Dittmann

Otto-von-Guericke University of Magdeburg
ITI Research Group Multimedia and Security
Universitaetsplatz 2,
39106 Magdeburg, Germany
{ronny.merkel,tobias.hoppe,christian.kraetzer,jana.dittmann}
@iti.cs.uni-magdeburg.de

Abstract. While conventional malware detection approaches increasingly fail, modern heuristic strategies often perform dynamically, which is not possible in many applications due to related effort and the quantity of files.

Based on existing work from [1] and [2] we analyse an approach towards statistical malware detection of PE executables. One benefit is its simplicity (evaluating 23 static features with moderate resource constrains), so it might support the application on large file amounts, e.g. for network-operators or a posteriori analyses in archival systems. After identifying promising features and their typical values, a custom hypothesis-based classification model and a statistical classification approach using the WEKA machine learning tool [3] are generated and evaluated. The results of large-scale classifications are compared showing that the custom, hypothesis based approach performs better on the chosen setup than the general purpose statistical algorithms. Concluding, malicious samples often have special characteristics so existing malware-scanners can effectively be supported.

Keywords: static malware detection, adaptive classification, high scalability, comparison of hypothesis based and statistical classification.

1 Introduction and Motivation

Along with the capability of modern IT, also the spectrum of malicious software (malware) has grown more and more complex during the last decades. Current trends are the infiltration of secured connections at the client side (e.g. in online banking) or the extension to the so-called “Web 2.0”. Measures for detection (e.g. conventional signature-based approaches) and prevention are increasingly bypassed or disabled. An upcoming challenge is malware detection in vast number of files. Due to performance issues, new algorithms are required to provide fast offline analyses. As a preselection measure, these can support existing detection algorithms, which are potentially more precise but significantly more time-consuming. Conceivable application scenarios are malware scans by network operators (performing live in real-time), a posteriori analyses in archival systems (to scan for malware which was not detectable at archival time)

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 93–105, 2010.

or computer-forensic investigations that are increasingly performed after incidents and typically need to identify suspicious candidates from a vast number of files. Due to the static context of all these scenarios (the candidate files are not executed), utilising upcoming heuristic, behaviour-based approaches is difficult (since these usually require inspecting the dynamic execution of the code in an isolated environment and are more time-consuming). Therefore the optimal strategy for this scenario would be a heuristic approach, which limits itself to the evaluation of *static* features.

The aim of this work is the detection of malicious code based on statistical properties of executable files. On the example of the Windows PE format, a new prototype for statistical analyses is created, that extracts features of such files which can be obtained statically. This is extended by an adaptive, heuristical classification scheme which is hypothesis-based and can be adjusted for different application scenarios. We also test existing, statistical pattern recognition based classification algorithms to compare their accuracy. Also the combination with commercial products is determined. This way, an additional tool for malware classification is examined, which might also be useful to support and extend existing approaches.

In section 2, we introduce the relevant subset (Windows PE compliant files) considered in our investigations. Section 3 describes our new approach and introduces the selected features being evaluated. A hypothesis-based and a pattern recognition based classification scheme are explained. In section 4, the test setup is introduced and the results of the classifications in practical large-scale test are presented and compared. Section 5 concludes this paper and presents a short outlook.

2 Motivation of PE Executables as Chosen Code Representation

Today, the number of available representations of malicious code nearly equals the number of the various regular code forms in our modern IT environments. In general, code representations can be divided into natively executable code and interpreted code. Both can be categorised into different subclasses. Native code (also malicious) can appear in the form of self-sustaining programs or (not directly executable) code libraries. In the context of exploits, native malicious code can also be embedded into nearly arbitrary cover media. Also the wide spectrum of interpreted code ranging from simple textual script languages to complete programs in intermediate language notations being executed by interpreters or Just-In-Time-Compilers offers multifaceted ways to attackers to create malicious functionality. With respect to the context of web technologies, code representations can also be distinguished by means of their execution on the client or the server side. Especially in the context of the emerging "Web 2.0", the variety of code representations in the web technology domain is constantly growing. While the ideal solution for our practical implementation would have been a generic solution covering malicious code in all of these representations, we have to limit ourselves within this paper to a manageable subset of this wide spectrum. The most relevant reason is that some of these classes notably differ in their basic structure so the acquisition of common properties suitable for the classification would be very challenging. So we have to choose a compromise covering a relevant subset from the variety of code representations which can be evaluated comparably on a preferably wide range of common features. Also the *practical relevance* of

respective malicious code is an important requirement; for example a significant prevalence of respective malicious code is required in order to acquire sufficiently broad training and test sets.

Trading off these factors we first decided for the examination of native code representations having a very high prevalence and practical relevance today. Although interpreted representations have an increasing importance, these classes still have a much higher diversity which is the reason why we reserve them for future research. From the subclasses of native codes we decided for binary executables in Microsoft's Portable Executable (PE) format. One important reason is the high availability of respective reference code, since files in the PE format represent the biggest fraction of today's malicious code and also a sufficient number of non-malicious samples can be gathered without much effort as well, so the acquisition of appropriate training and test sets is realistic. Furthermore, the PE format is well documented [4] and relevant for programs and libraries on different kinds of systems (on PCs as well as on embedded systems like game consoles, PDAs, mobile phones or automotive devices).

3 Our Approach and the Selected Features

As stated in the introduction, the chosen approach is to perform a statistical evaluation of structural features of binary PE files. This heuristic approach on static features is expected to be a reasonable addition to former static approaches (since heuristic detection strategies today frequently rely on dynamic evaluations).

The approach is based on the assumption that many malicious code files contain structural indications which could lead to a successful identification in automated classifications. Some potential reasons we expect to cause such indications are:

- Structural changes performed by malicious code when bound to an existing (previously non-malicious) binary.
- Binaries built by malicious code generators having characteristic structural differences compared to programs built in commercial development environments (e.g. non-conformance to the PE specifications in certain issues which are not affecting its ability to be executed).
- General malicious code to which characteristic packers, encrypters or other obfuscations have been applied.
- Structural abnormalities caused by low-level programming techniques.
- Characteristic functional features of malicious code, e.g. the inclusion of critical combinations of API functions.
- Messages left by the malware authors (usually text strings).

Based on this assumption, our investigations and practical examinations follow a 4-step scheme:

Design of potential features: Based on existing work (e.g. in [5], [6] [7] and [8]), 44 potentially suited attributes A_i are identified, which can be extracted from PE files for malware classification purposes. To cover a wide spectrum of potential indications for the presence of malicious code, different domains of PE files have to be covered.

Therefore, characteristics from different domains[1] are selected:

- The outer file structure with several information from the main file headers (DOS header, PE headers), the inner file structure with the sections (information from their individual headers as well as the content within and between them).
- Some overall properties (like file entropy or contained string fragments) as well as functional aspects (e.g. scans for program flow redirection or statistically relevant combinations of imported functions).

In summary, 44 potential attributes are selected from these two domains. To remove unsuitable features, a 2-step feature reduction is performed on this basis (as results of this feature reduction step, the final features are introduced in section 4.2).

Modelling of measures about feature quality: As part of the theoretical concept, formulas are developed to describe the malware probability for each feature presentation as well as a quality measure for each feature. This is introduced in subsection 3.1.

Practical evaluation of feature qualities: Using a training set (see subsection 4.1 for details) we determine the relative occurrence of all feature presentations among non-malicious and malicious samples. The result is a statistic model describing the distributions of feature presentations, resulting quality measures as estimation for the discrimination of the features.

Classification concept and prototype evaluation: A prototype is implemented evaluating the selected attributes to decide for each sample if it is expected to be non-malicious or malicious. Based on two test sets (see subsection 4.1 for details), the prototype is evaluated and optimised with respect to the detection rates. The custom, hypothesis-based classification scheme is introduced in subsection 3.3 while the statistical pattern recognition based classification algorithms, which is used for comparison, is introduced in subsection 3.4..

3.1 Modelling a Measure for Feature Quality

Our formal model is based on the distribution of non-malicious and malicious programs in general, where P(s) describes the general frequency of malicious programs S and 1-P(s) the frequency of the non-malicious programs N. With respect to a given feature A_i and a certain feature presentation $a_{i,j}$ (a sample's value of feature j on attribute i), its probability of occurrence $P(a_{i,j})$ can be calculated from the measured occurrences $P_S(a_{i,j})$ and $P_N(a_{i,j})$ of the feature presentation among non-malicious and malicious code samples in training, respectively:

$$P(a_{i,j}) = P_S(a_{i,j}) * P(s) + P_N(a_{i,j}) * (1 - P(s)) \tag{1}$$

Whenever a certain feature presentation is found within a given sample, its conditional malware probability can be determined:

$$P(s \mid a_{i,j}) = \frac{P_S(a_{i,j}) \cdot P(s)}{P(a_{i,j})} \tag{2}$$

[1] Some of the available features can be heavily influenced by the use of runtime packers or crypters. However, as the evaluation (also see section 4) confirmed, most certain packers or crypters are preferably used for malicious (or vice-versa non-malicious) programs which is also respected by the approach.

Based on this, a discrimination measure for feature presentations $D(a_{i,j})$ is defined. It results from the absolute value of the difference between the conditional malware probability and the general probability of malware:

$$D(a_{i,j}) = | P(s \mid a_{i,j}) - P(s) | = | \frac{P_S(a_{i,j}) * P(s)}{P(a_{i,j})} - P(s) | \quad (3)$$

Where $P(s|a_{i,j})$ is the probability of s, given $a_{i,j}$.

Finally, the quality measure of features (attributes) $0 \leq Q(A_i) \leq 100$ is described by the proportionately added discrimination measures of its feature presentations multiplied by a normalisation factor of 100/P(s). Within our work, we exemplarily assume P(s) having a value of 0.5 (i.e. there is no bias toward one of the two classes), so the normalisation factor is 200:

$$Q(A_i) = 200 * [P(a_{i,1}) * D(a_{i,1}) + P(a_{i,2}) * D(a_{i,2}) + ... + P(a_{i,n}) * D(a_{i,n})] \quad (4)$$

A value of 0.5 has the advantage that the detection starts from a neutral position, so for each individual file there is no a-priory tendency in any direction. Also practical aspects might justify this choice, e.g. if previous measures like whitelists already reduced the amount of non-malicious test samples significantly. However, other values for P(s) are possible as well, only the normalisation factor in the formula given above would differ.

3.2 Choosing Attributes and Their Feature Presentations

For the demonstration of the feature quality assessment we use the number of PE sections as an exemplary attribute. The examination of their distribution among non-malicious and malicious software samples reveals, that most malicious code training samples have a number of 3 sections (47.2%) while this holds true for non-malicious training samples in only 10.9% of all cases. Figure 1 shows a graphical illustration of the distribution of the detected section numbers.

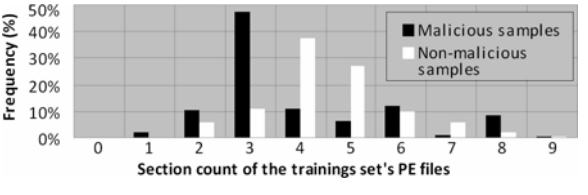


Fig. 1. Distribution of PE section count in training files

After the inspection of these values in the malicious and non-malicious samples from the training set, for each attribute certain cosets of feature presentations are defined. This is done in a way that the assignment of the evaluation subjects to these classes into non-malicious and malicious software is as discriminative as possible. With reference to the exemplary feature of the section count in the PE file, the optimal quality for three cosets is reached when the first feature presentation covers files with 0-3 sections (general occurrence probability 0.38), the second one files with 4-7

sections (general occurrence probability 0.55) and the third one files with 8 and more sections (general occurrence probability 0.29). These three feature presentations have malware probabilities $P(s|a_{i,j})$ of 0.78, 0.27 and 0.79 percent, respectively. The calculated quality measure for this feature is $Q(A_i)=50$.

3.3 Our Hypothesis Testing Based Classification Scheme

Respecting the class assignments (malicious / non-malicious) of the test samples and the contents of their extracted feature vectors, a hypothesis-based classification model is created. For this purpose, the distributions of the feature presentations are calculated to determine the malware probabilities $P(s|a_{i,j})$ introduced in subsection 3.1: for each attribute value the malware probability $P(s|a_{i,j})$ of its occurrence is determined (see subsection 3.2) from the training set, i.e. the statistical distributions obtained in training are used to test how characteristic each value is for malware.

For the classification of each file the 23 listed features are extracted. Since the real class assignments are known, these can be used to assess the classification's precision. To get stable (and sequence independent) classification results, the actual classification is performed as hypothesis testing, using the null hypothesis that the sample is non-malicious and the alternative hypothesis that the file is malicious. Out of several approaches discussed and compared in [1], the best performing approach uses a point system for weighting. If the malware probability $P(s|a_{i,j})$ for the determined feature presentation is above a threshold S_C of 60%, 1 point is granted for each additional exceedance by 1% (resulting in a maximal score of 40 points per attribute). To illustrate this by an example: During the analysis of the current sample file, the 3rd attribute ("number of sections", see subsection 3.2) is being evaluated. Given the file has two PE sections, its feature presentation is class 1 (0-3 sections). As explained in subsection 3.2, this feature presentation has a malware probability $P(s|a_{3,1})$ of 0.78. This malware probability exceeds the threshold S_C by 0.18, which means 18 points are added to the total score of the sample file.

This way, points are assigned and summed up for each of the 23 attributes of the current file, whereas a high point count indicates a high probability of malicious properties. The actual test is performed by the comparison of the total score with a threshold S_S. At this point, the assignment to the null- or the alternative hypothesis is done. The threshold value is freely adjustable and can be optimised for different application scenarios using the testing results (as shown in the results in subsection 4.3).

Continuing the example from above, after evaluating all 23 attributes the sample file might have been reached a total score of 78 points. If this exceeds the threshold S_S (e.g. 48), the file is finally classified as malicious.

3.4 The Statistical Pattern Recognition Based Classification Scheme

To compare and assess the results achieved with the hypothesis based classification scheme, we investigate the use of alternate classification solutions. This evaluation is performed by feeding the acquired feature vectors from the training and test samples as respective input for more common or established classification algorithms.

First, we estimate expectable results by having a closer look at the feature vectors of all samples from the training and both test sets, obtained using our prototype.

Every sample's feature vector contains 23 elements, each specifying the extracted feature presentation for the respective attribute. We determined the cardinal number of distinct feature vectors by removing duplicates. The results are presented and discussed in subsection 4.4.

For the actual classification tasks, we use the WEKA machine learning tool [3] and explicitly supply the training set and the test sets 1 and 2. From WEKA's broad range of classification algorithms, we select on the basis of initial tests: Simple Logistic [9] (regression models), Naïve Bayes [10], J48 [11] (decision tree), AdaBoostM1 [12], SMO (a multi-class SVM construct) and IB1 (1-nearest neighbour). All classifiers are evaluated with their default parameters, the results are presented in subsection 4.2.

4 Test Setup and Results

This section presents the test setup (subsection 4.1) and the results of the feature reduction (subsection 4.2) and both introduced classification approaches (subsections 4.3 and 4.4). Finally, an interpretation of the results is given in subsection 4.5.

4.1 Test Setup

A collection of around 7,200 malicious samples (provided by a project partner) and another collection of 15,000 non-malicious files (collected from different sources) were divided to yield two distinct sets for training and test. The training set consists of 5,387 malicious and 10,576 non-malicious samples at a total size of 2.18 GB while the test set contains 1,690 malicious and 4,334 non-malicious samples at a total size of 3.75 GB. To assess the generalisation of the results, another test set is added (referred to as test set 2) consisting of 10,302 malicious samples from a web collection.

4.2 Results of the Feature Reduction

During the Design of potential features (section 3), 44 considered features have been identified. This set of potential attributes has been reduced in two steps:

After this first, theoretic step, first practical evaluations have shown 6 of the considered features as inappropriate.

In the second step, the remaining 38 attributes were assessed as potentially characteristic features. The second, final reduction was performed after the *Practical evaluation of feature qualities (see section 3).* As introduced in subsection 3.1 and exemplified in subsection 3.2, the feature qualities $Q(A_i)$ have been determined for these 38 attributes. This was performed in a training phase, where PE files from the training set have undergone a feature extraction. As result of the feature reduction, all attributes with a determined feature Quality $Q(A_i) \geq 14$ are considered as useful for anomaly detection. Table 1 lists short descriptions of these 23 most characteristic features with their quality values obtained during the evaluation on the training set (see [1] for a more extensive documentation). For most attributes 2 or 3 feature presentations are defined. In some cases 4 and (in 1 case) 10 of such cosets have shown to be the most promising configuration. These 23 features have been chosen for the classification tests with the heuristic prototype and the two introduced classification schemes.

Table 1. Attributes chosen for anomaly detection with their feature qualities $Q(A_i)$

Attribute	Q	Attribute	Q
1 DOSHeader/StandardValues	19	13 OptionalHeader/Checksum	53
2 DOSHeader/lfanew	35	14 OptionalHeader/NumberOfDataDirectories	14
3 FileHeader/NumberOfSections	50	15 OptionalHeader/DataDirectory/ImportTable	36
4 FileHeader/PointerToSymbolTable	19	16 SectionHeader/Name	70
5 FileHeader/SizeOfOptionalHeader	14	17 SectionHeader/PointerToRawData	14
6 OptionalHeader/SizeOfCode	68	18 SectionHeader/PointerToRelocations, lineNos.	14
7 OptionalHeader/SizeOfInitialisedData	46	19 SectionHeader/Characteristics	67
8 OptionalHeader/SizeOfUnitialisedData	21	20 Sections/CodeRedirection	15
9 OptionalHeader/AddressOfEntryPoint	**72**	21 ImportTable	55
10 OptionalHeader/BaseOfCode&-Data	61	22 FileStringSearch	60
11 OptionalHeader/BaseOfCode	53	23 FileEntropy	67
12 OptionalHeader/MajorImageVersion	61		

4.3 Results from the Hypothesis Testing Based Classification

Table 2 lists the detection and error rates at different thresholds. For a balanced compromise between the detection rates on both test sets, an optimal threshold of S_S=43 is identified, where the best average detection rate among both test sets (85% and 81%, respectively) is achieved. The false negatives rate (FNR) at this threshold is 1% for test set 1 and 18% for test set 2 with a false positives rate (FPR) of 14% for test set 1 (remarks: Here, FPR and FNR cover all possible errors, therefore detection rate + FPR + FNR =100%, even when test set 2s FPR must be equal to 0 in all cases since it only consists of malicious samples). For the files in both test sets, Figure 2 illustrates the distribution of the total point scores of all samples. For each possible point score (max score of 920 points = 23 attributes * 40 max. points). Malicious samples of both test sets mainly occur in the right part while most non-malicious samples are present in the left part. A problematic range can be identified at total scores between 27 and 63 points, where a significant overlap of non-malicious and malicious samples exists.

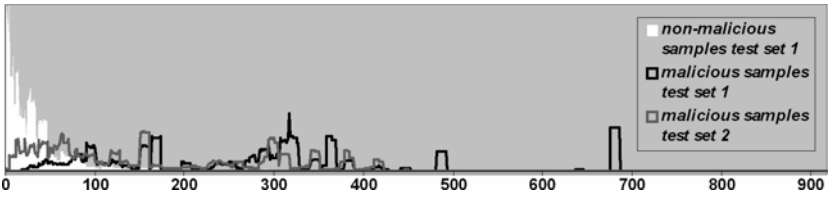


Fig. 2. Distribution of total scores in the test sets (smoothed)

The adjustable threshold allows configuring the prototype with respect to its detection performance. Table 2 shows the detection rates for 7 exemplary threshold values together with the respective FNR and FPR (where available) values. Depending on the focus of the classification, the precision can be freely tuned. For example, by using a low threshold it is possible to achieve a high detection rate of malicious samples accepting a higher FPR. Or the prototype can be configured to achieve a low number of false alarms (using higher thresholds). This flexibility allows a free

Table 2. Detection and error rates for test sets 1 and 2

Exemplary threshold	Test set 1 Detection rate	FNR	FPR	Test set 2 Detection rate	FNR
$S_S = 10$	47.5%	**0.02%**	52.4%	**99.7%**	**0.3%**
$S_S = 17$	58.4%	0.03%	41.6%	96.6%	3.4%
$S_S = 27$	66.8%	0.1%	33.1%	92.3%	7.7%
$S_S = 43$	85.0%	0.7%	14.3%	81.6%	18.4%
$S_S = 63$	88.4%	1.6%	9.99%	73.7%	26.3%
$S_S = 84$	**92.5%**	2.7%	4.8%	63.1%	36.9%
$S_S = 156$	91.2%	7.9%	**0.9%**	40.4%	59.6%

customisation of the prototype for a given application scenario. For an end user protection system, a low FPR might be intended while in a high security environment or a pre-selection process for further scanning a low FNR should be more important.

The described malware classification approach can be utilised to support existing mechanisms. We analysed this ability with reference to the two antivirus products *Avira AntiVir Personal* and *Kaspersky Anti-Virus*. First we scan our test sets with these products to compare their results with the result of our heuristic, hypothesis-based approach. We furthermore analyse the capability of our approach to extend such existing scanning tools. Table 3 shows the results of both commercial scanners using all signatures available at the time of the test (January 2009).

Table 3. Classification accuracies of two commercial scanners

		Avira AntiVir	Kaspersky
Test set 1	malicious samples	83,4%	**84,0%**
	non-malicious samples	**100,0%**	99,7%
Test set 2	malicious samples	97,5%	**99,9%**

As common for signature based methods, the tested antivirus products have a FPR close to zero. While this is very satisfying, the FNR for test set 1 is relatively high, i.e. a lot of malicious samples are not detected. One interesting aspect of these results is that the hypothesis-based classification approach performs quite well on these files because obfuscation techniques have been applied to many of the malware samples from test set 1 (taken from the wild), leading to characteristic changes. On the other hand it has difficulties on test set 2 containing relatively clean malware samples from a web collection. A combination of both approaches might therefore be reasonable, which we analysed by feeding malware not detected by the commercial scanners to our prototype (using the hypothesis-based classification strategy).

Table 4. Results on malicious files not detected using signatures

	Test set	Undetected malware (signature based)	Detection rates (prototype) on these files
Avira	1	281	**93,6%**
	2	254	78,7%
Kaspersky	1	**270**	92,6%
	2	**8**	**100,0%**

The results in Table 4 show, that the subsequent addition of our prototype can achieve a clear increase of the detection rates up to 15.5% on the cost of an FPR of 15% (balanced threshold of S_S=43). In scenarios where only an FPR of 1% is acceptable (S_S=156), still an increase of the detection rate by 9.4% would be possible. In scenarios like a high-performance pre-selection of potential malware for more exact but also more expensive scans (which rather require a low FNR), a hybrid implementation of the evaluated signature based scanners and our heuristic prototype could be applied as well. E.g., at the thresholds of S_S=43 and S_S=10, at least 99% (99.6%) of all malicious samples would be classified correctly at the cost of an FPR of 15% (52%). In principle, the order of application could also be inverted due to the commutativity of both, independent approaches. While the overall FPR remains the same (in the worst case it can be expected to be the sum of both individual FPRs when both systems detect a distinct set of false positives), it might be better to apply the faster scanner first.

4.4 Results from the Statistical Pattern Recognition

As described in subsection 4.2 we first remove all duplicates from the set of all feature vectors because these are redundant for the statistical pattern recognition. This paragraph presents the results from the analysis of the distinct feature vector sets. From the training set (15,963 samples) 1,241 distinct feature vectors have been retrieved. The feature vectors obtained from test set 1 (6,024 samples) and test set 2 (10,302 samples) could be reduced to 796 and 1,460 distinct feature vectors, respectively. Within the classification of the first test set, 546 distinct feature vectors appear that are already known from training, while the other 250 feature vectors have not been observed before. As a first pre-estimation, around 546/796 feature vectors (68.6%) should be classified correctly. Since we face a 2 class problem, about 125 of the 250 unknown feature vectors can be expected to be classified correctly (right guesses), so a total classification accuracy of ca. 84.3% (671/796) can be expected. For test set 2 - having far more unknown feature vectors (1,224) than ones known from training (236) - the estimated classification accuracy can equivalently be determined around 58.1% (848/1460).

Table 5 shows the results of the tests with WEKA's classifiers. The values contained in the second column (error on training data) describe the efficiency of the model generation using the respective algorithm. They can be interpreted as the maximum precision which can be expected as achievable detection rates within real tests. The last two columns contain their classification accuracies as measured in the tests on test set 1 and 2, respectively.

Table 5. Classification accuracies of the statistical classifiers

Classifier	Training set	Test set 1	Test set 2
SimpleLogistic	85.64%	83.37%	62.74%
NaiveBayes	81.13%	80.64%	64.11%
J48	**88.92%**	85.27%	**67.40%**
AdaBoostM1	83.88%	83.73%	49.79%
SMO	84.95%	82.42%	61.78%
IB1	88.62%	**87.53%**	50.55%

The results of the tests with the statistical pattern recognition strategies (Table 5) show that these do not achieve a significantly better precision in malicious code detection compared to the hypothesis testing based approach (Table 2). On test set 1, the hypothesis testig based approach achieves a detection accuracy of 85.0% (balanced mode; S_S=43). On this test set, all except 2 established algorithms achieve worse results: One exception is the J48 algorithm which is performing slightly better (0.27%), the other one is IB1 achieving a gain of 2.53%. However, all tested established algorithms have clear drawbacks on test set 2: The achieved results are between 14.2% and 33.8% below the results of the hypothesis-based approach (81.6%).

Out of the classification algorithms tested from WEKA, the J48 achieved the best general performance on our malware detection task. An exemplarily performed attribute selection on this classifier using WEKAs attribute selection mechanisms (here CfsSubsetEval evaluation using BestFirst search with default parameters - for details see [3]) showed that nine out of the 23 attributes are responsible for more than 94% of the classification accuracy achieved (see Table 6). The attributes selected as being useful are feature no. 2, 3, 6, 9, 10, 12, 13, 22, 23 (see Table 1 for their short descriptions and their calculated quality values). These results closely match the results obtained by using our quality measure $Q(A_i)$.

Table 6. J48 classification accuracy using attribute selection

Classifier	Training set	Test set 1	Test set 2
J48 with 23 attributes	88.92%	85.27%	67.40%
J48 with 9 attributes	88.92%	84.68%	63.36%

4.5 Interpretation of the Results

The tests done with the hypothesis based classification strategy (subsections 3.3/3.4) show that the classification accuracy is clearly lower on test set 2. This is also evident in the tests with the statistical classifiers (subsections 3.4/4.4) to an even greater extent. A reason might be that the malicious code from test set 1 mostly contains obfuscated in-the-wild samples provided by a project partner, while test set 2 contains less obfuscated, relatively clean malware samples from an internet collection. In general, while our heuristic approach on *static* features achieves high detection rates at above 90%, this accompanies high FPRs (see Table 2).

In comparison, the manually developed, hypothesis testing based classification strategy has a better overall performance compared with the statistical pattern recognition approach. One important reason for the better performance might be that our hypothesis-based approach better takes care of the distribution of certain feature presentations amongst non-malicious and malicious samples. It respects this knowledge provided by $P(s|a_{i,j})$ when assigning the penalty points. Since the statistical classification algorithms (subsections 3.4/4.4) only get input in form of distinct, raw feature vectors and the respective class labels, these can not directly utilise such additional information.

5 Summary and Outlook

In this paper, we present results of a work on new approaches for malware detection in static context. Based on the identification and evaluation of significant features within PE files, a prototypical implementation has been created for feature acquisition and sample classification.

The three major results of this paper are:

1. A new feature extractor for statistical analyses on PE files.
2. A new application scenario adapted classification approach based on hypothesis testing, which allows by threshold adjusting for an optimisation of false positive and false negative rates.
3. The evaluation of the feature extractor and the classifier on large test sets, leading to very good results, e.g. as a promising combination with commercial antivirus products.
4. A direct comparison with statistical pattern recognition based classifications shows that the hypothesis based scheme achieves a better overall accuracy.

One major advantage of the newly introduced classification approach is that it is scaling much better than any signature based approach or statistical pattern recognition approach ever could. This is due to the fact that in the test phase of the classification task (i.e. the deployment of the system) for each sample to be checked only one comparison against the threshold S_S has to be made, in contrast to a check against the complete signature data base.

In future work, especially the appropriateness for generalisation of the test sets acquired for the practical evaluation could be analysed more intensely to further substantiate the potential and restrictions of the approach. Also the extension to other code representations, especially in the context of Web 2.0 technologies is a promising topic for future research.

References

1. Merkel, R.: Statistische Merkmale zur Anomaliedetektion in ausführbaren Dateien. Diploma thesis, Otto-von-Guericke-University of Magdeburg (2009)
2. Hoppe, T., Merkel, R., Krätzer, C., Dittmann, J.: Statistische Schadcodedetektion in ausführbaren Dateien. In: D-A-CH Security 2009, Syssec (2009)
3. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
4. Microsoft Corporation: Microsoft Portable Executable and Common Object File Format Specification, Revision 8.1 (2008)
5. Cohen, F.: Computer Viruses - Theory and Experiments, publication for PhD thesis (1984), `http://all.net/books/virus/index.html` (last access March 2010)
6. Szor, P.: The Art of Computer Virus Research and Defense. Symantec Press (2005)
7. Skoudis, E., Zeltser, L.: Malware: Fighting Malicious Code, 2nd edn. Prentice-Hall, Englewood Cliffs (2004)
8. Treadwell, S., Zhou, M.: A heuristic approach for detection of obfuscated malware. In: Proceedings of the 2009 IEEE ISI, Richardson, Texas, USA, June 08-11, pp. 291–299 (2009)

9. Landwehr, N., Hall, M., Frank, E.: Logistic Model Trees. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) ECML 2003. LNCS (LNAI), vol. 2837, pp. 241–252. Springer, Heidelberg (2003)
10. Borgelt, C., Timm, H., Kruse, R.: Probabilistic networks and fuzzy clustering as generalizations of naïve bayes classifiers. In: Reusch, B., Temme, K.-H. (eds.) Computational Intelligence in Theory and Practice, Heidelberg, Germany (2001)
11. Quinlan, R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo (1993)
12. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Proc International Conference on Machine Learning. Morgan Kaufmann, San Francisco (1996)

A Frame of Reference for Research of Integrated Governance, Risk and Compliance (GRC)

Nicolas Racz[1], Edgar Weippl[1], and Andreas Seufert[2]

[1] TU Vienna, Institute for Software Technology and Interactive Systems, Favoritenstr. 9-11, 1040 Vienna, Austria
{racz,eweippl}@ifs.tuwien.ac.at
[2] Steinbeis Hochschule Berlin, Institut für Business Intelligence, Gürtelstr. 29A/30, 10247 Berlin, Germany
Andreas.Seufert@i-bi.de

Abstract. Governance, Risk and Compliance (GRC) is an emerging topic in the business and information technology world. However to this day the concept behind the acronym has neither been adequately researched, nor is there a common understanding among professionals. The research at hand provides a frame of reference for research of integrated GRC that was derived from the first scientifically grounded definition of the term. By means of a literature review the authors merge observations, an analysis of existing definitions and results from prior surveys in the derivation of a single-phrase definition. The definition is evaluated and improved through a survey among GRC professionals. Finally a frame of reference for GRC research is constructed.

Keywords: governance, risk, compliance, GRC, integrated, definition.

1 Introduction and Motivation

The acronym "GRC" (governance, risk and compliance) has rapidly penetrated the business community over the last years. It has made its way into software labels, marketing slides and department names in global enterprises. In the early days of GRC, PricewaterhouseCoopers [1] noted: "In itself GRC is not new. As individual issues, governance, risk management and compliance have always been fundamental concerns of business and its leaders. What is new is an emerging perception of GRC as an integrated set of concepts that, when applied holistically within an organisation, can add significant value and provide competitive advantage." In the authors' opinion this emerging perception – contrary to the acronym itself – is not well-established. In business as well as in research the awareness of the concept of integrated GRC is rather low. People are struggling to describe the idea behind the term. "Definitions of GRC are as varied as they are fluid" [2] to a degree that it was even recommended to avoid definitional debates [3]. This is partly owed to the lack of a scientifically grounded definition; instead software vendors and consultants publish definitions that suit their products and services. We could throw GRC into the corner of buzzing acronyms if market reports and surveys were not attributing a growing importance of GRC in the future [4] – and an already strong impact today. In 2008 about 40 billion

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 106–117, 2010.

US-dollars were spent on services, technology and content related to GRC [5]. The business network LinkedIn lists close to 4,000 GRC professionals. Do they really work in a blurred, intangible domain? Researchers and professionals in IS security view GRC as a means to draw the attention of management to information security and to make its benefits understandable and tangible for business people.

This research was carried out in order to develop a frame of reference that supports GRC research in general and the creation of reference models for integrated GRC according to the process model for an empirically grounded reference model construction [6] in specific. The frame construction goes hand in hand with the development of a single-phrase definition of GRC. Both items may be used as a starting point by researchers when approaching the topic in a structured, scientific manner. Our paper will help to shed light on what we talk about when we talk about GRC. After all we do not forever want to treat GRC "like a large black box: a mysterious container full of improved processes and software for automation" [7].

2 Research Methodology

The methodology applied to carry out this research consists of four stages.

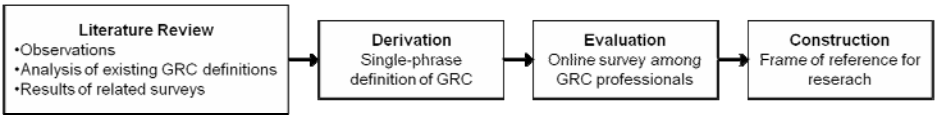


Fig. 1. Research methodology

The first stage is a review of GRC publications following the classifications of Fettke [8] for reviews in business informatics. Its properties are as follows:

Property		Category	Application
1. Type		natural language and mathematical-statistical	quantitative in observations, qualitative in definitions analysis
2. Focus		theory	GRC definitions are theoritical concepts
3. Target	Formulation	explicit	see "motivation" section
	Content	integration	integration of publication content in a definition for research
4. Perspective		neutral	no leading hypothesis defined in advance
5. Literature	Selection	explicit	see below in "methodology" section
	Scope	selective	keywords, sufficient length, degree of product-independent information, text-based format
6. Structure		holistic by topic	identification of commonalities
7. Target group		practicioners, general and specialised researchers	primarily directed towards researchers
8. Future research		explicit	see "future research" section

Fig. 2. Publication review properties (based on Fettke 2006)

Immediately striking a reader of GRC-related publications is the massive number of topics mentioned. Numerous methodologies such as business rules management, business process management, or enterprise content management are as present as processes such as auditing, planning and control. Seen as separate topics, corporate governance, risk management and compliance alone are vast areas impossible to grasp in a single literature review. Since we wanted to identify the meaning of GRC as a whole and not that of its fragments alone, we restricted the review to publications that explicitly mention all three topics as in "governance, risk (management) and compliance" or "GRC".

Publications were found using the search engines of WISO, EBSCO, ACM, IEEE Xplore, SpringerLink, Emerald, Google, the Vienna University of Technology's and Ludwigshafen University of Applied Sciences' libraries, and through manual browsing on relevant websites. From the findings only those results were chosen that fulfilled the three criteria of sufficient length, sufficient degree of product-independent information and text-based format. Eventually 107 sources published between 2004 and 2009 made it to the final list. They were analysed using mathematical-statistical and natural-language methods. The exact methodology applied is case-specific for each observation. It is therefore presented later in this document together with the respective observations.

In a second stage the observations, the analysis of existing definitions and the results of two related surveys were used in the derivation of a single-phrase working definition of integrated GRC.

In a third stage an anonymous online survey was conducted to evaluate and improve the working definition. We posted the survey in GRC expert groups of the business networks XING and LinkedIn. Eventually 131 GRC professionals took part. They responded to four questions: (i) a rating of the definition on a scale from 10 (best) to 1 (worst) with the option to refuse a ranking if they felt that a single-phrase definition of GRC generally would not make sense; (ii) an optional free text comment to provide feedback; (iii) the type of organisation the respondent is working for and (iv) the respondent's GRC focus.

The participants constitute a cross-section of GRC professionals. 42% work in GRC consulting, 18% for GRC software vendors, 16% focus on GRC in their own organisation, 11% are auditors and 5% each work for research institutions or as freelancers. 4% work for other types of organisations. Participants' primary interests in GRC are GRC processes without technology focus (29%) followed by GRC technology (26%), compliance (19%), risk management (18%), and corporate governance (4%). The remaining 5% do not primarily focus on any of these topics.

The fourth stage of the research project is the construction of a frame of reference for research of integrated GRC based on the short-definition. Following the process model for empirically grounded reference model construction [6], the frame is a condensed high-level abstraction of a future reference model, created to support navigation within the problem domain of GRC. As proposed by Schlagheck [9] the frame of reference is developed early in parallel with the problem definition helping to scope GRC modelling and other research projects, to identify single model elements and to guarantee completeness.

3 Literature Review Results

The results of the literature review – key observations, the analysis of definitions and prior surveys – will be described in the following.

3.1 Literature Review – Key Observations

O1: There is basically no scientific research on GRC as an integrated concept. While lots of research exists on the "G", the "R", and the "C" as separate topics, the potential integration moves under the radar of scientific research. Of the 107 sources identified a mere two deserve the label "research paper" [10, 11]. Both publications only provide short definitions of governance, risk management and compliance separately. O1 demonstrates the lack of research participation in GRC.

O2: Software vendors, analysts and consultancies are the main GRC publishers. We categorised our sources by authorship, distinguishing software vendors, analysts, consultancies, scientific research personnel and independent experts. Co-authorship was applied in four cases. For interviews only the role of the interviewee was considered. The review shows that GRC software vendors are the most active group providing GRC publications (40), closely followed by analysts (34) and consultancies (31). Together these three parties participated in 94% of the selected GRC publications. GRC is obviously dominated and driven by the business community.

O3: Software technology is the prevailing primary topic. When publications are dominated by software vendors followed by consultancies that help implementing technologies, it is not surprising that software technology is the prevailing topic in these works. 57 publications (53%) primarily treat technology. This finding underlines the importance of technology as an enabler of GRC.

O4: Regulatory compliance is the main driver of GRC, challenged by risk management. We listed all reasons explicitly named as GRC drivers in publications. 43 out of 107 publications do not mention any GRC drivers. Of the remaining 64, 25 (39%) consider the increasing number of regulations to drive GRC. 18 (28%) name increased risk, 10 (16%) the potential for cost reductions, 8 (12,5%) mention the increased complexity of business due to market dynamics, globalisation and other factors. According to a study of AMR Research, risk management is about to surpass compliance as top GRC priority. "No longer just a U.S.-centric concern tied to compliance with 2002's Sarbanes-Oxley Act and other specific regulations, GRC has evolved into a set of practices to manage and mitigate the full array of risks organizations face" [12]. Comparing the drivers mentioned in 2007 and in 2008, we found that our review did not significantly support the AMR findings. References to risk as a driver hardly changed (23,5% in 2007; 24% in 2008), while the emphasis of regulations declined from 41% to 34%.

O5: ERM is an important methodology within GRC. Inspired by the article "Is ERM GRC? Or Vice Versa?" [13] we wanted to find out how often enterprise risk management (ERM) or its synonyms were mentioned in GRC publications. References to ordinary risk management were not accounted for. 58 publications (54%) mentioned ERM. The enterprise-wide perspective of risk seems to go hand-in-hand with GRC.

3.2 Literature Review – GRC Definitions

One in three of the analysed publications offers a GRC definition. Two thirds of these definitions explain what is understood by GRC as an integrated concept. The remaining third disregards that the total might be more than the sum of its parts and confines itself to defining the three terms of governance, risk management and compliance separately.

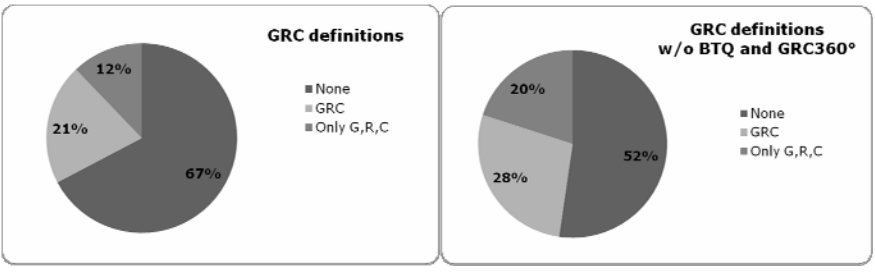


Fig. 3. GRC definitions in publications

Omitting the two journals steadily publishing GRC articles directed towards readers familiar with the term – "Business Trends Quarterly" (BTQ) and "GRC360°" – the percentage of GRC definitions rises to almost fifty percent. References to definitions made before are basically nonexistent; sometimes several different definitions are provided by a single organisation.

A separate definition of the three terms of governance, risk management and compliance is made in 12% of the publications and in 20% when leaving away BTQ and GRC360°. The exact meaning of the topics themselves is a study of its own and cannot be discussed in this document. According to [14] it might not even be purposeful: "To be clear, there are substantially more processes than governance, risk and compliance playing critical roles in GRC. But 13-letter acronyms rarely catch on." Still some of the authors follow the "G,R,C approach" in their definitions [15, 16, 17]. However the larger percentage of comprehensive GRC definitions in publications lets us conclude that the idea of an integrated concept is more widely supported. GRC is more than an umbrella term for governance, risk and compliance.

Looking at definitions of the integrated concept, some authors hold a technology-oriented view. Banham [13] cites a consultant stating that in contrast to ERM "GRC is more a technology platform for illuminating governance and compliance risk. It's useful to think about GRC in terms of an IT platform. [...] The technology helps you centralize and organize your policies, procedures, documentation requirements, risk assessment analyses and other content [for] dashboard reporting."

On the contrary KPMG [18] insists that "[GRC] is more than a software solution; it is a strategic discipline. GRC is a continuous process that is embedded into the culture of an organization and governs how management identifies and protects against relevant risks, monitors and evaluates the effectiveness of internal controls, and responds and improves operations based on learned insights." This view of GRC as an enterprise-wide management concept is supported by [1, 19, 20]. Corporate Integrity

[21] goes as far as calling GRC a "philosophy of business" that "permeates the organization: its oversight, its processes, its culture." Mitchell [10] speaks of "principled performance", which is picked up by Hovis [22]: "Integrated GRC is a cross-functional and extended enterprise capability that, when implemented, creates 'principled performance.' An integrated GRC effort is a transforming initiative, affecting how the enterprise will function both in its strategic orientation and in its operational focus."

The Open Compliance & Ethics Group [23] published an exhaustive definition that was reviewed by professionals from a variety of organisations: "GRC is a system of people, processes and technology that enables an organization to understand and prioritize stakeholder expectations; set business objectives congruent with values and risks; achieve objectives while optimizing risk profile and protecting value; operate within legal, contractual, internal, social and ethical boundaries; provide relevant, reliable and timely information to appropriate stakeholders; and enable the measurement of the performance and effectiveness of the system."

Switzer [16] emphasises integration: "We like to use the three letter term 'G-r-C' as a symbol for the need to integrate these efforts with each other and within business operations." Process-oriented perspectives emphasising improvements through integrated GRC are taken by [24] and [25], who describe GRC as a set of "initiatives [...] which look across [...] risk and control functions holistically and seek to enhance their efficiency and effectiveness."

From these definitions we concluded that (i) GRC is an integrated, holistic management concept for the topics involved, (ii) that technology is a key – but GRC is more than just technology, and (iii) that integrated GRC is supposed to improve the performance of processes.

3.3 Literature Review – Previous Surveys of the Understanding of GRC

The opinion of GRC professionals has previously been identified by two surveys. The first survey of over 400 organisations led to the following result: "The vast majority of respondents (75%) view GRC as 'a coordinated program involving people, processes and technology.' More than half (54%) viewed GRC as a valuable concept, representing the future of how GRC concepts will be addressed. Almost all respondents view GRC as a process rather than a product or a fad (only 3%) [...]." [4]. This shows a more deliberate idea of GRC than the results gathered by Approva [26] one year earlier. In this survey, 87.1% of over 200 respondents consider GRC a "term used to describe a group of internal policies & processes designed to manage risk", while hardly anybody opted that GRC was "just another acronym" (3.3%), the "name of a software category" (2.4%) or the "name of a functional department in my company" (3.3%). Only 3.8% of respondents were unfamiliar with the term.

4 Derivation and Evaluation of a GRC Working Definition

The multitude of GRC definitions makes it difficult to find a consensus; to a certain extent the definitions overlap, but some treat aspects that are disregarded in others. For our definition the 75% majority of the Kahn survey claiming that people, processes and technology are involved was taken as a starting point. Furthermore the

concept of "integrated" GRC, after ruling out the fragmented approach above, was followed. Incorporating the observations and the three conclusions drawn from the definitions analysis – the integrated, holistic management concept, technology being a key (but not the only one), and GRC being supposed to improve the performance of processes – we derived the following preliminary single-phrase definition: *'GRC is an integrated, holistic approach to corporate governance, risk and compliance ensuring that an organisation acts in accordance with its self-imposed rules, its risk appetite and external regulations through the alignment of strategy, processes, technology and people, thereby leveraging synergies and driving performance.'*

The survey conducted in order to validate and improve the definition brought about interesting results. Only three out of 131 respondents opted to answer "no rating – I think there should not be a one-sentence definition of GRC". The other 128 participants attributed the definition an average of 7.5 on a 10-point-scale. 78% rated it 7 or higher. Only 12% chose a rating of four or lower – the same percentage of respondents that supported the definition unconditionally, awarding a ten point rating.

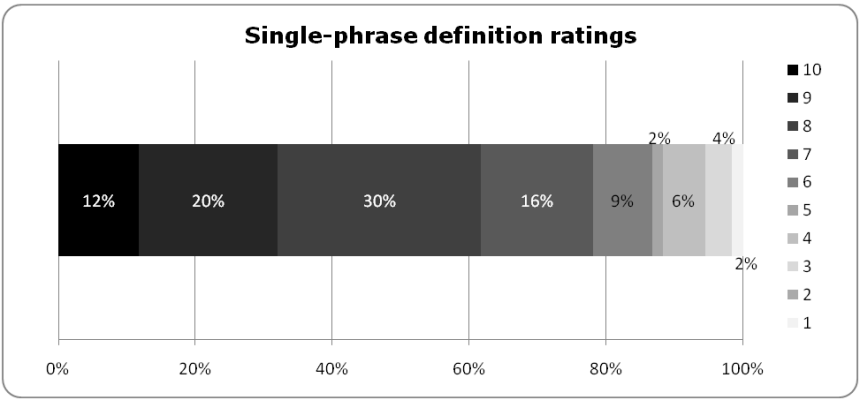


Fig. 4. Distribution of ratings of the single-phrase GRC definition

We interpret the result as a strong backing of our definition. Still we looked at the 74 comments provided by participants in order to introduce minor improvements. 18 respondents criticised that the definition was overly long and complex. 13 and 8 respondents, respectively, did not like the wording "leveraging synergies" or "driving performance". We replaced the terms with "improving efficiency and effectiveness", which includes the use of synergies and improved performance but is more general. "Self-imposed rules" was criticised as being clumsy; we replaced it with "internal policies". Several respondents asked for ethics to be included as companies such as Enron and Worldcom were fully compliant but still went bankrupt due to unethical actions. "Corporate" was replaced with "organisation-wide" as the former could imply a restriction of GRC to the C-level of a company. Lastly we moved "risk appetite" in front of "internal policies and external regulations" because participants felt the definition was too compliance-centric. The final definition is as follows:

'GRC is an integrated, holistic approach to organisation-wide governance, risk and compliance ensuring that an organisation acts ethically correct and in

accordance with its risk appetite, internal policies and external regulations through the alignment of strategy, processes, technology and people, thereby improving efficiency and effectiveness.'

5 Construction of a Frame of Reference for Integrated GRC

The definition was incorporated into a high-level frame of reference highlighting the key elements that should be examined when researching the integrated GRC concept.

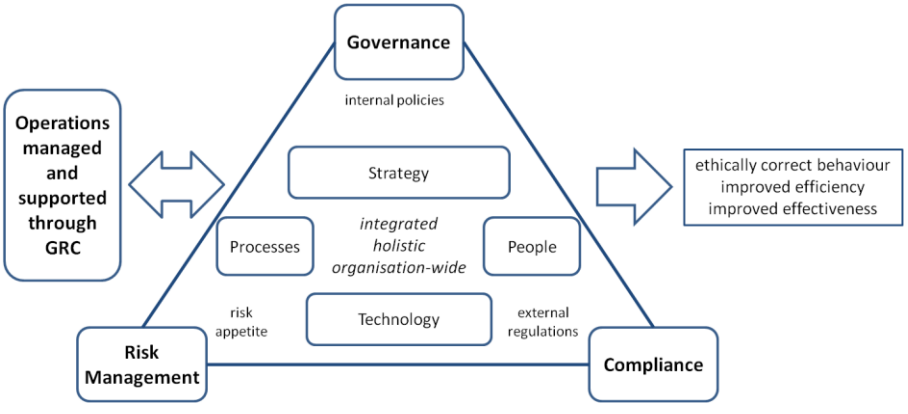


Fig. 5. Frame of reference for integrated GRC

Governance, Risk Management and Compliance are the core *subjects* of GRC. Each of the subjects consists of the four basic *components* of GRC: strategy, processes, technology and people. The organisation's risk appetite, its internal policies and external regulations constitute the *rules* of GRC. The subjects, their components and rules are now to be merged in an integrated, holistic and organisation-wide (the three main *characteristics* of GRC) manner – aligned with the (business) operations that are managed and supported through GRC. In applying this approach, organisations long to achieve the *objectives* of GRC: ethically correct behaviour, and improved efficiency and effectiveness of any of the elements involved.

Of course the components strategy, processes, people and technology are not exclusive to GRC. All operations of an organisation are constituted by these components. For the procure-to-pay cycle, for example, there is a strategy that sets and controls targets; there are the process steps from procurement to payment, and procurement staff as well as transactional and information systems enabling the cycle. GRC supports the management and the execution of these operations; e.g. through governance specifications for the handling of goods, segregation of duties across the procure-to-pay processes, or technology to monitor risks in the supply chain.

For information systems research another sub-category of GRC is of special interest: GRC processes that support the information technology operations of an organisation [27]. These GRC processes are commonly referred to as "IT GRC" [28].

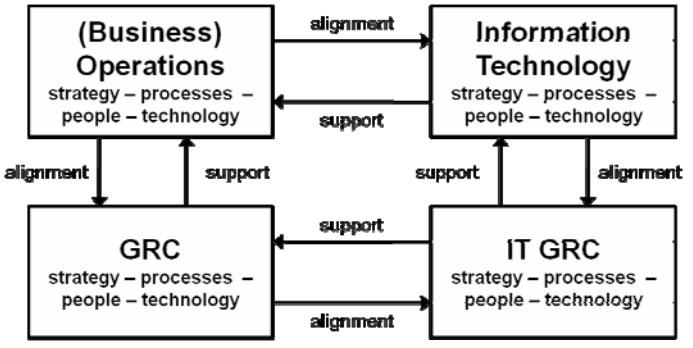


Fig. 6. GRC and IT GRC in the business and IT context

IT GRC deals primarily with issues of information security, IT compliance, IT and data governance, IT risk management and IT revision [29]. It is aligned with the overall GRC activities, the IT operations and indirectly with the organisation's (business) operations.

A universal analysis of GRC would have to consider all the components of the above figure. Analysis – the separation of a whole into its component parts – helps a researcher to focus on certain aspects. For example a research project could be restricted to examining IT GRC technology, such as IT security software and systems monitoring tools. A more comprehensive project might include the whole of IT GRC and its integration with the IT components. A researcher who does not want to dive into the depths of technology might focus on the integration of GRC processes with a specific business process. Sometimes it is difficult to draw a clear line between the four boxes; there are even intentional overlaps. For instance in most cases GRC technology is information technology. Depending on the perspective of the researcher, classifications can be made as it suits the research project best. For scoping it is just important that relevant components are not left away.

Once the components in scope have been chosen, the same can be done for the rules that are to be considered. The rules of GRC are basically defined by compliance requirements, the risk management process and the organisation's governance codices. No matter if they are stated in regulations, internal policies or target agreements, in the end they are all normative or restrictive instructions that may potentially be represented and used in an integrated manner. The large number of rules might require a researcher to focus on certain rules and leave others away (e.g. include the COBIT framework but ignore ISO 27001). In any case it should be examined in how far the GRC characteristics (integrated, holistic, organisation-wide) are present in the subject of research.

Eventually GRC research should investigate the impact of integrated GRC in their models or subjects of research; is there an improvement in the objectives of ethically correct behaviour, efficiency and effectiveness? Effects may arise in any of the GRC subjects, all operations, IT, GRC and IT-GRC subcomponents, and in the handling of the GRC rules.

A short example will help to understand how this frame of reference supports scoping and approaching a GRC research project. Assuming a researcher wants to examine an organisation's GRC approach and its effects on the procure-to-pay cycle, excluding IT-GRC. The researcher needs to consider the following points:

- Is the organisation's approach to governance, risk management and compliance integrated, holistic and organisation-wide across the four components of strategy, processes, people and technology?
- What does the procure-to-pay cycle look like across the four components?
- Which rules affect the procure-to-pay cycle? Which of these rules need to be considered in the research project? Does the organisation treat these rules in an integrated, holistic and organisation-wide manner?
- Do the GRC specific components interact with their "general" counterparts? E.g. does the GRC strategy influence the setting of targets for the order-to-cash cycle? Are automated controls implemented in the order-to-cash application and are they linked to GRC systems?
- Are the objectives of GRC realised? Is adherence to the rules in the order-to-cash cycle efficiently and effectively ensured? Are there side effects such as improved efficiency and effectiveness of the procure-to-pay performance (e.g. lower cost, improved goods quality)? Is non-ethical behaviour prevented?

Naturally these questions may be complemented by specific questions relevant to the respective research project. Orientation along the frame of reference helps to create a high-level process model to structure the research.

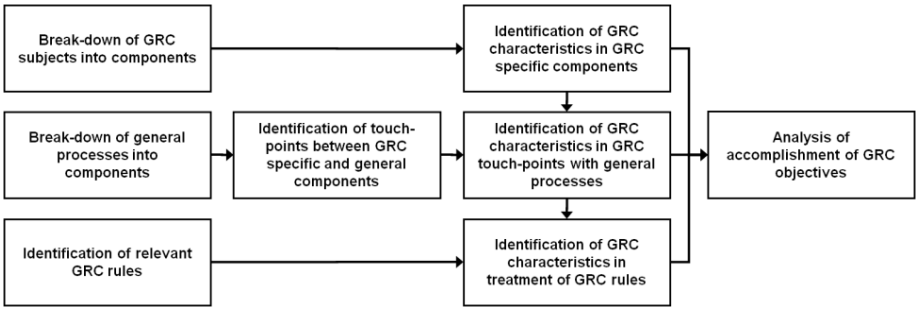


Fig. 7. Exemplary process model for integrated GRC research

6 Discussion

Admittedly, putting the complexity of GRC into a single phrase is provocative. One sentence cannot catch all inherent notions. However, in contrast to other definitions, the definition presented here considers the commonalities and the focus of the whole of prior publications and research on GRC. So far it is the only definition that has been derived in an empirical, scientific manner. Moreover it has experienced GRC professionals' acceptance as shown by the survey. Thus compared to prior definitions it should be more representative for the whole spectrum of GRC.

Of course the approach to derive a definition by means of a literature review has certain disadvantages. Some sources were of rather poor quality. The publishing groups have a business interest, which questions the objectivity of their articles. We assume that the large number of publications reviewed largely makes up for this disadvantage. Another approach could have been to conduct structured interviews with GRC experts.

The effort however would have been incomparably higher if an objective result not dominated by a small number of opinions was to be achieved. In addition we doubt that the quality would have been significantly higher; the statements given in interviews would not have had a scientific foundation either.

The frame of reference naturally only displays a high-level abstraction of GRC. It does not visualize the massive complexity of GRC, but it is not meant to do that. As long as it helps researchers to gain a quick first understanding of integrated GRC in order to structure their research, it fulfills its purpose.

The contribution of this research paper consists of three aspects. Firstly, for the first time GRC publications have been reviewed; the lack of research on GRC is now obvious. Secondly, for the first time a GRC definition has been derived rigorously in a scientific manner and it has been validated by GRC professionals, thus proving its relevance. Thirdly, a frame of reference has been constructed that may be used for GRC reference modeling or other research of integrated GRC. The knowledge base of the information systems research framework [30] has been extended (while the use of our results is not restricted to IS research, of course). If the complexity of GRC has so far been a barrier holding off research, we hope that we have lowered this barrier.

7 Conclusion and Future Research

The analysis at hand clearly shows that integrated GRC is a widespread topic that has not yet been adequately researched. We can see what happens to a topic that lacks a common forum for communication of professionals as research could offer. The information provided publicly remains at a high level; understandably neither software companies nor consultancies want to give away their knowledge for free. Different products and marketing efforts have created a domain consisting of lots of shared buzzwords but missing clarity. The myriad of perceptions of GRC harms the development of a rising topic. At least there is a consensus on a few key points regarding GRC which we included in our results. Our definition and the frame of reference are a first step towards a more active role of research in integrated GRC. In the near future we will try to enlarge the basis for GRC research by breaking down and describing the frame's components in order to create a research framework and reference model for integrated GRC in information systems management. We encourage other researchers to build on our results and to use the definition and the frame of reference in their own research of GRC.

References

1. PricewaterhouseCoopers: 8th annual global CEO survey, http://www.globes.co.il/Serve/Researches/documents/8thAnnualGlobalCEOSurvey.pdf
2. Leibs, S.: One for three. CFO Magazine (September 2007), http://www.cfo.com/article.cfm/9689509
3. Dittmar, L.: Demystifying GRC. Business Trends Quarterly 2(4), 16–18 (2007)
4. Kahn Consulting: GRC, E-Discovery, and RIM: state of the industry, http://www.kahnconsultinginc.com/library/KCI-GRC-RIM-EDD- survey.pdf
5. Rasmussen, M.: 2008 GRC drivers, trends & market directions, http://www12.sap.com/community/showdetail.epx?ItemID=11997

6. Ahlemann, F., Gastl, H.: Process Model for an Empirically Grounded Reference Model Construction. In: Fettke, P., Loos, P. (eds.) Reference Modelling for Business Systems Analysis, pp. 77–97. Idea Group, Hershey (2007)
7. Broady, D.V., Roland, H.A.: SAP GRC for dummies. Wiley, Indianapolis (2008)
8. Fettke, P.: State-of-the-Art des State-of-the-Art. Eine Untersuchung der Forschungsmethode 'Review' innerhalb der Wirtschaftsinformatik. Wirtschaftsinformatik 48/4, 257–266 (2006)
9. Schlagheck, B.: Object-oriented reference models for process and project controlling. In: Foundation-construction-fields of application. Deutscher Univ.-Verlag, Wiesbaden (2000)
10. Mitchell, S.L.: GRC360: A framework to help organisations drive principled performance. International Journal of Disclosure and Governance 4(4), 279–296 (2007)
11. Tapscott, D.: Trust and competitive advantage: an integrated approach to governance, risk & compliance (2006), http://www.findwhitepapers.com/whitepaper1714/
12. Kelly, J.: Risk management surpasses compliance as top GRC priority, http://go.techtarget.com/r/3484977/6129174
13. Banham, R.: Is ERM GRC? Or vice versa? Treasury & Risk 2(6), 48–50 (2007)
14. Mitchell, S.L.: GRC – more than three letters, http://grc360.blog.oceg.org/2007/08/grc-more-than-three-letters.html
15. Hoffmann, M.: Governance, Risk und Compliance (GRC) – ein integrierter Ansatz. IM 24(1), 74–81 (2007)
16. Switzer, C.S.: Integration innovation. Business Trends Quarterly 2(4), 26–32 (2007)
17. Curran, B.: Defragmenting GRC. Pharmaceutical Technology 4(16), 20–23 (2007)
18. KPMG: Governance, risk, and compliance. Driving value through controls monitoring, http://www.kpmg.ca/en/services/advisory/documents/GovernanceRiskCompliance.pdf
19. Economist Intelligence Unit: Managing risk through financial processes. Embedding governance, risk and compliance, http://graphics.eiu.com/marketing/pdf/SAP%20GRC.pdf
20. Wechsler, P.: The GRC harmony. Treasury & Risk 2(6), 13 (2008)
21. Corporate Integrity: What is GRC?, http://www.corp-integrity.com/about/grc.html
22. Hovis, J.J.: CIO at the center, http://www.oracle.com/dm/08q3field/ogec_wp_cio.pdf
23. OCEG: GRC capability model. Red Book 2.0 (2009), http://www.oceg.com
24. Vemuri, A.: Strategic themes in risk and compliance. FINsights 2, 2–5 (2008)
25. Frigo, M.L., Anderson, R.J.: A strategic framework for governance, risk, and compliance. Strategic Finance 90(8), 20–61 (2009)
26. Approva Corporation: 2007 Approva GRC survey (2007), http://www.approva.net/survey
27. Teubner, A., Feller, T.: Informationstechnologie, Governance und Compliance. Wirtschaftsinformatik 50(5), 400–407 (2008)
28. IT Policy Compliance Group: 2008 Annual Report. IT Governance, Risk, and Compliance (2008), http://www.itpolicycompliance.com/pdfs/ITPCGAnnualReport2008.pdf
29. Rath, M., Sponholz, R.: IT-Compliance: Erfolgreiches Management regulatorischer Anforderungen. Schmidt, Berlin (2009)
30. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Quarterly 28(1), 75–105 (2004)

Business and IT Continuity Benchmarking

Wolfgang Neudorfer[1], Louis Marinos[2], and Ingrid Schaumüller-Bichl[1]

[1] Upper Austria University of Applied Sciences, Campus Hagenberg,
Department Secure Information Systems
{wolfgang.neudorfer,ingrid.schaumueller-bichl}@fh-hagenberg.at
[2] European Network and Information Security Agency
louis.marinos@enisa.europa.eu

Abstract. The paper introduces a benchmarking approach for business and IT continuity. Multiple use cases of the benchmark are presented enabling organizations of varying sizes and sectors to determine their continuity requirements in a comprehensive manner. Furthermore, the benchmark can be used to compare different business continuity methodologies, but also to identify an appropriate methodology that meets an organization's requirements. This will help organizations to establish and sustain an effective and efficient business continuity process without the need of extended know-how in that area.

Keywords: business continuity management (BCM), risk management (RM), benchmarking, continuity profile, continuity requirements.

1 Introduction

Today's organizations are exposed to numerous sources of threats and vulnerabilities which may have critical impact on their business continuity. In addition, the time after which unavailability may cause irrevocable damage or even threaten the survival of the organization gets shorter and shorter. Therefore, the introduction of a business continuity process that improves the organization's resilience may be of vital importance for an organization.

The establishment and ongoing management of such a process can be both time-consuming and expensive. With the presented benchmarking approach we aim at supporting organizations in understanding both what business continuity is about and what their continuity requirements are. The goal of the paper is enabling organizations of varying sizes and sectors to answer the following questions:

1. What are the continuity requirements for organizations of typical size and complexity?
2. How can an organization determine its continuity profile and identify its basic requirements without prior knowledge in business continuity?
3. How can an organization determine its detailed continuity requirements?
4. How can different business continuity methodologies be compared?

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 118–129, 2010.

5. How can an organization identify a methodology that meets its own requirements?

The paper is structured according to these questions, whereas Sect. 4 responds to every question with a particular use case of the benchmark.

2 Background

The business continuity process [3] from the European Network and Information Security Agency (ENISA) forms the basis for the current paper. The process depicted in Fig. 1 is being developed as a consolidated overview of relevant methodologies, standards and literature [3, p. 14]. Main objective for the development of this continuity process was to serve as an all-encompassing basis of continuity issues covered by existing approaches. The process has been used for the development of an inventory of continuity methods and tools [4]. Given the properties and due to the independency of ENISA, the process is considered as a solid structure for the presented benchmark.

Figure 1 shows the ENISA business continuity process consisting of six stages[1] and 24 sub-processes. The content of the stages is outlined in the following paragraphs, more detailed information can be found in [3, pp 27-64].[2]

In the first stage the peculiarities of the organization and its environment are identified and responsibilities assigned. The determination of business critical processes, their interdependencies and their supporting assets are considered to be of utmost importance. The BCM policy finally sets out the organization's aims, principles and the approach to BCM.

In the second stage the business impact analysis (BIA) is conducted. The main outcomes from this analysis are the identification of recovery time objectives (RTO), the maximum tolerable period of disruption (MTPD), the recovery point objectives (RPO) and the minimum level of resources needed by critical processes. A recovery profile shows over time what assets are being recovered as well as the specific resources (staff, premises, equipment) that are required at any time to support the recovery efforts.

In the third stage the BCM approach is designed. Given the results from the previous stages, feasible recovery options are developed and a cost-benefit analysis is conducted. Top management finally signs off a business continuity strategy. As a result of this the business continuity plan (BCP) is designed.

In the fourth stage the BCP is delivered. Following ENISA, the BCP is not a single document but rather a suite of documents used by different teams and in different points of time. These documents shall be seen as suggestions, not all of them have to be necessarily implemented.

[1] A stage is considered to be a group of sub-processes.

[2] It should be noted that the scope of the process does not address the continuity of critical business processes itself but rather the continuity of ICT that is needed to provide an automated means for the critical processes. Nevertheless a holistic perspective on the organization is given throughout the process.

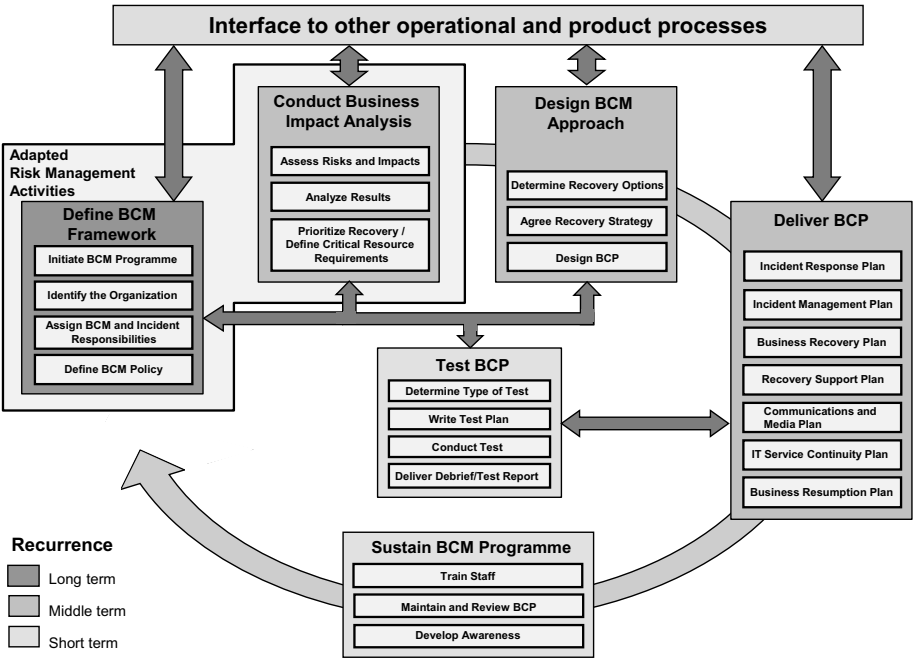


Fig. 1. ENISA business continuity process [3, p.22]

In the fifth stage the BCP is tested. Testing can be achieved through many different test types reaching from simple walkthroughs to complete scenario exercises. Hence, a test plan including scope, timetables and success criteria is worked out.

In the sixth stage the BCM programme is sustained. Modification of the BCP may be necessary due to various reasons reaching from identified shortcomings to changes in the business itself. This stage also ensures that involved personnel are well-trained and capable of handling an incident of unavailability. In addition, it is necessary that all staff is informed about the key principles and that business continuity is embedded in the organizations culture.

3 Benchmarking Approach

In 2007-2008 ENISA has defined a risk assessment and risk management benchmark [5]. The approach followed for risk management has been adopted for the development of the present continuity benchmarking.

The benchmark is built upon the ENISA business continuity process and divides BCM in the same six stages and 24 sub-processes as described in Sect. 2. In addition, the sub-processes are further characterized by information a sub-process needs (i.e. input) and information a sub-process delivers (i.e. output). While inputs and outputs have been defined for all 24 sub-processes, we do not

describe how an input is turned to an output (i.e. processing step). We make the assumption that this can happen in various ways. We have left this information out on purpose in order not to restrict ourselves by means of how a user can best perform a process but what is processed by the process. We believe that this decision is a step towards neutrality of our approach. For the developed benchmark, 87 inputs and 65 outputs have been defined in total.

Due to size limitations we present here for the sake of illustration the inputs and outputs of "P.2 Identify the Organization". The used numbering consists of three parts separated with dots. Starting with an I for input and an O for output, the second part identifies the sub-process the input or output is defined for. The third part is a consecutive number of the inputs or outputs of the particular sub-process.

- Inputs of P.2:
  - **I.2.1** *Business areas of the organization:* A definition of all areas the organization is operating in.
  - **I.2.2** *Strategy of the organization (goals, objectives, strengths, weaknesses, opportunities and threats, culture, structures):* The organizations business strategy is a crucial knowledge for BC planning. Planned mergers or the introduction of new products have a strong influence on BCM.
  - **I.2.3** *Description of internal stakeholders:* An understanding of all internal stakeholders (e.g. employees, business owner, the board) interests and obligations is necessary.
  - **I.2.4** *Description of external stakeholders:* An understanding of all external stakeholders (e.g. shareholder, customer, government, supplier, environment) interests and obligations is necessary.
  - **I.2.5** *Description of assets (inventory list, people, systems, processes, capital, etc.):* The assets of the organization are protected within the BCM against unavailability risks. Their location during normal business operation is an important factor in BCM design.
  - **I.2.6** *Financial and political information:* A definition of the financial and political environment of the organization.
  - **I.2.7** *Relevant legal and regulatory information:* A list of regulations and legislation the organization is faced with.
  - **I.2.8** *Information about geographical, social and cultural conditions:* The geographical position(s) of the organization has/have a big influence on BC planning. Social and cultural conditions can have influences too.
- Outputs of P.2:
  - **O.2.1** *Description of critical business processes and interdependencies between them:* These business processes are critical for the survival of the organization. The identification of critical processes and their interdependencies is of utmost importance. Special attention should be drawn to changes in business (see also I.2.2).
  - **O.2.2** *Assets supporting critical business processes:* Assets that are supporting critical processes are crucial to the organization.

- **O.2.3** *Description of relationships between O.2.1 and O.2.2:* The understanding of the relationships between the assets and the critical business processes is essential to BCM. Attention should be drawn at the interdependencies of the critical processes.
- **O.2.4** *All records of the external environment of the organization:* A summary of the environment of the organization (e.g. external stakeholders, competitors, financial and political information, social and cultural conditions).
- **O.2.5** *List of relevant obligatory laws and regulations (with respect to obligations):* Legal and regulatory information are important as the organization may suffer from liability issues due to an unavailability.
- **O.2.6** *List of key responsibilities and positions inside the organization:* Personnel that will be involved in the handling of unavailability.

A desired side-effect of the defined inputs and outputs of the business continuity process is the development of an integrated approach with a risk management process. As the BC process makes use of risk assessment and risk management techniques, inputs and outputs of the RM process (as defined in [5]) have been reused to establish an integrated, cost-effective and concise management process for both risk and business continuity. This is considered to be of particular importance because none of either process can be managed in isolation from the other.

For the valuation/rating of inputs and outputs we introduce two concurrently used scales, namely scores and weights. Scores are used to quantify the importance of an input/output, while weights are used to qualify them. The details of scores and weights are as follows:

Scores: Depending on the particular use case, scores indicate the amount of information an input or output should carry. A low score indicates that an input, for example, is only rudimentary described, while a high score indicates that the input is described in great detail. The scoring of the sub-process can be specified by calculating the average of all inputs and outputs of the particular sub-process.

Weighting: Not all inputs and outputs are considered mandatory during implementation of a continuity plan. Some of them are considered to be negligible or not needed yet recommended. As a consequence, a weighting is performed on inputs and outputs.

For better visualization and comparison a radar chart is used for the results of the benchmark. The radar chart can be drawn at the level of sub-processes or at the level of inputs and outputs, depending on the desired detail of visualization. Examples of these charts are shown in Fig. 2 and Fig. 4 respectively later in this paper.[3]

[3] The axes of the chart are scaled to start at -1 to avoid the problem of blank cells should a sub-process have zero score [5, p.6].

4 Use Cases of the Benchmark

In the following section we present five practical use cases of the defined benchmark. These use cases focus on different target groups reaching from managers without deep technical knowledge to experts in business continuity. Use cases 1 and 2 are attractive to non-experts and do not require previous knowledge while use case 3 can be used by a specialist to generate precise results. Use cases 4 and 5 may be used by both, experts and non-experts.

4.1 Use Case 1: Determining Typical BC Organizational Requirements

If an organization decides to establish a BCM process it may wish to understand what typical continuity requirements are. Therefore, this paper defines five characteristic types of organizations [5, p.8]:

1. Small business with limited usage of information and communication technologies (ICT).
2. Small to medium-sized business with more extensive usage of ICT.
3. Medium-sized private business with simple governance requirements.
4. Medium to large-sized business with more complex governance requirements.
5. Large-sized business with rigorous governance requirements.

Users of the benchmark can use these characteristic requirements in order to find out how much their continuity profile (see Sect. 4.2) deviates from typical profiles of organizations of the same or similar type.

The benchmark can be used to evaluate the requirements of the types by using the scoring of Sect. 3. The adapted scheme for this use case is shown in Table 1. Figure 2 summarizes the results of the use case and visualizes the performed scoring of all five types in a radar chart.

Table 1. Scoring scheme for use case 1 at sub-process level [6, p.21]

Description	Score
The sub-process is not required for this type.	0
A simple and informal sub-process is required for this type.	1
A formal and documented sub-process is required for this type.	2
A detailed sub-process with full documentation and auditing is required for this type.	3

4.2 Use Case 2: Generation of Continuity Profiles

Obviously, the generic alignment profiles of the first use case cannot exactly describe the requirements of an organization. For a more accurate determination the benchmark can be used to identify a more specific continuity profile. Based on the risk profile in [6], the continuity profile is generated through answering a questionnaire focusing on

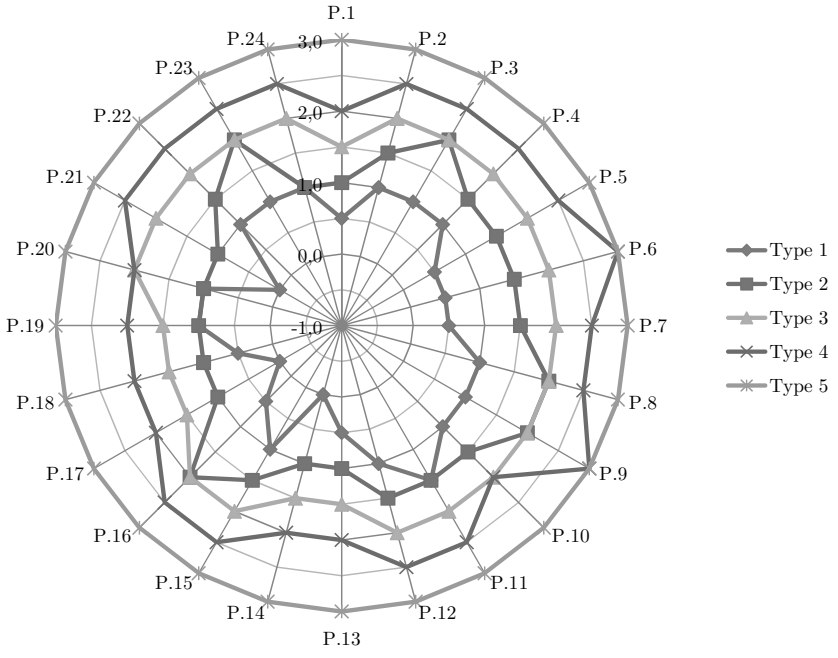


Fig. 2. Typical profiles for organizations of use case 1

- the exposure to threats and vulnerabilities (that can lead to an unavailability) and
- the impact of unavailabilities.

of the organization. By analyzing the results from the questionnaire each continuity profile provides information about an organization's BCM requirements and gives recommendations as well as additional information for the establishment of a BCM process.[4]

Questionnaire. In order to qualify an organization's requirements for BCM the presented approach delivers 14 questions about the exposure to threats and potential sources of vulnerabilities (referred to as "exposure") and the potential impact of these (referred to as "impact"). The questions can be classified in the following sections:

- Qualification of exposure:
  - As a result of the organization's nature and its business model
  - As a result of threats
  - As a result of vulnerabilities

[4] Nevertheless, it should be noted that the identification of a continuity profile shall not be confused with a business impact analysis including detailed risk assessment and risk management methodologies.

– Qualification of impact:
  - As a result of its business strategy and its environment.
  - As a result of the organization's legal and regulatory requirements.
  - As a result of the organization's dependency on ICT systems.

All questions offer the responding organization the option of choosing one of four possible answers. Each question is scored depending on the answer given. The scoring of regular questions is 1-2-3-4 points, while for questions which are of high importance for the continuity profile the scoring is raised to 1-2-4-8 points. To illustrate the approach, one question regarding the exposure (Table 2) and one question regarding the impact (Table 3) are shown as an example.

Table 2. Example question regarding the exposure

(3) - To what extent do you feel that your organization is exposed to natural disasters (e.g. storms, floods, bush fire, earthquakes, hurricanes, pandemic diseases) due to its geographic location(s)?	**Score**
Very little	1
Some	2
Potentially significant	4
Potentially critical	8

Table 3. Example question regarding the impact

(9) - What is the overall tolerable period of disruption for your organization (rough estimate)?	**Score**
1 week or more	1
2 to 4 days	2
less than 1 day	4
less than 4 hours	8

Analysis of the Answers. By summing up all scores regarding the exposure and all scores regarding the impact, an *exposure and impact vector* can be created. Through plotting the vector on a xy-chart preliminary indication of the exposure and impact can be given. Figure 3 illustrates the results of an example organization with a scoring of 20 points for exposure and 17 points for impact.

Figure 3 also reveals that each axis is divided into four parts, resulting in 14 sectors.[5] For each sector all 87 inputs and 65 outputs are separately evaluated using a similar scoring scheme as in Table 1, but at the level of inputs and outputs. As a result of the scoring and weighting a specific alignment profile for each of the 14 sectors can be drawn by calculating the averages at sub-process level. Table 4 shows the scoring of sector 4 which corresponds to the profile of the example organization.

[5] The approach is adapted from [6, p.10]. Further, [6] concludes that organizations expecting a low impact are unlikely to have significant or high exposure. Therefore, these two sectors have been ignored and are not considered in the plot.

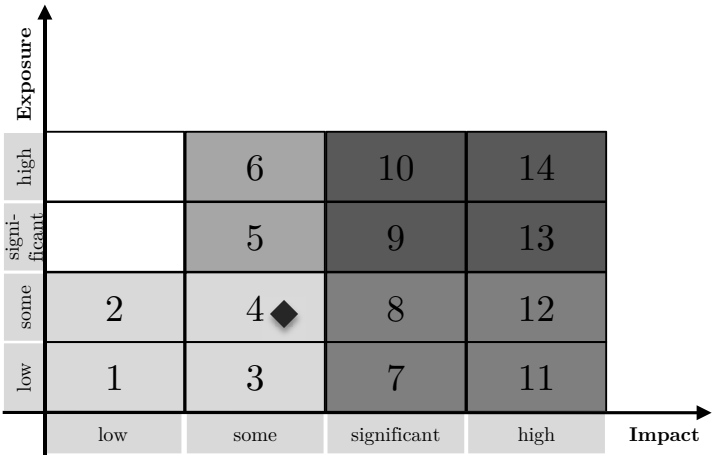


Fig. 3. Exposure and impact vector for example organization: the x-axis represents the impact; the y-axis represents the exposure

Table 4. Weighted scoring for sector 4

P.1	P.2	P.3	P.4	P.5	P.6	P.7	P.8	P.9	P.10	P.11	P.12
1	1	1	1	1	1	1	1	1	1	1	1
P.13	**P.14**	**P.15**	**P.16**	**P.17**	**P.18**	**P.19**	**P.20**	**P.21**	**P.22**	**P.23**	**P.24**
1	0	1	1	0	1	1	1	0	1	1	1

Determination of Requirements. On basis of the previously presented alignment profile general information, levels of concern, requirements, and recommendations of the 14 segments can be expressed. The following listing shows an excerpt of the description of sector 4.

- General information:
  - Organizations of this type may encounter some problems, but these are not likely to be critical to their business.
  - If your organization is of this type, you should understand the continuity risk to your business and have regular means of reviewing.
- Level of concern:
  - The business areas that are critical to the organization shall be identified and responsibilities have to be assigned.
  - Recovery options of critical business areas have to be determined.
- Requirements: (excerpt)
  - Identify critical business areas and understand their dependencies to ICT.
  - Define RTO and RPO for critical business areas.
  - Develop checklists for the recovery of ICT services.
- Recommendations: (excerpt)

- Develop preventative controls for the resilience of critical business areas.
- All (internal or external) changes that impact the organization are reviewed in relation to business continuity.

4.3 Use Case 3: Detailed Determination of Requirements

The second use case achieves the identification of a convincing continuity profile in very little time and without deep knowledge of the subject matter. However, experts in business continuity may wish to comprehend the characteristics of an organization in greater detail than a questionnaire is able to deliver. For this purpose the benchmark offers the opportunity to self-evaluate all 87 inputs and 65 outputs with the proposed scoring from 0 to 3 points (including intermediate values) in respect to the organization's needs. Also the suggested weighting of the inputs and outputs may be adjusted to the own requirements. Following this approach a very accurate alignment profile can be created.

4.4 Use Case 4: Benchmarking of BC Methodologies

A number of BC methodologies is currently available. Even for experts it is a difficult task to compare the different items and identify their strengths and weaknesses. Applying the scoring scheme shown in Table 5 the benchmark can be used to characterise the methodologies according to the degree of convergence they have to their equivalents in the benchmark.

Table 5. Scoring scheme for use case 4 [5, p.5]

Description	Score
Input/output is not mentioned at all.	0
Input/output described with only reference to an external process.	1
Input/output described in some detail with simple instructions.	2
Input/output described in great detail with exhaustive instructions.	3

For this paper the benchmark has been applied by three different experts to four common methodologies. These are:

- British Standard 25999-1 [2],
- BCI Good Practice Guidelines 2008 [8],
- BSI Standard 100-4 [1] and
- ENISA IT Business Continuity Management: An Approach for SMEs [7].

Furthermore, this paper consolidates the different ratings conducted by the experts and tries to eliminate subjectivity. Using the objective ratings it is again possible to visualize the results of the evaluation in a radar chart and directly compare the strengths and weaknesses of two or more items in a single chart. It should be noted that the comparison can be conducted at the level of subprocesses, or more detailed, at the level of inputs and outputs.

4.5 Use Case 5: Combination of Use Cases

By combining alignment profiles of a particular organization (as in use cases 1-3) and of those of business continuity methodologies (as in use case 4), it is possible to determine an appropriate methodology for the organization. In addition to that, the organization may also wish to select particular sub-processes from different methodologies to achieve best results. For example, although an organization may use the SME approach from ENISA it may decide that its circumstances require more detailed testing of the continuity plans as described in BSI Standard 100-4 [1]. For the sake of illustration, Fig. 4 shows the profile of the example organization and the methodology from ENISA for SMEs [7].

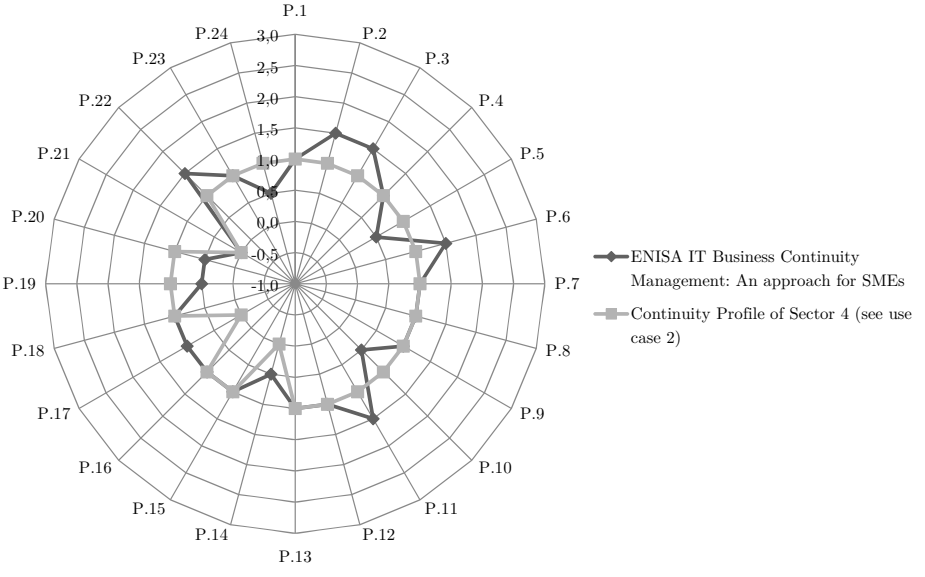


Fig. 4. Continuity profile of the example organization (see use case 2) and a BC methodology (ENISA IT Business Continuity Management: An Approach for SMEs)

5 Conclusion

Availability is a basic principle of communication and multimedia security. A way to proactively improve an organization's resilience against the disruption of its ability to achieve its key objectives is the introduction of a business continuity process [2].

This paper presented a business continuity benchmarking approach. Further, practical use cases showed how the benchmark can be used to determine the continuity requirements of organizations of various sizes and sectors. The use cases also showed how an appropriate methodology for the establishment and ongoing management of a business continuity process can be found.

Future work in this field may include the consolidation of the presented results and the assessment tool *Self Assessed Risk Profiler (SARP)* as introduced in [9]. SARP is able to generate the risk profile for an organization and automatically deliver the appropriate description of its risk assessment and management objectives [6]. The combination of the risk and the continuity profile as well as the comprehensive integration of the risk and business continuity management process (as discussed in Sect. 3) promises to deliver an effective and efficient approach to integrate these related disciplines and make them usable for non-experts and in particular for small and medium enterprises.

References

1. Bundesamt für Sicherheit in der Informationstechnik (BSI): BSI Standard 100-4: Notfallmanagement - Version 1.0 (2008)
2. British Standards Institution: British Standard 25999-1:2006 Business Continuity Management - Part 1: Code of practice (2006)
3. European Network and Information Security Agency (ENISA): Business and IT Continuity: Overview and Implementation Principles (2008)
4. European Network and Information Security Agency (ENISA): Inventory of Business and IT Continuity Tools, http://www.enisa.europa.eu/act/rm/cr/bcm-resilience/bc-tools
5. European Network and Information Security Agency (ENISA): Methodology for evaluating usage and comparison of risk assessment and risk management items (2007)
6. European Network and Information Security Agency (ENISA): Determining Your Organization's Information Risk Assessment and Management Requirements and Selecting Appropriate Methodologies (2008)
7. European Network and Information Security Agency (ENISA): IT Business Continuity Management: An Approach for SMEs (2010)
8. The Business Continuity Institute: Business Continuity Management Good Practice Guidelines 2008 (2007)
9. Pöttinger, J.: Selbsthilfe für IT-Risikomanagement - Ein Benchmarking-Ansatz, 11. Deutscher IT-Sicherheitskongress (2009)

Peer-to-Peer VoIP Communications Using Anonymisation Overlay Networks*

Ge Zhang and Simone Fischer-Hübner

Computer Science Department,
Karlstad University, Universitetsgatan 2, 65188, Karlstad, Sweden
{ge.zhang,simone.fischer-huebner}@kau.se

Abstract. Nowadays, Voice over Internet Protocol (VoIP) which enables voice conversation remotely over packet switched networks gains much attentions for its low costs and flexible services. However, VoIP calling anonymity, particularly to withhold "who called whom", is difficult to achieve since VoIP infrastructures are usually deployed in an open networking environment (e.g., the Internet). Our work studies an anonymisation overlay network (AON) based solution to prevent surveillance from external attackers, who are able to wiretap the communication channels as well as to manipulate voice packets in the channels. However, it has been demonstrated that the VoIP combined with traditional AONs are vulnerable to two attacks, namely watermark attack and complementary matching attack. Taking these two attacks into account, we investigate the "defensive dropping" method in VoIP: A VoIP user-agent sends packets to an AON in a constant rate, but packets during periods of silence are marked. Then, the AON drops some silence packets and forwards the remaining ones to their destinations. The result of our experiments shows that the dropping rate must be carefully selected to counteract both of the two attacks. Finally, we discuss further threats in terms of this solution.

1 Introduction

VoIP is becoming increasingly popular due to its benefit on low-cost, flexibility and scalability compared to traditional telephony systems. However, VoIP infrastructures are usually deployed in large-scale packet-switched networks, such as the Internet, which is open in contrast to traditional Public Switched Telephone Network (PSTN). In this way, it is relatively easy for attackers to break into the VoIP networking environment and access the service infrastructures. On the other hand, generated voice packets sometimes have to be routed over some untrustful networking environment to reach their destinations. Therefore, VoIP services face much security and privacy challenges which have not occurred on traditional telephony systems.

* The authors would like to thank Stefan Köpsell and Stefan Berthold for their valuable comments and suggestions.

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 130–141, 2010.

Like there are privacy requirements by users on other online applications, VoIP users may also request privacy protection. For instance, users may want to keep their conversation content secret, so that eavesdroppers on the communication channels are unable to intercept the content. This requirement can be realized by encrypting voice packets. Moreover, to prevent surveillance, users may sometimes want to make conversations anonymously so that potential attackers over the communication channels cannot find out "who called whom". To achieve this requirement, one potential solution is to employ anonymisation overlay networks (AON), which contain scalable relay proxies. An AON accepts k flows in and produces k flows out. The flows out are differently encoded than the flows in. Thus, it is difficult for attackers to relate between the flows and users. In practice, AONs are mostly used for email and web surfing, rather than VoIP. It is due to several specific features of VoIP, which make VoIP conversation on AONs vulnerable to several attacking methods: Attackers can embed a watermark into flows before they enter an AON by slightly delaying some randomly selected packets and then decode the watermark from the flows leaving the AON [18]. In this way, the attackers can correlate the flows separated by the AON to deduce "who called whom". Furthermore, VoIP users usually follow human conversation pattern, which means that one user speaks while another listens. Attackers also can exploit this pattern to pair flows for tracing [16]. This paper proposes a scheme based on the concept of defensive dropping [6] to prevent these two attacks: A VoIP user-agent sends voice packets to an AON for both speech and silence periods. However the packets during silence periods are marked and indicated by encrypted flag bits. When the AON receives the packets, it drops some silence packets and forwards the remaining ones to their destination. The dropping method makes the relationship between flows more difficult to be observed. The results of our simulation shows that this solution makes the two attacks more difficult, however the dropping rate must be carefully selected.

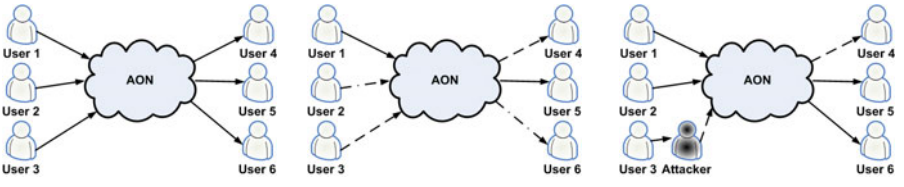
The rest of this paper is organized as follows. Section 2 introduces the background of VoIP and anonymisation overlay networks. Section 3 shows related work in VoIP anonymity. Section 4 proposes the solution and its evaluation. In Section 5, we discuss further security and anonymity issues regarding this solution. Finally, we summarize this paper and present future work in Section 6.

2 Background

2.1 VoIP, Codec and Silence Suppression

VoIP applications typically use two kinds of protocols: a signaling protocol for call setup and termination (e.g., the Session Initiation Protocol (SIP) [11]) and a media delivery protocol for voice packets transmission (e.g., the Realtime Transport Protocol (RTP) [12]). To realize a VoIP conversation, both caller and callee should send encoded voice packets to each other, so called *bidirectional voice flows*. Transmitted voice packets are encoded and decoded using a speech codec algorithm (e.g., G.711 [1] and Speex [4]) negotiated in the signaling level. The

codec takes the voice from users as input, which is typically sampled at either 8k samples or 16k samples per second (Hz). As a performance requirement, the packet inter-arrival time of voice flow is usually fixedly selected between 10 and 50 ms, with 20 ms being the common case. Thus, given a 8kHZ voice source, we have 160 samples per packet with 20 ms packets interval. Moreover, the size of each voice packet depends on the encoding bit rate of adapted codec. Two types of encoding bit rates can be distinguished: **Fixed Bit Rate (FBR)** and **Variable Bit Rate (VBR)**. With FBR (e.g., G.711), end points produce voice packets always with the same size. On the other hand, VBR (e.g., Speex) means that the encoding bit rate varies according to the type of voice. In this way, user-agents produce voice packets with different size. Moreover, there are two kinds of voice packets: the packets generated during speech of a user, namely *speech packets* and the packets generated during silence of a user, namely *silence packets*. Some VoIP user-agents allow discontinuous voice packets transmission (**silence suppression**) [21], which is a capability of user-agents to stop sending silence packets during silent periods of its owner. In this circumstance, networking resources (e.g., bandwidth) can be significantly saved.



(a) No characteristic can be distinguished for flows

(b) Each traffic flow has its own characteristic

(c) An attacker injects characteristics into a flow

Fig. 1. Traffic analysis attacks on a VAS

2.2 Anonymisation Overlay Networks (AON)

AONs enable anonymous communications in untrustful networks. D. Chaum [8] firstly proposed a mix-based concept: Mix-networks contain scalable relay proxies that accept k incoming messages and forward them. The proxies change the order and appearance of messages so that it is difficult for an eavesdropper to distinguish which outgoing message corresponds to which incoming message. As a result, given a message, the attacker is unable to pair its communication partners. Generally, two kinds of AONs exist in reality: high-latency AONs for time insensitive applications (e.g., email) and low-latency AONs for time sensitive applications (e.g., web surfing). As VoIP applications have restrict requirements on real time, it is fairly clear that high-latency AONs are not suitable to protect VoIP anonymity. Thus, the AONs in the rest of this paper only refer low-latency AONs.

Traffic analysis attack aims to correlate the flows on both side of AONs by exploiting a certain characteristic (e.g., packet size or packet inter-arrival time) of the flows. A flow entering an AON and a flow leaving it can be related if they share similar characteristics. For example, an attacker cannot relate the

flows in Figure 1(a) as all flows look similar. It is hard to say with whom user1 communicates (could be user4, user5, or user6). However, each flow has its own characteristics in Figure 1(b). In this case, attackers can easily relate the flows by their characteristics (user1 $\leftrightarrow$ user5, user2 $\leftrightarrow$ user4, user3 $\leftrightarrow$ user6). Instead of passively observing, Attackers can also modify a flow to insert more characteristics before it enters the AON (active attack), as illustrated in Figure 1(c).

Many proposed defending methods (e.g., reorder, packets-padding, dummy traffic and packets-delaying, etc) aim to eliminate (or generalize) the characteristics of flows to mitigate traffic analysis attacks. However, these methods usually introduce much overhead on networks and impact the service performance.

3 Related Work

Many researchers have recently paid their attention on VoIP privacy. RFC3323 [10] and C. Shen, et al. [13] discussed the requirement of VoIP anonymity and proposed Trusted Third Party (TTP) based solutions for protecting sensitive information on signaling layer. M. Srivatsa, et al. [14] [15] proposed and compared several privacy-aware routers setup protocols for VoIP applications in P2P networking environments. C. V. Wright, et al. [20] [19] demonstrated that attackers can observe the used language or even the content of a conversation from the varying size of voice packets even if the packets are end-to-end encrypted. However, the scope of our paper is different to theirs: *our paper focuses on VoIP anonymity on media layer to hide who called whom from external attackers.*

C. A. Melchor et al [9] discussed three techniques providing strong traffic analysis resistant for VoIP media flow communications. The three techniques are global dummy traffic, broadcasting and private information retrieval respectively. They further evaluated the performance of these techniques based on a theoretical model. Nevertheless, these techniques are too costly to be deployed in reality.

X. Wang, et al. [18] investigated a practical solution in which users can make a skype [3] call over a commercial AON. Their test proved that VoIP over AONs can be practical. However, they also demonstrate that such a solution is still vulnerable to active timing attack: An attacker can embed unique watermark into the encrypted VoIP flow by slightly delaying of random selected packets. In this way, an attacker can find out who called whom by encoding and decoding watermarks on both side of an AON. O. Verscheure et al. [16] [17] proposed a method to reveal who called whom by exploiting the human conversation pattern: When one speaks, the other listens. This "alternate in speaking and silence" represents a basic rule of VoIP communication. Therefore, an attacker can easily pair the caller and callee by matching the bidirectional flows on signaling layer as long as silence suppression has been applied in the VoIP system. Nevertheless, neither of them suggested corresponding countermeasures. In this paper, we address a potential defending solution to mitigate these two attacks.

4 VoIP Anonymization Using AONs

4.1 System Model

User side: We limit user-agents so that they only support FBR codec instead of VBR codec. Thus, each generated voice packets should be of equal size. Otherwise, with VBR traffic analysis is easily possible. We further assume that user-agents provide an option for silence suppression. In this way, user-agents constantly produce voice packets (both speech and silence packets) if silence suppression is disabled. Otherwise, user-agents will stop generating speech packets. Finally, we assume that each voice packet is end-to-end encrypted (e.g., using SRTP).

AON: We assume that the anonymity service is provided by a fixed sequence of n proxies, which is similar to some existing implementations (e.g., An.on [7]). The AON supports bidirectional communications from the first one to the last one and vice versa. We treat the entire AON as a black box and will not focus on the system-specific details of the path establishment protocol between them. We assume that the signaling service can coordinate at least k pairs of users to establish (and terminate) voice flows over the AON simultaneously. We split the users into two groups: namely $\mathbb{L}$ and $\mathbb{R}$, where callers in $\mathbb{L}$ call callees in $\mathbb{R}$ over the AON, as illustrated in Figure 2. The k callers in $\mathbb{L}$ are indicated as $L_1, L_2, ..., L_k$ and the callees in $\mathbb{R}$ are referred as $R_1, R_2, ..., R_k$. However, readers should notice that the relationships between the users in the two groups are not deterministic: It does not mean that L_i certainly called R_i $(1 \leq i \leq k)$.

Voice flows: As introduced above, a VoIP conversation consists of 2 flows with opposite directions from two users. Considering the AON deployed in the middle, the 2 flows are splited into 4 flows. We use $\overrightarrow{L_iM}$ denotes the flow from L_i to the AON and $\overrightarrow{ML_i}$ indicates the flow from the AON to L_i $(1 \leq i \leq k)$. The same notation is applied for the flows between the AON and R_i.

4.2 Threat Model

We assume that attackers are unable to control all the proxies in the AON. Thus, attackers treat the entire AON as a blackbox. We assume that attackers have

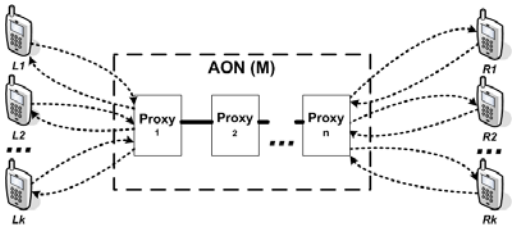


Fig. 2. System model: k pairs of users build VoIP conversations over an AON

the ability to observe and manipulate the flows entering and leaving the AON (e.g., the attacker is an Internet Service Provider (ISP)). However, we further assume that it is computational infeasible for the attackers to decrypt the voice packets in any flow. The attackers need to find out who called whom by relating the flows on both side of the AON. *For example, let us assume that* L_1 *called* R_2. To confirm this, the attackers should relate $\overrightarrow{L_1M}$ with $\overrightarrow{MR_2}$, $\overrightarrow{ML_1}$ with $\overrightarrow{MR_2}$, $\overrightarrow{ML_1}$ with $\overrightarrow{R_2M}$ or $\overrightarrow{L_1M}$ with $\overrightarrow{R_2M}$ by a certain kind of pattern. Two known threats on VoIP anonymity are the following ones:

- Watermark attack [18]: Suppose a flow contains continuous voice packets P_1, P_2, ..., P_n with time stamps t_1, t_2, ..., t_n. An attacker intercepts the flow before it enters the AON. She randomly selects 2 packets: P_i and P_j and extend to 4 packets in 2 groups (P_i, P_{i+d}) and (P_j, P_{j+d}). Then the attacker can obtain the inter-packet delay (IPD) for this two groups: $IPD_1 = t_{i+d} - t_i$, $IPD_2 = t_{j+d} - t_j$. Finally, the attacker can calculate the normalized difference $IPDD = (IPD_2 - IPD_1)/2$. If the VoIP user-agents send voice packets in a constant rate, the IPDD should be symmetric centered around 0. To insert watermark into the flow, the attacker can delay P_i and P_{j+d} for a small time interval α to embed bit '1' or delay P_{i+d} and P_j to embed bit '0'. The marked flow should be forwarded to the AON and the attacker can observe the marked flow on the other side: the IPDD distribution of selected packets should be shifted from 0 by α. [18] shows that only α=3 ms delay is enough to successfully embed watermark in Skype flows. Certainly, networking delay jitter can result in error bits. Nevertheless, the attacker can select more packets in a flow under attack to increase redundancy. With introducing only 3 ms delay, it is rather difficult for proxies of the AON to detect and synchronize the delayed packets. Considering the example shown above, the watermark attack can help the attacker to correlate either $\overrightarrow{L_1M}$ with $\overrightarrow{MR_2}$ or $\overrightarrow{ML_1}$ with $\overrightarrow{R_2M}$.
- Complementary matching attack [16]: Some VoIP user-agents apply codec with silence suppression in which the user-agents either generate small size packets or does not generate packets at all during the silence period. Moreover, human conversations usually follow the "alternate in speaking and silence" rule: When one speaks, the other listens. In this way, an attacker can detect silence or speech by observing a flow. For example, if the attacker has recorded all bidirectional flows during a given time T as set $\mathbb{F}$, she can select F_i and F_j as query flows. The attacker can calculate *the pairing index value* C:

$$C(i,j,T) = \sum_{t=1}^{T} \frac{XOR(F_i[t], F_j[t])}{T} \tag{1}$$

  $F_x[t] \in 0, 1$, where 1 indicates that the flow F_x represents speech at time t and 0 indicates silence at time t. Thus, according to the "alternate in speaking and silence" rule, the higher of C(i,j,T), the higher probability that F_i and F_j

belong to one conversation. [16] demonstrates the high accuracy provided by this method. Regarding the previous example, the complementary matching attack enables attackers to correlate either $\overrightarrow{ML_1}$ with $\overrightarrow{MR_2}$ or $\overrightarrow{L_1M}$ with $\overrightarrow{R_2M}$.

4.3 Defensive Method

In this paper, our proposed defensive solution is fundamentally probabilistic. We aim to counteract against watermark attack [18] and complementary matching attack [16]. However, we do not aim to provide a solution to prevent long-term statistical attacks. We notice that the watermark attack correlates flows by exploiting normal distribution of IPDD while the complementary matching attack takes advantage of on-off flow pattern caused by silence suppression. As a result, we have a dilemma:

Dilemma 1. *If users apply silence suppression, they are vulnerable to the complementary matching attack; If users do not apply silence suppression, they are vulnerable to the watermark attack.*

If silence suppression is applied, the packet inter-arrival time for a given flow is not constant, which means the distribution of IPDD is indeterministic. This makes the watermark attack more difficult to success. However, in return, the on-off behavior on flows exposes the silence and speech period of users so that the complementary matching attack is easy to mount. On the other hand, if silence suppression is not applied, then the complementary matching attack does not work at all. Nevertheless, with constant packet inter-arrival time, the watermark attack is rather easy. Taking the dilemma into account, we propose a solution based on the "defensive dropping" concept [6]. The user-agents do not apply silence suppression but Voice Activity Detection (VAD): The user-agents generate voice packets to the AON in a constant rate whatever they detect silence or speech. However, the originator user-agent can instruct the proxies of the AON to drop some of the silence packets according to a dropping rate dr. This can be easily achieved by putting one bit ('0' for keeping and '1' for dropping) inside the encryption layer for each proxy. The proxy for dropping a silence packet is randomly selected. Dropping these silence packets introduces less negative impact on the performance of a VoIP conversation.

Theoretically, this solution can decrease the success ratio of both two attacks. Firstly, it weakens the timing relationship between flows entering the AON and flows leaving it. All flows entering the AON are with constant packet inter-arrival time, but all flows leaving the AON are with varying time characteristics. Furthermore, not all silence packets will be dropped so that the on-off behavior on flows are still unclear, which means the complementary matching attack is more difficult. Nevertheless, the amount of dropped packets depends on the dropping rate dr. In extreme cases, either $dr = 0\%$ (no silence packets should be dropped) or $dr = 100\%$ (all silence packets should be dropped) violate our design. We demonstrate our simulation to show the impact of the value of dr to the two attacks in the next section.

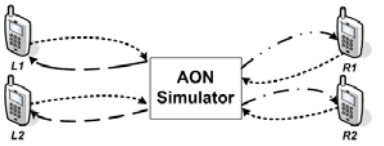


Fig. 3. The AON simulator accepts padded flows and produces simulated outgoing flows

4.4 Simulation

To find out whether the proposed solution can effectively prevent the attacks mentioned above and how the *dr* affects the result, we did a series of simulations.

Raw data preparation: We collected trace files recorded from 2 real VoIP conversations by using X-lite[5] with silence suppression enabled. There are 4 users (labeled as L_1, L_2, R_1 and R_2) participating in these 2 conversations, each of which elapsed for 10 minutes. We let L_1 *called* R_1 *and* L_2 *called* R_2. We collected the 4 voice flow traces originated by the 4 users respectively.

User-agents simulator: We then deliberately padded silence packets into each flow to normalize the packet inter-arrival time. In this way, the time intervals between any two adjacent packets are nearly equal. The padded flows simulate the 4 flows from user-agents to AON: $\overrightarrow{L_1M}$, $\overrightarrow{L_2M}$, $\overrightarrow{R_1M}$ and $\overrightarrow{R_2M}$. Furthermore, regarding n proxies of AON, each packet in these flows contains n 1-bit flags indicating whether such a packet should be dropped and which proxy should drop it. Only padded silence packets can be randomly selected for being dropped according to a dropping rate *dr*. Speech packets should not be selected.

AON simulator: We implemented a AON simulator using Perl language. The AON simulator accepts the 4 padded flows ($\overrightarrow{L_1M}$, $\overrightarrow{L_2M}$, $\overrightarrow{R_1M}$ and $\overrightarrow{R_2M}$), drops the packets marked by user-agents and generates the corresponding flows leaving the AON ($\overrightarrow{ML_1}$, $\overrightarrow{ML_2}$, $\overrightarrow{MR_1}$ and $\overrightarrow{MR_2}$). The AON simulator is based on an ideal environment that no delay, jitter occurs during the transmission.

Figure 3 shows an overview of our test. In this way, we use our simulators to simulate the flows entering the AON as well as the flows leaving it. Thus, there are 8 flows in total. We further simulate the attacks with varying dropping rates (*dr*) on the 8 flows to investigate whether the attacks can still be successfully mounted.

In the first simulation, we investigate the watermark attack. We embed a 4-bit unique watermark with 50 redundancy for each flow entering the AON ($\overrightarrow{L_1M}$, $\overrightarrow{L_2M}$, $\overrightarrow{R_1M}$ and $\overrightarrow{R_2M}$). The packets for watermarking are randomly selected from the first 500 packets in each flow. We embed watermarks with a delay of only 10 ms ($\alpha = 10ms$). Then we use our AON simulator to simulate the corresponding flows leaving the AON ($\overrightarrow{ML_1}$, $\overrightarrow{ML_2}$, $\overrightarrow{MR_1}$ and $\overrightarrow{MR_2}$). We verify whether the same watermark can be recovered from these flows again.

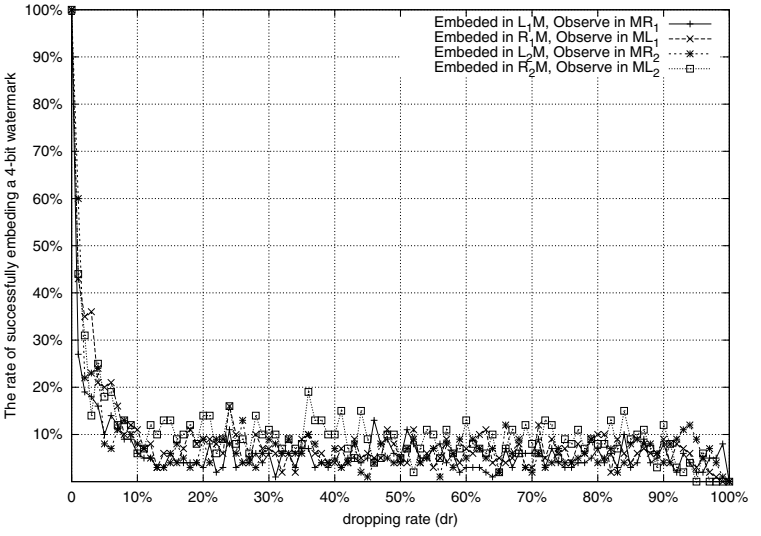


Fig. 4. The relationship between the dropping rate dr and the success ratio of watermark attack

We scale the dropping rate (dr) from 0% to 100%. The simulation for each dr is repeated for 100 times, and the result is shown in Figure 4. We find that with $dr = 0\%$, all the watermark can be successfully recovered. Thus, the success ratio of embedding a watermark is 100%. However, as long as $1\% \leq dr \leq 10\%$, the success ratio decreases significantly. Moreover, when $10\% \leq dr \leq 100\%$, the success ratio repeatedly varies between 0% and 20%.

In the second simulation, we focus on the complementary matching attack. As introduced above, the flows entering the AON ($\overrightarrow{L_1M}$, $\overrightarrow{L_2M}$, $\overrightarrow{R_1M}$ and $\overrightarrow{R_2M}$) are with a constant packet inter-arrival time. Thus, the on-off pattern of these flows are not distinguishable for attackers. As an alternative, attackers can still try to correlate the flows leaving the AON ($\overrightarrow{ML_1}$, $\overrightarrow{ML_2}$, $\overrightarrow{MR_1}$ and $\overrightarrow{MR_2}$), since some silence packets will be dropped for these flows. In this way, we use the Equation 1 to calculate the pairing index C for 4 pairs of flows: $\overrightarrow{ML_1}$ with $\overrightarrow{MR_1}$, $\overrightarrow{ML_1}$ with $\overrightarrow{MR_2}$, $\overrightarrow{ML_2}$ with $\overrightarrow{MR_1}$ and $\overrightarrow{ML_2}$ with $\overrightarrow{MR_2}$.

The value of the pairing index C for each correlating attempt is plotted in Figure 5. We find that the pairing indices are low and nearly equal when the dropping rate dr is between 0% and 10%. The pairing indexes increase with the dr. However, the pairing index (C) of $\overrightarrow{ML_1}$ with $\overrightarrow{MR_1}$ and $\overrightarrow{ML_2}$ with $\overrightarrow{MR_2}$ increase significantly faster than the pairing index of $\overrightarrow{ML_1}$ with $\overrightarrow{MR_2}$ and $\overrightarrow{ML_2}$ with $\overrightarrow{MR_1}$. Especially when the $dr = 100\%$, the pairing index for $\overrightarrow{ML_1}$ with $\overrightarrow{MR_1}$ and $\overrightarrow{ML_2}$ with $\overrightarrow{MR_2}$ are around 80% and 75% respectively, which are distinguishable higher than the value of $\overrightarrow{ML_1}$ with $\overrightarrow{MR_2}$ and $\overrightarrow{ML_2}$ with $\overrightarrow{MR_1}$ (only 53% and 50% or so). In this case, it is easy to find out that L_1 called R_1 and L_2 called L_2.

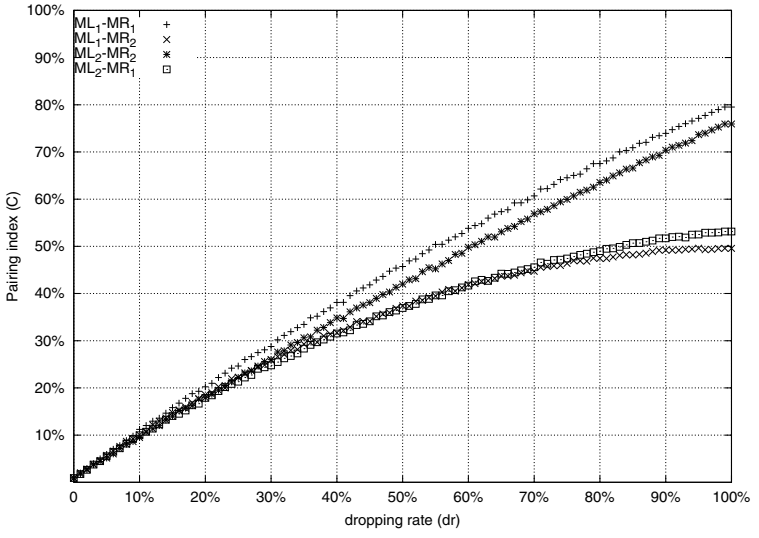


Fig. 5. The relationship between the dropping rate dr and the success ratio of complementary matching attack

The result of our experiments shows that the value of dropping rate (dr) should be carefully selected. If it is set too low, the solution will be still vulnerable to the watermark attacks. However, if it is set too high, the solution is vulnerable to the complementary matching attack. Fortunately, from the result shown in Figure 4 and Figure 5, we can find an optimal range for dr (e.g., from 5% to 10%), which is a trade-off point.

5 Further Threats

This section discusses further classical attacks on this solution.

- Passive attack: The attackers observe the unique features between the flows entering and the flows leaving the AON to pair the flows. Our solution enables that all the flows entering the AON share the same features (e.g., the equal packet size and packet inter-arrival time) and all the flows leaving the AON have different features (e.g., packet inter-arrival time) in a highly indeterministic manner. In this way, passive attack is not easy to launch.
- Replay attack: The attackers can record several packets from a flow entering the AON and replay these packets later to the AON. Our solution is unable to prevent this attack. Nevertheless, in future work, we consider to assign a time stamp for each generated voice packet. Voice conversation has a high requirement on the real time. Usually, a voice packet arriving too late, saying after 400 ms, is considered less useful [2]. In this way, we can arbitrarily set a timeout threshold to drop late-arrival packets to prevent replay attack.

– Short-term intersection attack: Given k pair of users are making conversations over an AON. Meanwhile, a new conversation is built over the AON between A and B, who are not belong to the k pair of users. Thus, it is easy to find out that A called B. We leave this problem to be solved by signaling services: the signaling service should coordinate a number of users to establish (and terminate) conversations over the AON simultaneously as a batch to achieve required anonymity set.

6 Conclusion and Future Work

Considering the unique properties of VoIP voice flows, we focus on an AON-based VoIP anonymity solution to defend against the attackers who are interested in "who called whom". In this paper, we especially consider two known attacking methods on VoIP anonymity: watermark attack, which encodes and decodes watermark on the flows of both sides of the AON by slightly packets delay; and complementary matching attack, which pairs flows by the on-off traffic mode and human conversation patterns. We propose a solution based on defensive dropping concept: The user-agent randomly marks a certain amount silence packets and these silence packets will be dropped by the AON. In this way, the relationship between corresponding flows can be obscured. The result of our simulation shows that the effectiveness of this solution highly depends on the dropping rate (dr). We finally discuss more potential attacks regarding this solution.

The scope of this paper is limited in the VoIP media layer. Nevertheless, media layer and signaling layer are usually viewed as a whole for VoIP applications. Thus, our future work will extend the scope to the anonymity protection on signaling layer. For example, how the signaling service can coordinate at least k pairs of users to establish (and terminate) voice flows over the AON simultaneously to achieve required anonymity set. Moreover, we are going to investigate the performance, usability aspects of this solution in the future.

References

1. G.711, http://www.itu.int/rec/T-REC-G.711/e (visited October 21, 2009)
2. ITU-T. Recommendation G.114 - One-way Transmission Time (2003)
3. Skype, www.Skype.com (visited October 21, 2009)
4. Speex, http://www.speex.org/ (visited October 21, 2009)
5. X-lite, http://www.counterpath.com/x-lite.html (visited November 15, 2009)
6. Wang, C., Levine, B.N., Reiter, M.K., Wright, M.: Timing attacks in low-latency MIX systems (extended abstract). In: Juels, A. (ed.) FC 2004. LNCS, vol. 3110, pp. 251–265. Springer, Heidelberg (2004)
7. Berthold, O., Federrath, H., Köpsell, S.: Web MIXes: a system for anonymous and unobservable Internet access. In: Federrath, H. (ed.) Designing Privacy Enhancing Technologies. LNCS, vol. 2009, pp. 115–129. Springer, Heidelberg (2001)
8. Chaum, D.L.: Untraceable electronic mail, return addresses, and digital pseudonyms. Commun. ACM 24(2), 84–90 (1981)

9. Melchor, C.A., Deswarte, Y., Iguchi-Cartigny, J.: Closed-circuit unobservable Voice over IP. In: Proceedings of the 23rd Annual Computer Security Applications Conference (ACSAC '07), Los Alamitos, CA, USA, pp. 119–128. IEEE Computer Society, Los Alamitos (2007)
10. Peterson, J.: A privacy mechanism for the Session Initiation Protocol (SIP), RFC 3323 (2002)
11. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., Schooler, E.: SIP: Session Initiation Protocol, RFC 3261 (2002)
12. Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V.: RTP: A transport protocol for real-time applications, RFC 3550 (2003)
13. Shen, C., Schulzrinne, H.: A VoIP privacy mechanism and its application in VoIP peering for voice service provider topology and identity hiding. Technical report (2008)
14. Srivatsa, M., Liu, L., Iyengar, A.: Preserving Caller Anonymity in Voice-over-IP Networks. In: Proceedings of the 2008 IEEE Symposium on Security and Privacy (SP '08), Washington, DC, USA, pp. 50–63. IEEE Computer Society, Los Alamitos (2008)
15. Srivatsa, M., Liu, L., Iyengar, A.: Privacy in VoIP networks: A k-anonymity approach. In: Proceedings of the 28th IEEE Conference on Computer Communication (INFOCOM '09), Washington, DC, USA. IEEE Computer Society, Los Alamitos (2009)
16. Verscheure, O., Vlachos, M., Anagnostopoulos, A., Frossard, P., Bouillet, E., Yu, P.S.: Finding "Who is talking to whom" in VoIP networks via progressive stream clustering. In: Proceedings of the 6th International Conference on Data Mining (ICDM '06), Washington, DC, USA, pp. 667–677. IEEE Computer Society, Los Alamitos (2006)
17. Vlachos, M., Anagnostopoulos, A., Verscheure, O., Yu, P.S.: Online pairing of VoIP conversations. The VLDB Journal 18(1), 77–98 (2009)
18. Wang, X., Chen, S., Jajodia, S.: Tracking anonymous peer-to-peer VoIP calls on the Internet. In: Proceedings of the 12nd ACM conference on Computer and communications security (CCS '05), pp. 81–91. ACM, New York (2005)
19. Wright, C.V., Ballard, L., Coull, S.E., Monrose, F., Masson, G.M.: Spot me if you can: Uncovering spoken phrases in encrypted VoIP conversations. In: Proceedings of the 2008 IEEE Symposium on Security and Privacy (SP '08), Washington, DC, USA, pp. 35–49. IEEE Computer Society, Los Alamitos (2008)
20. Wright, C.V., Ballard, L., Monrose, F., Masson, G.M.: Language identification of encrypted VoIP traffic: Alejandra y Roberto or Alice and Bob? In: Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium (SS '07), Berkeley, CA, USA, pp. 1–12. USENIX Association (2007)
21. Zopf, R.: Real-time Transport Protocol (RTP) payload for Comfort Noise (CN), RFC 3389 (2002)

SIP Proxies: New Reflectors in the Internet

Ge Zhang[1], Jordi Jaen Pallares[2],
Yacine Rebahi[2], and Simone Fischer-Hübner[1]

[1] Karlstad University, Karlstad, Sweden
[2] Fraunhofer FOKUS, Berlin, Germany

Abstract. To mitigate identity theft in SIP networks, an inter-domain authentication mechanism based on certificates is proposed in RFC 4474 [10]. Unfortunately, the design of the certificate distribution in this mechanism yields some vulnerabilities. In this paper, we investigate an attack which exploits SIP infrastructures as reflectors to bring down a web server. Our experiments demonstrate that the attacks can be easily mounted. Finally, we discuss some potential methods to prevent this vulnerability.

1 Introduction

The current most widely used authentication method for SIP is based on HTTP hash digest mechanism. Therefore SIP users need to share secrets (e.g., user account, password) with their home domains for message source authentication. However, such shared secrets are usually not distributed to other domains. Thus, identity misuse in multi-domain environments is possible, and accordingly will lead to some security problems such as VoIP SPAM and caller impersonation. To address this problem, J. Peterson, et al., [10] proposed a certificate based inter-domain authentication mechanism: An originator SIP domain signs an outgoing message with its private key and a recipient SIP domain verifies the signature of the incoming message. Thus, the recipient SIP domain needs to have the certificate of the originator domain for verifying messages. Nevertheless, there is a serious problem in the designed certificate distribution scheme: A recipient SIP proxy should download the certificate according to a URL address specified in the SIP message. *However, at the moment of certificate downloading, the SIP message is still unverified and untrusted.* Therefore, this mechanism offers an opportunity for an attacker to let a SIP proxy connect to an arbitrary remote host. In our previous work [15], we demonstrated an attack which exploits the time for certificate downloading to block a victim SIP proxy. In this paper, we aim to investigate another attack: The attacker can collect a list of SIP proxies as reflectors, then use the reflectors to flood a victim web server. The results of our experiments demonstrate that this attack can cause a serious impact, especially on a web server which supports HTTPS protocol.

The idea behind this attack is that a SIP proxy needs to "pull" a certificate according to a unverified request. To prevent the problem, we modified distribution method to "push" a certificate: An originator should actively provide its

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 142–153, 2010.

domain certificate. Our contribution in this paper is two-fold: First, we investigate a reflector DoS attack generated by the certificate distribution scheme described in RFC 4474. Secondly, we address improved schemes to deal with the threat.

The current paper is organized as follows: Section 2 provides an overview of SIP as well as its inter-domain authentication mechanism. Section 3 investigates the threats caused by the inter-domain authentication mechanism. Countermeasure solutions are discussed in Section 4. We summarize related work in Section 5, and in Section 6, we provide a conclusion.

2 Background of SIP and RFC 4474

SIP [12] is a text-based protocol designed to establish, modify, or terminate a session (e.g., VoIP calls) between two or more partners. Several SIP networking components are essential to be introduced: User Agent (UA), proxy server, registrar server. A UA represents a communication endpoint. According to its role in a communication, a UA can be a UA client (UAC) which initializes a SIP message, or a UA server (UAS) which receives a SIP request and makes a response. A proxy server (or proxy) forwards SIP messages between different SIP components in the network. A registrar server is where users login and announce their availability in the network. Generally, some security checks (e.g., authentication) are enforced on a proxy. SIP users are identified by their Uniform Resource Identifiers (URI) [2]. A URI consists of a domain name and a user name (e.g., `sip:Alice@kau.se`). The format of a SIP message is similar to HTTP protocol, with message headers and a message body with its corresponding values (e.g, *From: Alice@kau.se* is to denote the sender of a message). SIP messages are distinguished as two categories: SIP requests (e.g., REGISTER, INVITE, ACK, BYE, etc) and SIP responses (1xx, 2xx, etc). In this paper, we focus on inter-domain requests. We define that a SIP request transverses over different SIP domains as *inter-domain request*. The proxy which initializes an inter-domain request is defined as *originator proxy*, and its domain is named as *originator domain*. While the destination proxy is named as *recipient proxy*, within a domain defined as *recipient domain*.

VoIP attackers frequently use fraud SIP identities in order to be untraceable. As a countermeasure, RFC 3261 [12] proposed a HTTP digest based authentication mechanism using shared secrets (e.g., user name and password). Unfortunately, this solution does not work in multi-domain environments since the secretes are usually not distributed. In this way, attackers can easily fraud their SIP identities in multi-domain environments without being detected. To prevent such a problem, an inter-domain authentication mechanism based on certificates has been proposed in RFC 4474 [10] and illustrated in Figure 1. Let us say that a caller sends a calling request to a callee in a multi-domain scenario. The originator domain signs the digest of this request with its private key. The generated signature is encoded within a new header filed *Identity* appended to the original request. Moreover, the originator domain attaches another new header field

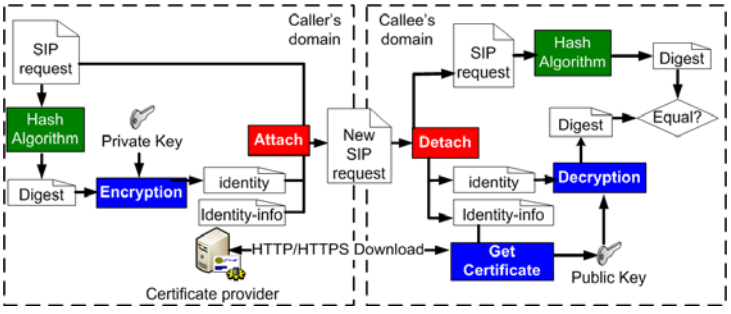


Fig. 1. The inter-domain authentication mechanism defined in RFC 4474

Identity-info, which contains the Uniform Resource Locator (URL) [3] indicating where the certificate can be downloaded from[1] as well as authentication parameters (e.g., the algorithm used for verification). This modified request is then forwarded to the recipient domain. Thus, the recipient domain will first check whether the certificate for the originator domain can be found locally (e.g., it has already been cached). If the certificate is not cached, the proxy has to download it according to the URL given in the *Identity-info* field. If the downloading does not succeed or the downloaded certificate is invalid, the recipient domain will neglect this request and reply with a negative response. Otherwise, the proxy uses the public key extracted from the certificate to verify the signature in the *Identity* header field. The request will be finally forwarded to the callee only if the verification is successful.

3 Reflecting Attacks Using SIP Proxies

We assume that all the SIP proxies mentioned along this paper are implemented according to the authentication scheme specified in RFC 4474. A serious problem of this authentication scheme is the fact that the recipient proxy cannot know whether an incoming inter-domain request is spoofed or not until the authentication operation takes place. To do so, the recipient proxy needs the corresponding certificate. If the certificate is not cached, the proxy has to download it from the location specified in this *Unverified* request. In this way, an attacker can cheat one or many proxies to let them connect with any host for "certificate downloading". This vulnerability can lead to DoS attacks on a web server, despite of whether this web server really supports domain certificate downloading or not. In this paper, we focus on a system model which is a large-scaled network consists of numerous SIP proxies and web servers (e.g., the Internet). We assume that the attacker can connect to the network and craft SIP requests to a number of SIP proxies. The goal of the attacker is to reduce the availability of a targeted web server.

[1] According to RFC 4474, the URL indicated in the *Identity-info* header field and the originator domain indicated in the *From* header field must be matched.

Fig. 2. The testbed for measuring resource consumption

Attack description. DoS attacks generally take advantage of unbalanced resource occupation between client side and server side. To understand the situation of resource consumption in the given context, we performed a measurement with a testbed configured in Figure 2.

The testbed consists of three networking components as follows:

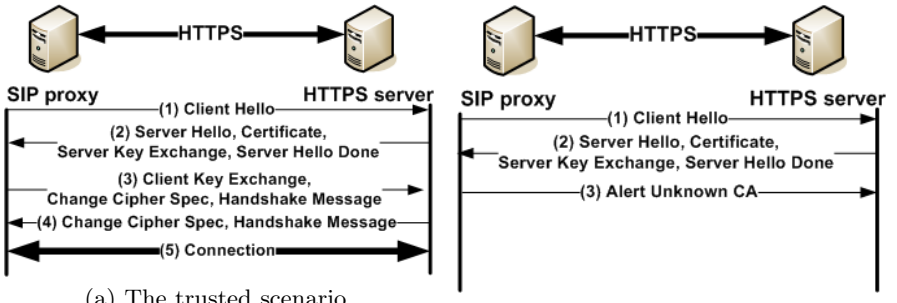
1. A web server: In this test, we employ Apache [1], a widely deployed web server program. We configure it to support both HTTP and HTTPS connections. To enable HTTPS connections, we generate a self-signed X.509 RSA certificate on the web server[2]. The size of this HTTPS certificate is 806 bytes with 1024 bits public key.
2. A SIP proxy: We employ SIP Express Router (SER) [13] as our SIP proxy, which is configured with 16 parallel process queue.
3. An attacker: The attacker generates crafted inter-domain SIP requests. The attacker can designate the scheme name (http://, or https://) of the *Identity-info* URL in a SIP request. The attacker is implemented using SIPp [14], an open source SIP traffic generation tool.

Hardware: Each component mentioned in this paper is independently established on a Pentium 4 machine with 512 Mb RAM running the linux Ubuntu operating system, with 100 Mbps local network access.

Our measurement investigates the CPU overhead on the web server and the attacker respectively for attacking. The purpose of this test is to find out the difficulty of mounting this attack. A DoS attack targets resources of the victim. Moreover, the DoS attack is reasonable for the attacker only if it is not required to spend a lot of resources in equipment to achieve the task. During the following test, we investigate the attacker and the proxy CPU utilization according to different attacking rate configured for the attacker. Our measurement is repeated for three different circumstances:

1. HTTP scenario: The attacker floods the proxy with requests in which the *Identity-info* header fields begin with the input "http://". In this way, the proxy will need to connect to the web server using HTTP protocol.

[2] This certificate is only used for web server authentication during HTTPS handshake, not for SIP inter-domain authentication purpose. To distinguish it from the certificate for SIP authentication, we call it as HTTPS certificate in the rest of this paper.



(a) The trusted scenario

(b) The untrusted scenario

Fig. 3. Establishing a HTTPS connection between the proxy and the web server

2. HTTPS (Trusted) scenario: The attacker sends requests to the proxy in which the *Identity-info* header fields begin with scheme name "https://". The proxy first receives the certificate from the web server and it trusts this self-signed HTTPS certificate (e.g., it is from a trusted root CA). Therefore, the proxy will exchange session keys and build HTTPS connections with the web server. The procedure is illustrated in Figure 3(a).
3. HTTPS (Untrusted) scenario: Similarly, the attacker send the proxy with requests in which the *Identity-info* header fields begin with scheme name "https://". However, the proxy does not trust the self-signed HTTPS certificate of the web server in this scenario. In this way, the handshake is interrupted after the SIP proxy checks the HTTPS certificate sent from the web server. Thus the HTTPS connection and downloading cannot be successful. We depict this procedure in Figure 3(b).

Figure 4 compares the average CPU utilization on the attacker and the web server with different attacking rates and scenarios. We can hardly find any variability of the CPU utilization on the attacker whatever the used attacking rate and the scenario type. On the other hand, The CPU utilization on the web server stayed 4% or so for the HTTP scenario. The value increases from around 4% to almost 74% for the HTTPS (trusted) scenario. However, this value surges from 4% to around 72% for the HTTPS (Untrusted) scenario. With an increasing attack rate, the CPU utilization on the web server surges for the HTTPS scenarios. The CPU utilization on the web server is much higher than it in the attacker in the HTTPS scenario. This result is mainly due to the random number generating and the cryptographical operations, which consume a lot of CPU resources. As shown in the results, it is easy to consume a significant resources of a HTTPS web server by using little resources on the attacker side. This unbalanced situation might be due to the following reasons:

1. HTTP scenario: TCP was specifically designed to provide a reliable end-to-end communication with a "flow control" method. However, in return, the "flow control" method consumes additional bandwidth. On the other hand, UDP, without a "flow control" method, is more efficient than TCP.

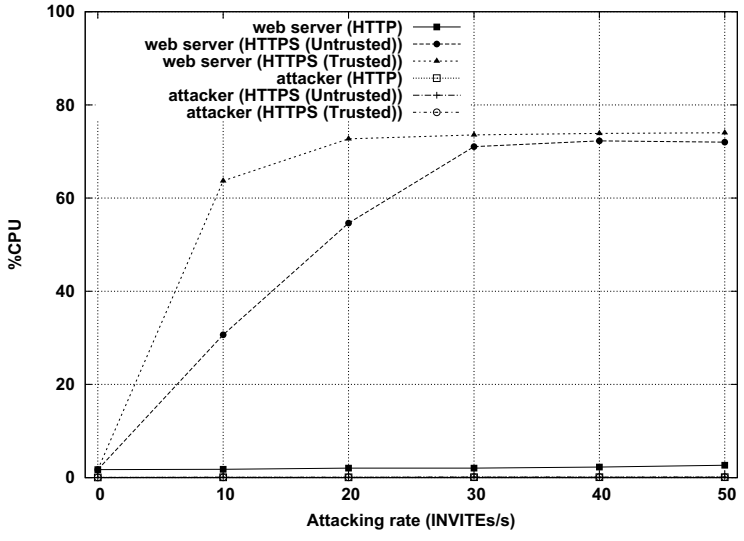


Fig. 4. The consumption of CPU resources on different components

SIP communications can be built on either TCP or UDP. During the test, the attacker sends requests to the SIP proxy over UDP while the SIP proxy builds HTTP connections with the web server over TCP. That is why that a little more resource consumed on the web server than on the attacker. Anyway, the gap is rather small.

2. HTTPS (Trusted) scenario: According to Figure 3(a), it is clear that much more resources are needed for building up a HTTPS connection since it spans over several steps. For example, to exchange session keys with the proxy, the web server has to perform a lot of computation that occupies a big part of the CPU resources.
3. HTTPS (Untrusted) scenario: Although the connection cannot be setup in this scenario, the CPU utilization on the web server is still high. From Figure 3(b), the server key exchange is performed in step 2. The proxy cancels the transaction in step 3 with an alert message, however, even in this case, the server has to perform a key exchange operation. By default, it is an RSA key exchange that takes place by creating a temporary RSA key pair. This procedure consumes a lot of CPU resources.

In this way, we assume that an attacker has a target HTTPS web server with URL: `https://www.victim.com`. To bring it down, an attacker just needs to select a list of SIP proxies in the network as reflectors. Then, the attacker floods the proxies with numerous crafted SIP requests in which the *identity-info* header fields include the same value: `https://www.victim.com/[random_string]`. As a consequence, the SIP proxies will continuously build connections with the victim web server and the CPU resources of the web server will be depleted.

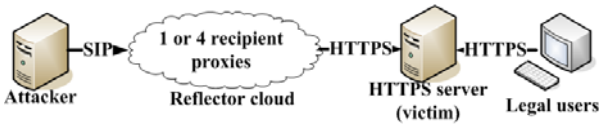
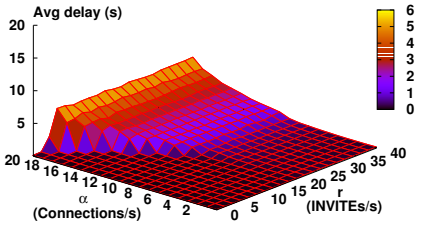
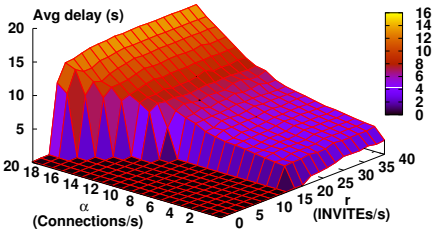


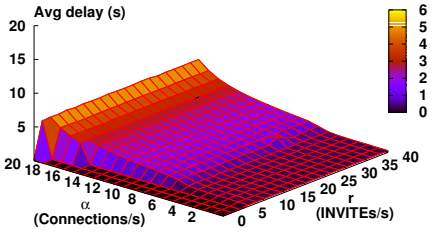
Fig. 5. The testbed for distributed reflector attacks on HTTP web servers



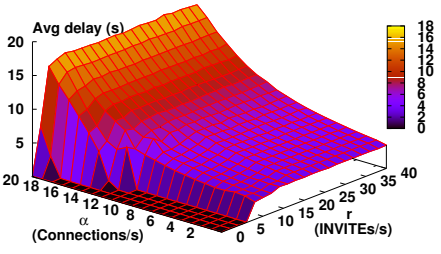
(a) 1 SIP proxy in untrusted scenario



(b) 4 SIP proxies in untrusted scenario



(c) 1 SIP proxies in trusted scenario



(d) 4 SIP proxies in trusted scenario

Fig. 6. The victim web server response delays for different parameters

Experiments setup. To investigate the attack in detail, we setup our testbed in the way depicted in Figure 5. The testbed consists of the following components:

1. The attacker: The attacker works in the same way as in the previous test. It generates SIP requests with the *identity-info* header field including the URL of the victim's HTTPS web server. The attacking rate r (INVITE/s) can vary from 0 to 40.
2. Reflector cloud: The attacker employs SIP proxies as attacking reflectors. As soon as the SIP proxies receive requests from the attacker, they will try to establish HTTPS connections with the victim web server. In this experiment, we setup the reflector cloud with either one or four SIP proxies. Please notice that the attacking rate r (INVITEs/s) is for the whole cloud. Therefore, if there is only one proxy in the cloud, then the proxy receives attacking requests at the rate r. However, if there are four proxies in the cloud, then each proxy receives the attacking requests at a rate equal to $r/4$.
3. HTTPS web server: The configuration of the web server is similar to the previous test. However, it only supports HTTPS protocol in this test.

4. Legitimate users: We employ httpref [6], a web server performance measurement tool, to simulate legitimate users, which constantly build HTTPS connections with the victim web server. The performance of the victim web server can be demonstrated by observing the delay of connecting the legitimate users to the web server. The legitimate users can be configured to constantly build HTTPS connections at a rate α (connections/s), varying from 1 to 20.

In the test, we measure the average connecting delay between the legitimate users and the web server. The result is shown in Figure 6. It shows that average delay of the connection between legitimate users and the web server is about few milliseconds without attack (when $r = 0$), no matter what the other parameters are. However, under attack, the delay can be up to 14 (s) or 16 (s) if the attacker selects 4 reflector proxies. By using only one reflectors, the attack impact is not so high but still serious: The maximal delay is around 4-5 (s). Generally, the attacking impact in trusted scenarios is slightly higher than the one in untrusted scenarios. Furthermore, a higher attacking rate and more legal connections will certainly lead to a higher attacking impact on the web server. A connection attempt with 16 seconds delay is usually regarded as unsuccessful for most web browsers. As a result, legitimate users cannot visit a HTTPS web server if the server is under flooding from the reflector proxies. It is also worth to mention that this kind of attack can be easily performed in reality: An attacker first collects a list of SIP proxies in the Internet as reflectors. Then the attacker floods the reflectors with crafted requests containing certificates location at a victim HTTPS server. In this way, the HTTPS server can be brought down due to too many connecting requests from the reflector proxies.

4 Countermeasures

4.1 Unified Certificate Repository

One solution is to build a global unified certificate repository to store certificates for different SIP domains. Thus, the location of the unified certificate repository can be hard-coded on a SIP proxy, instead of being specified by the requests. In this way, wherever the SIP requests come from, the recipient proxy should download certificates from such a known unified certificate repository. As a result, attackers no longer can arbitrarily specify which host a proxy should contact with. Unfortunately, however, a unified certificate repository is not scalable and might be difficult to be accepted by service providers in the Internet.

4.2 Alternative Authentication Methods

As alternatives, other security mechanisms at the transport layer or network layer can also be considered as candidates for inter-domain authentication purpose. For example, establishing a Transport Layer Security (TLS)[5] connection or an IPSec[8] tunnel between an originator proxy and a recipient proxy. As for

the use of TLS within this context, RFC 4474 mentions that while the creation of a chain of transitive authentication between the originating UA, the local proxy and the remote proxy via TLS may work well in some architectures, transitive trust is inherently weaker than an assertion that can be validated end-to-end. In the case a SIP request is crossing multiple SIP domains, this transitive trust becomes even less effective. RFC 3325[7] proposes a solution to this problem implementing the *P-Asserted-Identity header*. However, this solution only allows hop-by-hop trust between intermediaries, and no end-to-end authentication. In addition to that, it assumes a managed network of nodes with strict mutual trust relationships, an assumption that is incompatible with widespread Internet deployment. Similarly, IPSec cannot assure an end-to-end protection, either.

Furthermore, TLS is connection-oriented and requires also a handshake. In contrast, the mechanism described in RFC 4474 is connectionless and does not require any handshake operation. Moreover, TLS aims at providing confidentiality, integrity and authentication for the entire traffic. Nevertheless, the mechanism proposed by RFC 4474 only focuses on providing authentication and integrity for selected parts of a SIP request. In this way, the RFC 4474 mechanism, a lightweight protection method, introduces less performance overhead than the other methods. With the just discussed benefits related to end-to-end protection and performance, the RFC 4474 mechanism cannot be easily replaced by TLS or IPSec.

4.3 Encoding Certificates into the SIP Requests

According to RFC 4474, it is the recipient proxy which is responsible for downloading the certificate of the originator proxy. This action is qualified as *pulling* a certificate by the recipient proxy. This *pull* action produces the vulnerability, which makes the victim proxy connect to a networking host arbitrarily set by the attacker. In contrast, we address an alternative way for certificate transferring in which an originator proxy actively *pushes* a certificate to the recipient proxy.

Method. Figure 7 illustrates the procedure of a potential "push" scheme. Firstly, we assume that a UA can receive a copy of the domain certificate after registration to this domain. When an originator proxy receives an inter-domain request from its user, the proxy will sign this request, and add the signature into the request as a new header *Identity*. The proxy will also attach the request with a new header *identity-info*. However, different to the one defined in RFC 4474, this *Identity-info* only contains verification parameters (e.g., which algorithm should be used). There is no URL contained in the *Identity-info* headers any more. The recipient proxy will search the domain certificate of the originator domain in the local cache after getting this request. If the certificate of the originator domain is cached locally, the proxy will then verify the request by using the cached certificate. Otherwise, the recipient proxy will send to the originator proxy a "4XX" response to indicate that a certificate is required for further processing. The UAC in the originating domain will then resend the request along with the domain certificate. Again the originator SIP proxy will sign

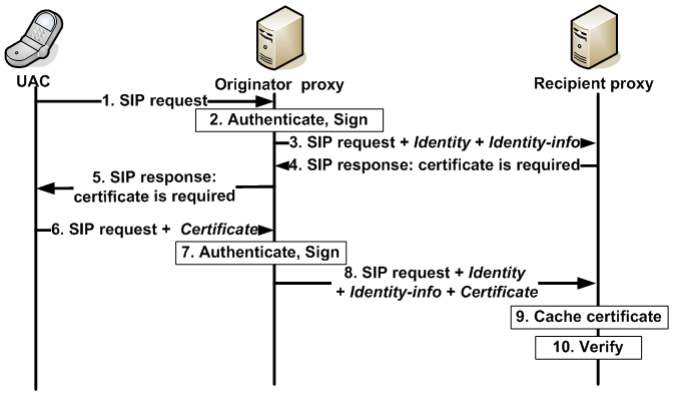


Fig. 7. An improved certificate distribution scheme

the request and attach *Identity* and *Identity-info* headers. Finally, the recipient can get the certificate from the request and cache it for a period of time. In this way, the cache is *pushed* by the UAC, instead of being *pulled* by the recipient proxy, which means the attacks addressed above are prevented. We suggest that the requests with certificates (Step 6 and 8) should be transmitted over TCP. The reason will be explained in this section later.

Further discussion. The certificate "pushing" scheme still faces different challenges. First, in order to be transmitted properly the certificate needs to be encoded in base64. Then we have to deal with the maximum package size for each SIP package: while the maximum size of a UDP packet is 64 Kbytes including the IP and UDP headers, the maximum transmission unit (MTU) depends on the kind of network we are using but we can assume that above 1500 bytes in an ethernet network we will have to deal with packet fragmentation. SIP allows both TCP and UDP transmission protocols. In case we are using TCP for the transmission we can count on that the network level will re-assemble the fragmented SIP packages. If we are using UDP, the reception of the fragmented packet may encounter some errors.

The size of a SIP message is usually around 1000 bytes, then the size of the certificate cannot exceed 500 bytes if we want to send it over ethernet using UDP. With this limitation, we can look at RSA type certificates with a public key of 2048 bytes and take the PEM encoding of it which is already base64. Such certificates will have an approximate size of 1350 bytes which is too much for a UDP transmission. If we take a look at Elliptic Curve Cryptography (ECC) [4] using for example a curve of type secp256k1, the approximate length of the PEM base64 encoded certificate will be around 782 bytes, which also makes it not suitable for transmission over UDP. Regarding RSA and ECC certificates, NIST recommends RSA keys of 3072 bytes for security beyond the year 2030, whilst in the case of ECC to achieve the same level of security requires a key length of 224 bytes [11].

As proposed, the infrastructure would need a caching function for the certificates and the first time they are sent it must be done over TCP. On the other hand, the idea of including certificates in the SIP messages could also be re-used to distribute the Certificate Revocation Lists (CRL) for the revoked certificates of SIP proxy servers. The CRL distribution cost would be lowered by using SIP headers to transport certificates over TCP. On the other hand, in order to support different certificate types, the RFC4474 will need to be extended to signal which kind of signature algorithm is used to include the ECC certificates which use the ECDSA signature algorithm. The current standard only specifies that all implementations must support the RSA-SHA1 signature algorithm.

5 Related Work

Using reflectors for distributed denial-of-service attacks is not a new concept. Paxson [9] summarized several types of attacks using reflectors. In his paper, a reflector is defined as "any IP host that will return a packet if sent a packet." It can be a DNS server, a web server or a IP router, etc. Thus, attackers need to collect a large number of reflectors (e.g., 1 million) and send to the reflectors spoofed requests announced coming from a victim. The reflectors will in turn generate responses from themselves to the victim, so that the resource of the victim (e.g., bandwidth) is occupied. Different from these classical reflecting attacks, the attack proposed in this paper takes SIP proxies as reflectors. The attack can be successfully mounted with only a small amount of reflectors.

Our previous work [16] and [15] described some other problems and vulnerabilities of this mechanism. [16] addressed the confidentiality issues of certificate cache, which is especially vulnerable to timing attacks. In this way, attackers can observe and compare the timing required for different requests to find out the calling history between SIP domains. [15] proposed a threat by exploiting a long time for certificate downloading to block a victim SIP proxy.

6 Conclusions

With an inappropriate design of the certificates distribution scheme, the inter-domain authentication mechanism for SIP is vulnerable to Denial of Service attacks. An attacker can consume the resource of a HTTPS server by simply selecting a list of SIP proxies as reflectors. The result of our experiments demonstrated that a HTTPS web server can be easily taken down in this way. Finally, we proposed an improved certificate distribution scheme to prevent this vulnerability. More evaluation of the proposed scheme will be done in the future.

References

1. Apache, HTTPd server, http://httpd.apache.org/ (visited August 16, 2009)
2. Berners-Lee, T., Fielding, R., Masinter, L.: Uniform Resource Identifier (URI): Generic syntax, RFC 3986 (2005)

3. Berners-Lee, T., Masinter, L., McCahill, M.: Uniform Resource Locators (URL), RFC 1738 (1994)
4. Blake-Wilson, S., Brown, D., Lambert, P.: Use of Elliptic Curve Cryptography (ECC) algorithms, RFC3278 (2002)
5. Dierks, T., Rescorla, E.: The Transport Layer Security (TLS) Protocol Version 1.2, RFC 5246 (2008)
6. Httperf, http://www.hpl.hp.com/research/linux/httperf/ (visited August 16, 2009)
7. Jennings, C., Peterson, J., Watson, M.: Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks, RFC 3325 (2002)
8. Kent, S., Atkinson, R.: Security Architecture for the Internet Protocol, RFC 2401 (1998)
9. Paxson, V.: An analysis of using reflectors for distributed denial-of-service attacks. SIGCOMM Comput. Commun. Rev. 31(3), 38–47 (2001)
10. Peterson, J., Jennings, C.: Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP), RFC 4474 (2006)
11. Rebahi, Y., Pallares, J.J., Nguyen, T.M., Ehlert, S., Kovacs, G., Sisalem, D.: Performance analysis of identity management in the Session Initiation Protocol (SIP). In: Proceedings of the 2008 IEEE/ACS International Conference on Computer Systems and Applications, pp. 711–717. IEEE, Los Alamitos (2008)
12. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., Schooler, E.: SIP: Session Initiation Protocol, RFC 3261 (2002)
13. SIP Express Router 2.0, http://www.iptel.org (visited September 16, 2008)
14. SIPp, http://sipp.sourceforge.net/ (visited September 16, 2008)
15. Zhang, G., Fischer-Hübner, S., Ehlert, S.: Blocking attacks on SIP VoIP proxies caused by external processing. Springer Telecommunication Systems (2009)
16. Zhang, G., Fischer-Hübner, S., Martucci, L.A., Ehlert, S.: Revealing the calling history of SIP VoIP systems by timing attacks. In: Proceedings of the 4th International Conference on Availability, Reliability and Security (ARES '09), Fukuoka, Japan, pp. 135–142. IEEE Computer Society, Los Alamitos (2009)

Analysis of Token and Ticket Based Mechanisms for Current VoIP Security Issues and Enhancement Proposal

Patrick Battistello[1,2] and Cyril Delétré[1]

[1] Orange Labs, 2 av. Pierre Marzin, 22307 Lannion Cedex, France
[2] Telecom Bretagne, 2 rue de la Châtaigneraie, 35576 Cesson Sévigné, France
{patrick.battistello,cyril.deletre}@orange-ftgroup.com

Abstract. These last few years, the security of VoIP architectures has become a sensitive issue with many vulnerability announcements. This article first aims to distinguish the threats and the applicable protection mechanisms depending on the underlying VoIP architecture. We then investigate the properties of a specific class of existing call establishment mechanisms based on tokens or tickets. In the last section, an enhancement to these mechanisms is proposed which lifts some of the previously seen limitations, especially the DoS risks, the token storage constraint or the transport impact of large tickets.

Keywords: VoIP, security, SPIT, DoS, authentication, token, ticket.

1 Introduction

VoIP is a fast-growing application, with the first deployments starting around ten years ago, first in local networks then in wide or operator networks, for both residential and professional customers. This technology claims on one hand the integration of voice and data applications within a single network architecture, and on the other hand the support of advanced services (notification, presence, WebPhone, ClickToPhone) as well as the support of media flows other than voice in peer to peer or group communications.

In parallel to these advantages, the introduction of VoIP has also brought new issues, amongst which the QoS guarantee and the security of communications [1] that require specific solutions to reach the same quality as PSTN.

This article first aims to provide a state of the art of the major threats, and the associated protection mechanisms, distinguishing the different VoIP architectures. This is treated in section 2, starting from the historical intra-domain (or single domain) context and then moving to the more tricky inter-domain (or cross-domain interconnection) one. Then, section 3 focuses on a class of call establishment signalling mechanisms based on tokens or tickets, analysing pros and cons. In section 4, we introduce the framework of a call establishment mechanism operating in the signalling plane which lifts some of the previously seen limitations, especially the DoS risks, the token storage constraint and the transport impact of large tickets. Finally, section 5 provides the conclusions.

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 154–165, 2010.

2 Current Stakes in VoIP Security

This section analyzes the major VoIP risks according to the possible architectures, because as explained in [2] there is no single VoIP model. We will begin with the intra-domain context which constitutes the historical VoIP deployments. Then we will analyze the inter-domain context which is the intra-domain natural extension, but presents increased security risks. A wealth of literature about security and threats in VoIP being available ([1], [3], [4], [5], [6]), this section will rather treat threats per architecture types than be exhaustive; distinguishing threats specific to VoIP from those which are not. For each category of architectures, we enumerate the major protection mechanisms with their pros and cons.

Beforehand we remind the reader that VoIP networks are based on several protocols. On the signalisation side, the IETF SIP protocol has progressively become the fundamental brick in association with the IETF SDP protocol for negotiating the multimedia session characteristics[1]. The media flows are conveyed via the IETF RTP protocol associated with the RTCP control protocol.

2.1 Intra-domain Context

We gather in this chapter the set of architectures characterized by the fact that VoIP communications remain confined in a same network or administrative domain, even if in each of these architectures signalisation and media gateways may be added to interconnect with the PSTN. An essential characteristic of the intra-domain context is that the operator, or administrator, is able to authenticate each entity[2] and localize it for call routing.

2.1.1 Main Architectures in the Intra-domain Context

The *VoIP initial architecture*, according to ITU-T H.323 and IETF SIP standards is made up of VoIP endpoints connected to registrar servers for endpoint registering and localization and to proxy servers for call routing between endpoints. This is a centralized architecture which can also support non-VoIP endpoints through phone adaptor or circuit to VoIP gateway.

The *extended architecture* is an extension of the initial architecture in which a same VoIP domain is distributed over several sites, or VoIP sub-networks, interconnected between themselves. The interconnection is performed either at the IP level or at the VoIP level (IP Centrex solution).

In parallel to incumbent operators, new players from the computer world have proposed alternative architectures like the *VoIP P2P architecture* where registrar and proxy functions are distributed over a set of nodes (instead of being centralized by an operator). The P2PSIP proposal [7] reuses DHT (Distributed Hash Table) concepts to route calls between the VoIP nodes of the P2P network, in association with "classical" VoIP protocols (SIP, SDP, and RTP).

Another alternative is the *VoIP web-based architecture*. It aims to simplify the access to VoIP by bringing the VoIP endpoint directly in the client's web browser; this

[1] Among the various SIP messages, the INVITE request initiates the call establishment process.
[2] Based on a unique identifier attached to each VoIP endpoint inside the domain.

new type of endpoint is called a WebPhone. This model mostly reuses websites principles and is often associated with "social networking" application. In this centralized model, the VoIP operator authenticates each WebPhone and routes calls between them using classical web-based protocols, as for example *FlashPlayer* and TLS for security aspects.

2.1.2 Threats and Protection Mechanisms in the Intra-domain Context

Several threats have been identified in previous architectures: signalling or media flows tapping, DoS, identity hijacking or endpoint compromission. Nevertheless, these threats are not linked to VoIP protocols themselves but rather to the vulnerability of the underlying layers and they are common to many applications. Therefore they are solved with "classical" protection mechanisms: system and application hardening, access filtering, flows priorization and transport over secure links. However, security protocols like TLS may raise performance issues for the servers [8] when facing a large number of endpoints. Other non-VoIP specific threats have been identified in the *VoIP P2P architecture* [9] as well as in the *VoIP web-based architecture* where automated WebPhone account creation may lead to the same undesirable consequences as in WebMail. For the latter, the CAPTCHA[3] protection has showed its limits [10], and even if enhancements are proposed [11], the balance between CAPTCHA robustness and usability is tricky to find [12].

In parallel to these threats, targeted DoS attacks threaten highly the VoIP protocols because of their complexity and the need to keep dialog and call context. Another VoIP specific threat foreseen is SPIT (SPam over Ip Telephony), due to the low cost of VoIP calls and the possibility to automate the call broadcasting (two characteristics which do not exist in the PSTN). Call automation is possible from soft-phones, thus primarily in VoIP P2P and web-based architectures but now also in operator architectures with new mobile endpoints. Nevertheless, SPIT prevention in the intra-domain context is much easier than in the inter-domain one because the administrator is able to authenticate the endpoints and usually stands in the signalling path. This way, detection algorithms as proposed in [13] are efficient, provided detection thresholds are correctly tuned.

2.2 Inter-domain Context

Initially, VoIP inter-domain call establishment was envisioned in the same way as e-mail. This first approach, noted "open model" hereafter, presents many security risks whose origins are explained by one of the Internet founders in [14]. Thus VoIP interconnections are currently set up in a more conservative way, called "closed model" (or private federation) hereafter. Before browsing these two models, we must highlight the strong regulation constraint that applies to VoIP as opposed to many web services like e-mail or social networks which are seen as "best-effort" services and thus much less regulated (up to now). According to [15], VoIP regulation poses severe constraints, especially in the inter-domain context and may turn some models off.

[3] Completely Automated Public Turing test to Tell Computers and Human Apart.

2.2.1 Open-Interconnection Model

This model is promoted mainly by IETF and assumes that IP connectivity between endpoints, proxies or domains is sufficient to establish multimedia communications. In theory, the calling endpoint (or proxy) triggers a DNS lookup based on the callee identifier to localize one of the incoming proxies in the receiving domain. It then sends the INVITE request to the receiver proxy, without any preliminary relationship needed between the sending and the receiving domains.

The first major issue here is that VoIP identifiers are designed to convey a domain part but as, explained in [16], because of the large hard-phone installed base and of the PSTN predominance, most VoIP calls originate from or terminate in a PSTN cloud. Consequently caller (or callee) identifiers lack the domain part, so inter-domain calls are tricky to route and also to verify for the receiving domain which may be misled with spoofed caller identifiers. Authors of [16] acknowledge that a public-ENUM-like system is required to turn an E.164 phone number into a routable SIP URI, but they argue that such a system will never emerge.

Other security threats related to this model are summarized in [17]; they concern the identification and localization functions of the receiver, the DoS risks over interconnection points, the tapping or degradation of signalisation or media flows conveyed over Internet and the SPIT risks. The accumulation of DoS and SPIT risks over the interconnection points is called the "pinhole problem" in [16] and constitutes the second major issue with this model.

Concerning the SPIT threat, a parallel was quickly drawn with SPAM in [18]. This synthesis shows that SPIT and SPAM have common motivations and origins[4]. By the way, similar protection principles (except content filtering) may apply: white or black lists checking, circles of trust, CAPTCHA or mathematical puzzle submission, payment at risk. These first mechanisms generally increase the complexity of the protocol exchanges and require the sources to be authenticated and stable. Unfortunately, it was shown that hackers were able to quickly change the sources of theirs attacks [19]. Another prevention technique called "consent-based" performs preliminary notification before the establishment of a first call to fetch the agreement of the receiver, thus solving the "introduction problem" explained in [18]. Finally, marking mechanisms can be used, based on several call criteria, to evaluate the probability that this one may be SPIT [20], [21]. Most of the above protection principles apply equally to the closed model described in next chapter.

Among the pro-active security mechanisms, *SIP Identity* [22] was proposed for the open-interconnection model. It is built on the same principles as the IETF DKIM protocol for e-mail: the sending domain signs the SIP INVITE request with a signature calculated over the key fields of the request and the receiver is able to verify the request integrity and to authenticate the sending domain[5]. However, we foresee several limitations with this protocol:

---

[4] Mainly the difficulty to authenticate sources, the possibility to address interconnection points from the Internet and the existence of compromised endpoints botnets.

[5] Beforehand, the receiving domain has to fetch the sending domain public key.

- The sender's public key fetching and checking, as well as the signature verification are resource consuming steps for the receiver that may expose him to DoS risk[6].
- The signature validity authenticates the sending domain but does not guarantee that the caller's phone number (or identity) has not been spoofed.
- Since the INVITE request has no confidentiality protection, it can not convey a session key to encrypt the media flows.

To solve this last limitation, IETF proposals ZRTP or DTLS-SRTP which perform a key handshake in the media plane (after the call signalling is established) may be used but they are challenging as regards to legal requirements[7] and moreover they require integrity protection at the signalling level for end-to-end security, as explained in [24]. When restricting the study scope to VoIP signalling security, TLS (or IPSec) may be used for hop-by-hop security and media key encryption. However, this approach leaves the E.164 phone number problem unresolved and requires the establishment of secure link prior to sending the INVITE request. As a result, specific security protocols like IETF MIKEY have been standardized for the establishment of secure multimedia sessions (without relying on lower layers security). The MIKEY protocol enables the key exchange to be integrated in the VoIP signalling and it supports several authentication modes. A more detailed analysis of MIKEY, along with other multimedia key exchange protocols (SDES, ZRTP, DTLS-SRTP), can be found in [24]; this study shows the DoS risks on several of these protocols.

2.2.2 Closed-Interconnection Model

In order to limit the risks identified in the open model, the first VoIP domains interconnections were performed through the PSTN, i.e. using contract between operators, secured links and more generally architectures isolated from the Internet.

In parallel, the 3GPP consortium has specified architectures to offer the same level of security and QoS as in PSTN but with IP protocols. These architectures are thus based, as in PSTN, on trusted links and secured interconnections between at least two domains, the model being designed to support any number of peers. The relevant standards are the IMS (IP Multimedia Subsystem); they specify a set of VoIP functions which reuse as much as possible the existing VoIP and security protocols (SIP, SDP, RTP, IPSec, and Diameter). The IMS defines the VoIP access for both mobile and fixed endpoints, as well as the roaming functions.

Within the IMS, security issues listed above are solved: the confidentiality and integrity of media and signalisation flows are guaranteed with IPSec tunnels. The user authentication is done in each domain and E.164 information is shared between domains to enable secure call routing and caller identification. Finally, protection against DoS attacks relies on topology hiding functions that make the interconnection points addressable only from the partner domains. In return, the IMS security model presents an inherent complexity which may have performance impacts, according to [26], especially in the core network.

[6] Some performance enhancements were proposed in [23] based on elliptic curves algorithms.
[7] This is explained in section 7.5.3 of [25] dealing with media plane security.

3 Analysis of Token or Ticket Based Security Mechanisms

In the remaining of this article we only consider the security protocols applying at the VoIP signalling level and in this section we focus on a specific class of security mechanisms based on tokens or tickets, which may be generated with a cryptographic function. These mechanisms address some of the issues identified previously, especially the "introduction problem" explained in [18], the routing/verification of E.164 phone numbers and also the "pinhole problem" explained in [16] which is the accumulation of DoS and SPIT risks over the interconnection points. They may apply to several of the architectures seen previously, especially the open-interconnection model which has the highest risks.

3.1 Out-of-Band Token Exchange

This mechanism [27] addresses the SPIT threat and also the introduction problem. It recognizes that black lists have limited effectiveness and that white lists are useful only if the first call from a new person is not systematically blocked. It assumes people usually have at least one previous contact before placing a VoIP call, such as e-mail or electronic business cards exchange. During this "cross-media relation", users exchange information which is then inserted as an authentication token in the VoIP call establishment request.

A first type of information is a VoIP identifier (e.g. SIP_URI) that the caller offers to potential callees. The callee stores this token and later uses it to label and filter any incoming call from this caller. Alternatively, a second type of information is weakly-secret value provided by a callee to potential callers who then insert it also in the SIP INVITE request. In both cases this can be considered as a "consent-based" approach.

The main advantage of this mechanism is the simplicity of token checking for the callee which limits the DoS risks and is efficient against blind calls. On the other hand, the callee has to store or distribute many tokens in a secure way to prevent spoofing. Since tokens have mid to long-term validity, any token theft will have disturbing consequences for the callee. Finally, obtaining and inserting tokens with non-VoIP endpoints may be a constraint for a lot of users.

3.2 Return Routability Test (RRC)

This mechanism [28], in association with SIP Identity [22], aimed to solve the risk of caller identifier spoofing. Its objective is to determine if a domain rightfully "owns" an E.164 phone number which appears in association with this domain name in the SIP caller identifier field of an INVITE request. Although the corresponding IETF draft has expired, this mechanism is mentioned here as an historical approach.

The basic idea is that when the callee receives a SIP INVITE request with an E.164 phone number in the caller identifier field, it sends an out-of-dialog verification request towards that E.164. The verification request contains the claimed E.164 caller identifier and a unique random token (nonce). Upon receipt of this request, the caller (or sending proxy), if it is legitimate for this E.164 number, sends back an acknowledgement containing the received token and the domain signature according to [22].

The callee then has to check that the signature in the acknowledgement response and the one in the SIP INVITE request are from the same domain.

The tricky point here is that the correct routing of the verification request to the E.164 calling number relies on SIP routing and thus leads to the already explained phone number routing problem. From this point, it appears that a trusted third party is necessary at least for secure routing and verification of phone numbers identifiers and this principle is retained in the following mechanisms.

3.3 Secure Call Establishment with KMS Like Trusted Third Party

This mechanism was initially proposed within 3GPP [25] and then in IETF [29] as an extension to the MIKEY protocol; it is designed for integration in VoIP signalling. It follows Kerberos protocol basis and assumes each VoIP entity (A and B) has a pre-shared secret with KMS (Key Management Server) which is the trusted third party.

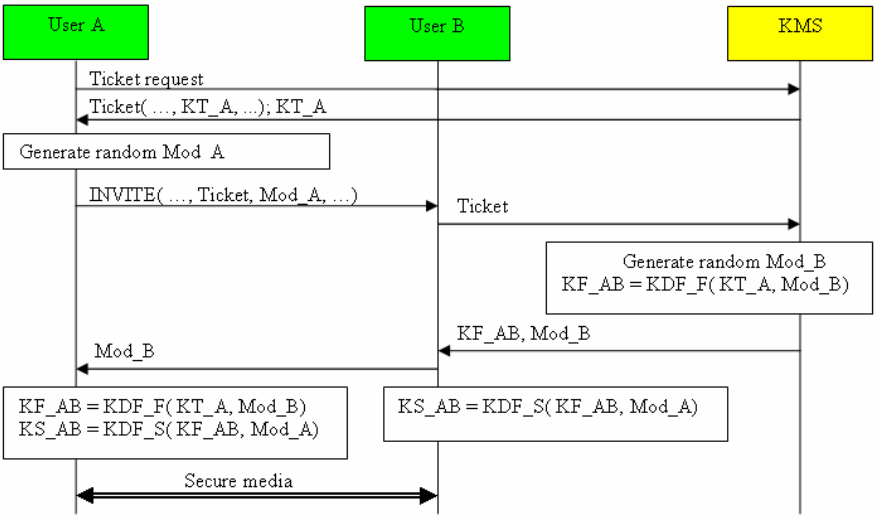


Fig. 1. Session signalling (from 3GPP TR33.828 source)

As shown in Figure 1, entity A authenticates to KMS which returns a ticket containing the master key KT_A encrypted for entity B along with the same key encrypted for entity A. Entity A includes this ticket with a random value Mod_A in the call establishment request sent to B. Then B forwards the ticket to KMS for checking. If the ticket is valid, KMS chooses a random value Mod_B and derives the session key KF_AB from KT_A and Mod_B. Then Mod_B and KF_AB (encrypted) are sent to B which forwards Mod_B to A. Entity A then derives KF_AB and both entities simultaneously compute the session key KS_AB.

In this approach, endpoint authentication and location as well as call routing are under KMS responsibility which guarantees the security of the whole exchange. However, in-band keys exchange creates large tickets and has a negative impact on transport. Also, we anticipate that entity B and KMS are both exposed to DoS attacks

because entity B has no means to check if the request from entity A is valid and will indiscriminately forward invalid tickets to KMS while creating a new context on its side.

3.4 Hybrid Model

The very recent proposal *VIPR* [16] merges VoIP, P2P, PSTN technologies and benefits from the PSTN historical reliability to establish secure routing and authentication information.

In brief, each node of the P2P network is a call agent (CA) for a given VoIP domain. Each CA publishes in the P2P DHT all the user identifiers held by the domain. Once a PSTN call is completed, the calling CA contacts the called CA in the receiving domain[8] and uses the previous PSTN call information as pre-shared secret. In return, the called CA creates (and sends to the calling CA) a cryptographic token bound to the callee and the calling domain. This token also contains VoIP routing information (SIP URI) to reach this destination and is then inserted into future VoIP calls from this calling domain to this specific callee.

The main advantage of this model is to achieve secure phone numbers routing and verification, thus avoiding caller identity spoofing and associated SPIT risk. Also, the mechanism is gradual since the more PSTN calls are established, the more tokens are granted and the more VoIP calls can be further initiated. In return, we anticipate several issues. First, each domain must publish in the DHT all its user identifiers; this raises rather the same "philosophical" problems as ENUM adoption and also some scalability issues with large domains. Secondly, each domain needs to store securely a large amount of (call) tokens which is roughly proportional to its number of users. Moreover, since token validity is assumed to be endless, this requires robust cryptographic functions and very strong keys protection. Finally, this model relies on PSTN routing each time a new destination is called, that is to say *endlessly*.

4 Proposed Enhancements

4.1 Framework Objectives and Positioning with Regards to State of the Art

The SIDE (Secure Inter-Domains Exchange) framework described in this section is a secure VoIP call establishment mechanism, operating at the signalling level, and aimed at open inter-domain interconnections. Its main objective is to reduce DoS and SPIT risks on the receiver side, i.e. to optimize the steps related to authentication and establishment of a master key between far end entities. Complementary objectives are: support for sporadic communications, with no need to set up a secure connection before placing the SIP INVITE request, support for endpoint roaming through visited domains and compatibility with current VoIP architectures (especially UDP transport). It also addresses the usual security requirements of integrity and confidentiality protection. Because of the paper size constraint we only provide an overview of this mechanism without proving its security properties.

[8] The contact address of the called CA is provided by the DHT based on callee identifier.

The SIDE framework borrows some principles from the mechanisms presented in section 3 and aims at improving them. From [27], it keeps the idea of identification token inserted in the SIP INVITE but removes the need of "cross-media relation" to obtain the token; also the token validity is limited to only one call for security reasons. From [28], it keeps the principle of a nonce token returned by the callee domain upon reception of the "call establishment request". This nonce token is indeed necessary for the caller to compute the session key and to further create a valid SIP INVITE request. From *KMS* approach [29], it keeps the principles of session establishment based on a trusted third party and the use of symmetric cryptography (as opposed to [22]) for performance purpose. However, the protocol exchange is organized in a different way as [29] to reduce the DoS risks on the receiving domain. Also the session key is not conveyed in the signalling exchange thus reducing the messages size and the cryptanalysis risks. From *VIPR* approach[16], it keeps the principle of (at least) one trusted third party per VoIP domain which is responsible for endpoint authentication, call routing and verification. The need for a trusted third party appeared all along the paper both for security reasons and for call routing, but it is also necessary for regulatory constraints [15] such as legal call interception. As opposed to [16], the proposed framework does not require the storage of a huge amount of cryptographic tokens.

4.2 Design Principles

The following principles were retained for design:

- *Inter-domain initial exchange*: this phase is performed only once between each pair of domains and enables them to exchange their lists of SIDE Primary Server (SPS), a Shared Master Key (SMK) and a couple of other parameters such as the maximum number of simultaneous transactions. Each domain shall declare at least one SPS which is considered as a trusted third party and is in charge of transactions authorization. SIDE Intermediary Servers (SIS) may also be involved in transactions but do not need to be declared thus making the management easier. The initial phase is under the control of the receiving domain which does not need to check *a posteriori* for the sending domain policies as in [22]. In the same way as existing protocols (TLS, IKE, MIKEY), the SMK is based on Pre-Shared Key (PSK) session establishment.
- *Call notification*: before triggering the SIP-INVITE request, a notification request is first sent to the receiving side to query consent. Since this request is much lighter than SIP-INVITE it saves processing for the receiving domain, especially if the call is declined. Furthermore, information is exchanged in clear during the notification phase, in the form of a Nonce-Token (NT), to compute the Transaction Master Key (TMK). Eventually, pre-routing may be achieved during this phase to direct the SIP INVITE request to the right entity in the receiving domain, thus saving processing in intermediary entities.
- *Insertion of an Identification-Token* (IT) inside the SIP-INVITE request which authenticates a given transaction and can be checked straightaway by the receiver[9], thus limiting DoS risks.
- *Use of symmetric cryptographic* algorithms for performance considerations.

[9] Checking of the received IT value is done by comparison to the pre-computed expected value.

4.3 Overview of the Protocol Exchange

Figure 2 shows the sequencing of a SIDE transaction, that is the protocol exchange leading to the acceptation of the SIP-INVITE request with the Identification-Token by the receiving entity[10]. We assume the inter-domain initial exchange has been previously completed and led to the establishment of the SMK secret value between the two domains.

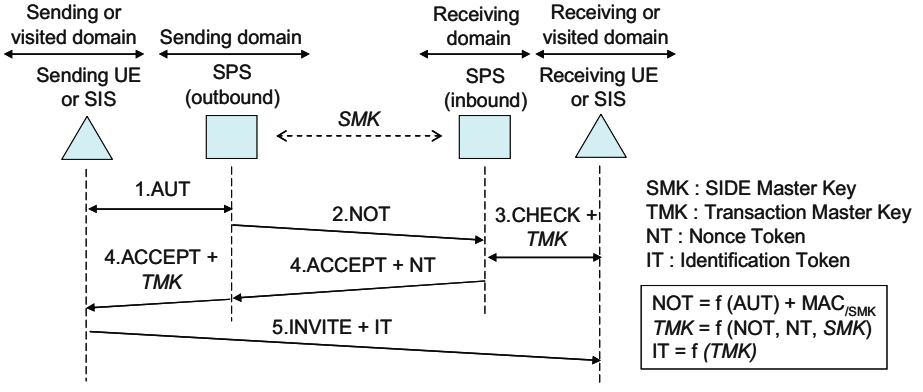


Fig. 2. SIDE protocol exchange overview

Step 1: The user endpoint (UE) or SIS in the sending domain detects an inter-domain call request and sends a transaction authorization request (AUT) towards the SPS. This request is based on the underlying SIP INVITE information and contains the call main characteristics, especially caller and callee identifiers, contact and media addresses. Optionally, the SPS requires and triggers user endpoint authentication.

Step 2: If SPS accepts the AUT request, it sends a notification request (NOT) towards one of the SPS in the receiving domain it previously learnt from the *Inter-domain initial exchange*. The NOT request combines AUT information with sending SPS information and a MAC (Message Authentication Code) based on the SMK key.

Step 3: If the NOT request is valid (MAC check) and the receiving SPS accepts the transaction it then computes the TMK value which is based on the information contained in the NOT request, the SMK key and a random value NT (Nonce-Token) created for this transaction. Prior to accepting the transaction, the receiving SPS may check white or black lists, query UE consent, or apply any other relevant filtering. If the transaction is accepted, the callee is sent the TMK value along with the transaction characteristics.

Step 4: The receiving SPS notifies its consent to the sending SPS; the answer includes the NT value and a MAC based on SMK key for integrity protection. Since the NT value is short and does not need to be encrypted, there is a significant advantage compared to [29] where the ticket is much longer and the keys it conveys have to be

[10] And the subsequent sharing of the TMK secret between the initiator and the responder.

encrypted. Upon receipt of the NT value, the sending SPS computes the TMK key and forwards it to the sending entity.

Step 5: The sending entity derives the IT (Identification-Token) value from the received TMK value and inserts it into the SIP-INVITE request it sends towards the receiving entity. The IT value in itself is not sufficient to ensure the INVITE request integrity, thus a MAC code based on a key derived from TMK should be added. The receiving entity checks the IT value of the SIP-INVITE request from a pre-calculated list and retrieves the corresponding TMK key it now shares with the caller; it also verifies the MAC value. The IT check is trivial for the receiving entity in contrast to [29] where the KMS has to be contacted to verify the ticket and extract the corresponding key.

5 Conclusions

In the first section we identified the current stakes in VoIP security showing that the risks are limited in intra-domain context but become higher in the inter-domain context due to possible DoS or SPIT on the interconnection points and the difficulty to authenticate network sources and validate caller identifiers. Whereas the closed-interconnection model removes most of these risks it may have difficulties in supporting more sporadic any-to-any services. The open-interconnection model requires efficient security mechanisms which have been analysed in sections 2 and 3. Finally the paper presented a very general overview of the SIDE framework which is inspired from the mechanisms presented in section 3 while bringing some enhancements. Future work around this framework includes: prototyping, performance analysis, formal verification of security properties and finally design of an alternate mode not requiring notification for each call.

Acknowledgments. The authors would like to thank Henry Gilbert, Joaquin Garcia-Alfaro, Nora Cuppens and Frédéric Cuppens for their helpful comments as well as Sarah Cook for her help with the English wording.

References

1. Blake, E.A.: Network Security: VoIP Security on Data Network-A Guide. In: Information Security Curriculum Development Conference (2007)
2. Feijoo, C., Gomez-Barroso, J.L., Rojo-Alonso, D.: A European Perspective of VoIP in Market Competition. Communications of the ACM (November 2008)
3. Abdelnur, H., et al.: Assessing the security of VoIP Services. In: The 10th IFIP/IEEE Symposium on Integrated Management (2007)
4. Griffin, S., Rackley, C.: Vishing. In: InfoSecCD'08: Proceedings of the 5th annual conference on Information security curriculum development (2008)
5. Endler, D., Collier, M.: Hacking VoIP Exposed. McGraw-Hill Osborne Media, New York (2006)
6. VoIPSA: VoIP security and privacy threat taxonomy. Public Realease 1.0 (October 2005)
7. Jennings, C., et al.: A SIP Usage for RELOAD. IETF Draft draft-ietf-p2psip-sip-03 (October 2009)

8. Coarfa, C., Druschel, P.: Performance Analysis of TLS Web Servers. ACM Transactions on Computer Systems, TOCS (2006)
9. Sit, E., Morris, R.: Security considerations for peer-to-peer distributed hash tables. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, p. 261. Springer, Heidelberg (2002)
10. Yan, J., Ahmad, A.: A Low-Cost Attack on a Microsoft CAPTCHA. In: Proceedings of the 15th ACM conference on Computer and communications security (2008)
11. Athanasopoulos, E., Antonatos, S.: Enhanced CAPTCHAs: Using Animation to Tell Humans and Computers Apart. Communications and Multimedia Security (2006)
12. Yan, J., Ahmad, A.: Usability of CAPTCHAs Or usability issues in CAPTCHA design. In: Symposium On Usable Privacy and Security (SOUPS) (July 2008)
13. Mathieu, B., et al.: SPIT mitigation by a network level anti SPIT entity. In: VSW'06: Third annual security workshop (2006)
14. Cerf, V.G.: Spam, Spit and Spim. Communications of the ACM (April 2005)
15. Hill, J.: The Storm Ahead: How CALEA will turn VoIP on its head. In: InfoSecCD '06: 3rd annual conference on Information security curriculum development (2006)
16. Rosenberg, J., Jennings, C.: Verification Involving PSTN Reachability: The ViPR Access Protocol (VAP). Draft IETF draft-rosenberg-dispatch-vipr-vap-00 (November 2009)
17. Niccolini, S., et al.: SPEERMINT Security Threats and Suggested Countermeasures. Draft IETF draft-ietf-speermint-voipthreats-01 (July 2009)
18. Rosenberg, J., Jennings, C.: The Session Initiation Protocol (SIP) and Spam. IETF RFC5039
19. Pathak, A., et al.: Botnet Spam Campaigns Can Be Long Lasting: Evidence, Implications, and Analysis. In: SIGMETRICS '09: Measurement and modeling of computer systems (2009)
20. Nassar, M., et al.: Holistic VoIP intrusion detection and prevention system. In: IPTComm '07: Principles, systems and applications of IP telecommunications (2007)
21. Fiedler, J., et al.: VoIP defender: highly scalable SIP-based security architecture. In: IPTComm '07: Principles, systems and applications of IP telecommunications (2007)
22. Peterson, J., Jennings, C.: Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP). IETF RFC4474 (August 2006)
23. Rebahi, Y., et al.: Performance analysis of identity management in the Session Initiation Protocol (SIP). In: ACS International Conference on Computer Systems and Applications, AICCSA (2008)
24. Floroiu, J., Sisalem, D.: A comparative analysis of the security aspects of the multimedia key exchange protocols. In: Principles, systems and applications of IP telecommunications, IPTCom (2009)
25. IMS Media Plane Security. 3GPP TR33.828-161 (December 2009)
26. Tonesi, D.S., Salgarelli, L., Tortelli, A.: Securing the signaling plane in beyond 3G networks: analysis of performance overheads. Security and Communication Networks (2009)
27. Ono, K., Schulzrinne, H.: Have I Met You Before? Using Cross-Media Relations to Reduce SPIT. In: Principles, systems and applications of IP telecommunications, IPTCom (2009)
28. Wing, D.: SIP E.164 Return Routability Check (RRC). IETF draft-wing-sip-e164-rrc-01 (February 2008)
29. Mattsson, J., Tian, T.: MIKEY-TICKET: An Additional Mode of Key Distribution in MIKEY. IETF draft-mattsson-mikey-ticket-00 (October 2009)

Entropy of Graphical Passwords: Towards an Information-Theoretic Analysis of Face-Recognition Based Authentication

Stefan Rass[1], David Schuller[2], and Christian Kollmitzer[2]

[1] Universitaet Klagenfurt, Institute of Applied Informatics, System Security Group, Universitaetsstrasse 65-67, 9020 Klagenfurt, Austria
stefan.rass@uni-klu.ac.at

[2] AIT Austrian Institute of Technology GmbH, Quantum Technologies, Department Safety & Security, Lakeside B01A, 9020 Klagenfurt, Austria
{david.schuller.fl,christian.kollmitzer}@ait.ac.at

Abstract. We present an information-theoretic discussion of authentication via graphical passwords, and devise a model for entropy estimation. Our results make face-recognition based authentication comparable to standard password authentication in terms of uncertainty (Shannon-entropy) that an adversary is confronted with in both situations. It is widely known that cognitive abilities strongly determine the choice of alphanumeric passwords as well as graphical passwords, and we discuss various selected psychological aspects that influence the selection process. As a central result, we obtain a theoretical limit to the entropy of a face-recognition based authentication in the light of some social engineering techniques (dictionary attacks on graphical passwords). Remarkably, our results hold independently of any information that can be obtained from the internet or through other forms of social engineering. Thus, we obtain very general bounds on the quality of authentication through face-recognition that solely depend on the authentication mechanism.

Keywords: Graphical Passwords, Face-Recognition, Authentication, Shannon-Entropy.

1 Introduction

Graphical passwords are an elegant way of overcoming a certain vulnerability of standard password authentication. A naive user may write down a password somewhere, or perhaps tell it on the phone, if an adversary manages to trick the user into believing that the call is from some honest service center. Composing an access code from images rather than symbols prevents writing it down, and also hampers giving the secret away otherwise. Using faces as images greatly supports memorability of the secret access code, and has therefore become a popular approach. Consequently, we consider face-recognition based authentication in the following, bearing in mind that the ideas presented here can easily be extended to various other types of graphical passwords (cf. the related work section).

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 166–177, 2010.

A major contribution of this work is an information-theoretic measure of quality that a face-recognition challenge presents to the adversary. Since the quality of alphanumeric password policies can be measured in terms of entropy (using combinatorial considerations), an analogous measure for graphical passwords is certainly desirable. However, the literature about graphical passwords hardly provides any assertions about entropy of graphical passwords. It turns out that this goes much beyond the usual combinatorics that arises for standard symbolic passwords. In the case of face-recognition based authentication, such an entropy measure is, however, obtainable with combinatorial tools. The main task accomplished in this work is estimating the maximum possible entropy of an authentication challenge based on face selection. We demonstrate how to do this in section 3, along with a discussion of possible variations to the model. A further important point discussed in this work are psychological aspects regarding the memorability and selection process of images, particularly faces. Choosing weak passwords due to mental limitations of the human brain is a well-known problem. Similar concerns apply for some graphical password authentication systems, so choosing "weak face-sequences" is an equally likely incident. The second contribution of the paper is a discussion of such aspects, found in section 2.1. The paper closes with a small example illustrating the derived results, and discussing various directions of future research.

Related Work: The author of [1] gives various alternatives for alphanumeric passwords, along with discussions of security. The work of [2] contains vulnerabilities in face-recognition based authentication systems. An introduction to various kinds of graphical passwords is found in [3,4] as well as [5]. The latter discusses the information content of (graphical) passwords in general, but does not provide specific upper bounds for the scenario we consider here. The authors of [6] present an implementation and empirical evaluation of a graphical password authentication system. Though entropy is discussed there briefly, a thorough formal analysis is missing. An interesting idea is given in [7], where graphical passwords are made resistant against spying over the shoulder of the user (shoulder-surfing). In [8], useful hints towards building a secure graphical password authentication system are found. We follow a similar path in this paper. Empirical studies regarding the performance of graphical passwords are given in [9]. Passfaces™ [10], Déjà Vu [11] or Awase-E [12] implement authentication algorithms similar to the ones we consider here. We describe this mechanism in more detail in section 2. Other approaches to graphical passwords involve finding and clicking certain pass-positions within an image or drawing passwords on a grid. See [13] for further references.

2 Authentication through Face-Recognition

Face-recognition is one cognitive action that human brains are well accustomed to. Consequently, one would recognize faces easily, but describing a face to another person such that this face could too be recognized by the other person is rather difficult. So why not use a sequence of faces (or general images) instead

of a sequence of symbols for forming a password? This is the basic idea behind using faces for authentication: instead of being prompted to type in a password, the user is prompted to select a couple of faces from a pool of given pictures, with several decoy pictures among them (see [10,11,12]. In the sequel, we shall assume that the order in which these are chosen during the authentication is of no relevance (in section 3.3, we discuss how to account for this too). At the time when the "password" is chosen, the user is free to specify some images that will be recognized. This is completely analogous to choosing a password that can be remembered. For logging into the system, the user will pick the right ones among the decoy pictures to complete the authentication. The obvious advantage is that neither the faces can be written down, nor can easily be described to another person to permit that one to authenticate herself. In that sense, this approach appears superior to standard passwords, but a direct comparison is not trivial. For alphanumeric passwords, combinatorial considerations quickly lead to estimates how many passwords can be chosen according to the given policy. This can be taken as a measure of quality of the password-selection process. Can similar things be done for face-recognition based authentication? An answer is provided in section 3.1.

2.1 Psychological Aspects

We know that as early as the second and third days of life, babies are able to distinguish between happy and sad faces, by their second or third month they develop an affective consonance with their mother, to the extend that they reproduce more or less synchronized facial expressions [14]. It is true that these are most rudimentary forms of empathy, much less sophisticated than those underlying our social conduct when we reach maturity, but both require the capacity to understand the emotions of others, to read signs of pain, fear, disgust, and joy in their faces [15].

Empathy is the intimate and fundamental potential of socialization, however, there has been much controversy about the two main but contrastive theories over the last three decades which try to explain the ability to share emotions. The "theory of mind" argues that the development of other views and feelings can only be conceived with a previous knowledge or other outside appearances. It is defined by a cognitive determination individually given by experience. In the early 1990s, Giacomo Rizzolatti and his team discovered the mirror neurons and its location in the premotor cortex and partly in the cerebral cortex, which is responsible for the coordination of motion and also in the broca-areal which is responsible for the development of language. There are obvious reasons to define a connex between the mirror neurons an its location in brain areas that are indispensable for the ability to interact socially.

A critical review took place of how one is strongly contagioned by expressed emotions of others. These neurons seem to have the surprising property of responding not only when a subject performed a given action, but also when the subject observed someone else performing that same action. Gallese, one of Rizzolatti's group members, speculated, that the mirror neurons enable a person to

share intentions, views and goals. However, studies analyzing the correlation of language and the thinking of the state of mind of others question if the simulation mechanism fulfills also the ascription of emotions notionally. "This would mean that people with a damaged amygdala do not recognize fear in the faces of his fellow-men, nevertheless they often manage to see that fear anyhow," says Rebecca Saxe, a assistant professor at the Massachusetts Institute of Technology [15]. Other colleges see it differently. Claus Lamm from the University of Zurich [15]: "When we observe emotions it is primary about the symphatisation and not about the attribution of mindsets. Simulation theories employ emotions and actions, directly observable conditions, for example laughing or tears. "The theory of mind" generally examines the understanding of not directly observable conditions." Kai Vogeley from the University of Cologne says both sides are right. He differs between conscious thinking about others and prereflexive empathy, where we instantaneously understand others. In other words, the mirror neurons probably help us so we can intuitively emphasize with somebody, before we precisely form an imagination of the other person.

In times of growing media technologies an the insertion of attractive methods to capture clients, the image becomes more and more, what appears to be rated as a powerful expression to transfer emotions and action. In relation to our analysis of how people choose faces when they want to maximize security in internet operations we see how the levels of consciously clarifying expressions of others and intuitive immediate reactions to them mix together in various individual possibilities that appear to develop incomprehensibly. Though, if we look at social networking websites such as Facebook where users explicitly inform other users with illustrative material from their own livelihood and social backgrounds, we assume similarities between faceimages that are displayed (friends, admired movie stars, etc.) and faces that are picked for passcode identities, as we are constituted to refer to our images that represent our livelihood. These images are activators of emotive contexts that rely strongly on self-cumulative needs that want to be transferred in a interdependent form.

The exchange of selected pictures has become a prestigious way of socializing and the exclusion of images that could reveal own doubtful stories are commonly accepted. Therefore, pictures not reflecting on a specific individual coded cognition, are not chosen for safety-purposes. This fact may attract hacker, who inform themselves about such individual characteristic traits by looking at the images shown on personal websites. They can be looked upon as fairly reliable evaluations in order to raise the possibility to find security weaknesses. It is not yet clear if face-recognition systems do not have similar "weak spots" compared to passwords regarding the divulgement of the social surroundings behind published pictures of someone. Standard password dictionary used to go into other systems illegally are as imaginable for graphical passwords. With forty three muscles we can create more than ten thousand face expressions but only a few of them are seen regularly, foremost expressions that show our basic emotions like fear or happiness. Unconsciously created Microexpressions may be a way of

diminishing the vulnerability when choosing face images for authentication, as it is even more difficult to describe faces that seemingly express a "hidden look".

2.2 A Simple Dictionary Attack

Assume that the adversary has collected a pool of pictures that can be related to the subject whose graphical password should be identified. Such a pool can be constructed from social networking sites, such as Facebook (www.facebook.com), Twitter (www.twitter.com), MySpace (www.myspace.com) and many more; perhaps also photosharing websites like Flickr (www.flickr.com), for instance. A general purpose dictionary of pictures of people that are familiar to many persons may for instance be constructed using celebrity pictures that can be found via the image search feature of most of the popular search engines. A standard remedy against dictionary attacks are retry counters locking the login mask after a small numbers of failures.

It has been demonstrated that such sites can be exploited for automated social engineering [16], so using the same technology for collecting photos appears as an easy next step. A simple attack strategy is matching pictures from the login challenge screen with those in the dictionary. If strong similarities can be identified, then those pictures can be chosen for a trial login. If less pictures can be recognized than are in the dictionary, then the remaining ones have to be chosen on another basis; perhaps without any help in the worst case. Doing the matching is a challenge by itself, but can be done.

Many digital cameras sold nowadays are able to recognize faces quickly in order to properly set the focus of the camera. Such algorithms (see [17,18]) can equally well be used for extracting faces from pictures of groups of people, or from pictures where the person is not the major content of the photograph. Several algorithms for matching pictures against each other are available in the computer vision literature (see [19]). Calculating similarity between images is a highly nontrivial task, as rotation (of the picture, as well as the person shown in the picture), can introduce severe difficulties. Nevertheless, similarity estimates *can* be obtained automatically, as has been demonstrated in the cited literature. We are basically concerned with the information-theoretic quality of passwords, leaving out the details of image processing here.

3 Entropy of Face-Recognition Challenges

In his seminal paper [20], C. Shannon introduced entropy as a measure of uncertainty of choices. In our case, this will be the choice of (password-)pictures from a given pool. Similarly as for alphanumeric passwords, where entropy measures the uncertainty of choice from the set of possibilities, graphical password selection too enjoys various degrees of freedom. Our model will upper bound the Shannon-entropy of the probability distribution modeling the selection of a graphical password, thus yielding a natural measure of quality that is interpretable and comparable to standard measures. Before giving the model in section 3.1, we briefly introduce the entropy concept for convenience of the reader.

Shannon constructed the entropy H of a given probability distribution in order to satisfy three conditions [20]: First, H is nonnegative and continuous. Second, H strictly increases with the number of choices, if each of them is equiprobable. Third, if a choice can be decomposed into several sub-choices, then the overall entropy is the sum of the first choice's entropy, plus the weighted sum of each subsequent choice's entropies, where the weight is the probability of facing that choice. We will extensively use that property in section 3.1.

Shannon proved [20, Theorem 2] that the only function satisfying all three requirements is of the form $H(p_1, \ldots, p_n) = -K \sum_{i=1}^{n} p_i \log(p_i)$, with the convention that $0 \log 0 = 0$ and K being any positive constant. A common choice for K is such that the logarithm is to the base 2, giving the entropy in bits. We will implicitly assume this throughout the remainder of this work.

It is well known that entropy is maximized for the uniform distribution, in which case we have $H(1/n, \ldots, 1/n) = \log n$, and minimal for any degenerate distribution (i.e. point mass) making H vanish. To measure the quality of a password, one identifies the set from which those passwords are chosen, and calculates the entropy of the distribution from which the passwords are drawn. Doing some combinatorics to determine the number of possible passwords, the maximal uncertainty occurs of each of these is chosen with equal probability. However, this makes remembering the password an almost infeasible task. Using mnenomics or other tricks to easily remember or derive passwords changes the shape of the selection distribution into something different from the uniform distribution and thus lowers the entropy. If the same password is chosen by everyone, then there would be no uncertainty and hence zero entropy. Though counting the number of passwords and calculating the maximum possible entropy is often easy, determining the empirical entropy of passwords is a highly nontrivial task (see [21] for some figures). In this work, we shall take the first step for graphical passwords, leaving empirical studies subject of future research and follow-up papers. For existing field trials, see [9].

3.1 Upper-Bounding the Entropy

We seek a measure of uncertainty that resembles the way in which the quality of passwords is measured by Shannon entropy. Passwords are most easily remembered when they are words or somehow constructed from mnemonics. Following that idea, assume that memorability of faces is supported by drawing them from the pool of familiar faces in one's mind. Assuming further that many of those are available on social networking sites (Facebook, Twitter, etc.) or public photo-sharing (flickr.com, etc.). Let us introduce a probability for having chosen one face that the adversary could locate on the internet. This probability, along with the probability of actually having photographs on the internet, helps setting up a decision tree (cf. figure 1). We will upper-bound the entropy of the resulting probability distribution in the following, yielding a theoretical limit of the entropy of graphical passwords, similar as this can be done for passwords. Throughout the remainder of this work, except where stated otherwise, we assume that the authentication system is insensitive to the order in which

the faces (or images in general) are chosen by the user. This requirement can be relaxed, as we discuss in section 3.3. The variables for our model are as follows: n denotes the number of pictures that are found on the web and can be related to the subject of interest. p is the probability of the subject under attack having no personalized information (pictures) available on the internet. m is the number of pictures presented at the login-screen, and k is the number of pictures to be chosen for login.

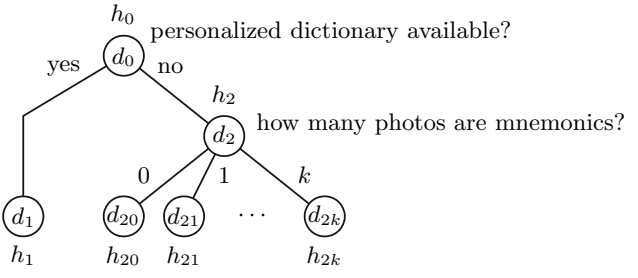


Fig. 1. Decision Tree

We give a step-by-step construction of the model, illustrating the idea using the decision tree displayed in figure 1. The first decision to be made is whether or not the subject under attack has pictures on the internet that enjoy a relation or some personal value. The decision d_0 has entropy h_0. If no, then we follow the left path, down to the face-recognition authentication with no additional information. The adversary gets a screen showing m pictures, k of which should be chosen. This choice d_1 has entropy h_1. Otherwise, if the internet does provide some personal photo albums or other pictures that can be related to the subject under attack, then how many of these have been used when the graphical password was chosen? Looking at decision d_2 (having entropy h_2), the user may have taken $1, 2, \ldots, k$ faces that looked familiar or are otherwise easily remembered. If none of the photos served as a graphical mnemonic (0), then the information from the web was worthless, and the entropy for the adversary is the same as h_1, making decision d_{20} basically identical to decision d_1. In each other case, we count the number of choices left to the adversary, and call the entropies $h_{21}, \ldots, h_{2k}$. We estimate each of these in the following, recursively combining them into an overall measure of uncertainty about the graphical password, in the light of "Web 2.0" [22].

Lacking precise estimates for the probability p, as well as the probability distribution on the set of decisions $\{d_{20}, d_{21}, \ldots, d_{2k}\}$, we can only search for some upper bound to the entropy. Unfortunately, it will not suffice to choose the uniform distribution everywhere in order to maximize the overall entropy, as can be illustrated using the first decision d_0: fix h_1 and h_2 for the moment, then the worst case value for p is determined via the following optimization problem:

$$\begin{aligned} \text{maximize over } p: \quad & H(p, 1-p) + ph_1 + (1-p)h_2 \\ \text{subject to} \quad & p \in [0, 1] \end{aligned}$$

The optimization goal can be simplified to $H(p, 1-p) + p(h_1 - h_2)$, discarding the constant additive term h_2. As H is differentiable, the optimal p is found by solving the equation $\frac{dH(p)}{dp} = h_2 - h_1$ for p within the range $[0, 1]$, giving

$$p^* = \frac{\exp(h_1 - h_2)}{1 + \exp(h_1 - h_2)}, \tag{1}$$

as the worst-case value for p. Though p is dependent on h_1, h_2, neither of them is dependent on p, so we are free to maximize h_1, h_2 separately. We start with a look at d_2 and its entropy h_2. Denote by $\pi = (p_0, p_1, \ldots, p_k)$ the probabilities for the events that $0, 1, 2, \ldots, k$ pictures have been found in the dictionary obtained through the internet. The entropy is found as $H(p_0, p_1, \ldots, p_k) + \sum_{i=0}^{k} p_i h_{2i}$, where $H(p_0, p_1, \ldots, p_k) \leq \log(k+1)$ with equality if π is the uniform distribution, and $\sum_{i=0}^{k} p_i h_{2i} \leq \max_i h_{2i}$ is trivial because a convex combination of values can never exceed the maximum term. The latter bound can be further explicated: suppose that i pictures have been found in the dictionary, then the remaining $k-i$ pictures are to be chosen from the given $m-i$ pictures on the login screen. Disregarding the order, this gives $\binom{m-i}{k-i}$ possible choices, so that $h_{2i} = \log \binom{m-i}{k-i}$ at most (again, using the fact that the entropy is maximal for the uniform distribution [20]). The recursive definition of the binomial coefficient through $\binom{m}{k} = \binom{m-1}{k-1} + \binom{m-1}{k} \geq \binom{m-1}{k-1}$, upon induction, instantly yields $\binom{m-i}{k-i} \leq \binom{m}{k}$ for all $i \geq 0$, so that $\max_i h_{2i} = h_{20} = h_1 = \binom{m}{k}$. Combining the bounds gives

$$h_2 = H(\pi) + \sum_{i=0}^{k} p_i h_{2i} \leq \log(k+1) + \log \binom{m}{k}. \tag{2}$$

Regarding h_1, a much simpler consideration tells us that without any dictionary of pictures, we are left with a choice of k pictures from a given set of m pictures at the login. This gives $\binom{m}{k}$ choices, and the entropy h_1 satisfies $h_1 \leq \log \binom{m}{k}$. Plugging the last inequality as well as (2) into (1) gives (after simplifying) $p^* = \frac{1}{k+2}$, as the worst-case value for p. Combining the maximal values into an overall upper bound for the entropy, we find that

$$H(p^*, 1-p^*) + p^* h_1 + (1-p^*) h_2 = H\left(\frac{1}{k+2}, 1 - \frac{1}{k+2}\right) + \frac{1}{k+2} \log \binom{m}{k}$$

$$+ \frac{k+1}{k+2} \left[\log(k+1) + \log \binom{m}{k}\right] = \log \left[(k+2) \cdot \binom{m}{k}\right],$$

after some messy algebra. Concluding all this, let H_{pf} denote the Shannon-entropy of a face-recognition based authentication challenge, where k faces are chosen from a set of m given images. Then

$$0 \leq H_{\mathrm{pf}} \leq \log \left[(k+2) \cdot \binom{m}{k}\right], \tag{3}$$

where the lower bound is sharp, because once the secret image sequence is known to the adversary, the entropy vanishes. The accuracy of the approximation is

dependent on the true shape of the distribution π, as well as the probability p. Both can only be determined through field trials, and are in turn depending on what and how many social networking activities the subject participates in. Hence, giving representative figures on the degree of approximation is beyond the scope of this work.

The reader may wonder why the parameter n (the number of pictures that have been obtained from the internet) nowhere appears throughout the whole analysis. The parameter n would have affected the distribution π that we used during the investigation of h_2 above. The more pictures are available, the more likely it is that some picture may have reminded the user of a face used for authentication. Though the number n may enjoy strong influence on the distribution, its impact vanishes due to the maximization argument that we used. Concluding that larger dictionaries give better chances to succeed in the authentication appears correct, but interestingly, it has no influence on the overall maximum entropy of the authentication challenge. This makes inequality (3) particularly appealing, as its value is solely determined by the parameters of the authentication challenge, and does not hinge on any unknown quantities.

Notice that the bound (3) can only be attained if the pictures are chosen such that no personal photograph (obtained from social networking websites) would provide a clue to the adversary on what pictures are likely to be the right ones. Furthermore, the probability p of finding such pictures has been optimized to yield a worst case estimate, so the actual entropy may be lower.

3.2 Example and Comparison

The correct values of p and n are to be determined by empirical studies, being subject of current and future research. Fortunately, neither of them appears in the bound (3). Figure 2 displays the upper bound of entropy for the parameter ranges $k = 5, \ldots, 8$ and $m = 16, \ldots, 64$. This corresponds to login masks displaying a 4×4 through 8×8-matrix of faces, of which 5 to 8 pictures have to be chosen. The numbers printed within the contour plot indicate the entropy measured in bit.

Notice that inequality (3) is only an estimate, and does not account for any psychological aspects that may have led to different choices. Taking those into account means deciding upon some faces being more likely chosen than others. Mathematically, this is expressed by deviating from the uniform distribution to a more bell-shaped distribution (possibly multimodal). We took this into account when we asked for the maximum entropy in our previous considerations. The result we obtained is independent of empirical estimates, and as such can be taken as a theoretical limit to the power of face-recognition based authentication in terms of uncertainty for an adversary.

Let us compare the graphical face-recognition based authentication to a standard password authentication: assume that a password is chosen from the set of letters (case-sensitive), as well as numbers and 5 special characters. The password is required to contain at least 6 symbols, one of which must be a number, and one must be a special character. Doing the combinatorics, we find that the

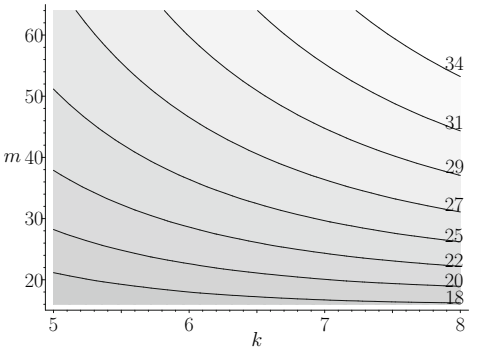


Fig. 2. Contour plot of entropy upper bound for various parameters k, m

number of passwords with 6 characters is $N = 30226681500$, giving $\lceil \log_2 N \rceil = 35$ bit entropy at best (notice that the true value will be smaller due to mnemonics used to choose and correctly recall the password later). On the contrary, a face-recognition authentication challenge asking for 8 pictures to be selected from a 8×8-matrix on the login screen has about 34 bit of entropy at most. It follows that, in this example, passwords are still superior to the graphical password approach, because the entropy is most easily increased by different password policies (simply set the length to 8 characters with the same constraints to get more than 60 bit of entropy), while the face-recognition authentication enjoy less degrees of freedom to do that. One way out of this dilemma, however, is to make the authentication order-sensitive (i.e. the order in which faces are selected by the user matters), so that additional uncertainty is introduced. We consider this in future research, and briefly discuss the alter model below.

A standard PIN challenge (found at most ATMs) would ask for, say 5, digits out of ten, giving $\log(10^5) \approx 17$ bit of entropy. Doing the same with a face-recognition challenge presenting 25 images, 5 of which shall be selected, the entropy comes to ≤ 19 bit. The true advantage, however, is the resilience against shoulder surfing, because memorizing a PIN by looking over one's shoulder is much easier than memorizing a sequence of faces.

3.3 Variations of the Model

Inequality (3) can as well be used with more than a single source when we consider a joint source of pictures being composed of the websites that a user most frequently visits. Such information can easily be obtained using Cookies, as is common practice.

Another interesting variation is achieved by letting the graphical password challenge account for the order in which the pictures are clicked by the user. This introduces additional uncertainty in the overall process, as even if some photos can be identified as similar to the given pictures, the order in which those are to be clicked remains unknown. The above calculations can straightforwardly be

carried out by replacing the binomial coefficient $\binom{x}{y}$ by $y!\binom{x}{y}$, which accounts for the order too. In addition, to each h_{2i} (for $i = 0, 1, 2, \ldots, k$) one needs to add the uncertainty about the permutation, which is $\log(k!)$, if the graphical password is composed of k pictures.

4 Conclusion

We presented an information-theoretic approach to measuring the quality of face-recognition based authentication challenges. As being developed to be an alternative to standard password authentication, graphical passwords call for a measure of quality that make them comparable to standard authentication mechanisms. As it turns out, upper bounds to the entropy can be derived without empirical knowledge. The derivation of the upper bound (3) indicates that this one might be crude, and could be improved upon empirical studies. These in turn may lead to more complicated, but tighter bounds on the true quality of a graphical password authentication. Psychological aspects, such as discussed in section 2.1 are an important ingredient for constructing a more accurate model. Nevertheless, the theoretical limit (3) that we obtained here already indicates some limitations compared to standard password authentication. Considering the numerical results obtained above, an authentication asking for a selection of 8 pictures from a pool of 64 only has an entropy of ≈ 34 bit at best. However, this bound is only valid under worst-case assumptions, including that no personal photograph obtainable for the adversary gives any clues for the authentication challenge. As far as our results indicate, a face-recognition based authentication (in a simple form) is not an attractive alternative to standard passwords authentication.

References

1. Brostoff, A.: Improving password system effectiveness. PhD thesis, University of London, Department of Computer Science (2004)
2. Duc, N.M., Minh, B.Q.: Your face is not your password - face authentication bypassing lenovo - asus - toshiba. Technical report, Security Vulnerability Research Team, Bach Khoa Internetwork Security (Bkis), Ha Noi University of Technology, Vietnam (2009)
3. Eljetlawi, A.M., Ithnin, N.: Graphical password: Comprehensive study of the usability features of the recognition base graphical password methods. In: Int. Conf. on Convergence Information Technology, vol. 2, pp. 1137–1143 (2008)
4. Suo, X., Zhu, Y., Owen, G.S.: Graphical passwords: a survey. In: 21st Annual Conf. on Computer Security Applications, p. 10 (2005)
5. Jermyn, I., Mayer, A., Monrose, F., Reiter, M.K., Rubin, A.D.: The design and analysis of graphical passwords. In: SSYM'99: Proc. of the 8th conf. on USENIX Security Symposium, Berkeley, CA, USA, p. 1. USENIX Association (1999)
6. Wiedenbeck, S., Waters, J., Birget, J.C., Brodskiy, A., Memon, N.: Authentication using graphical passwords: effects of tolerance and image choice. In: SOUPS '05: Proc. of the 2005 symposium on Usable privacy and security, pp. 1–12. ACM, New York (2005)

7. Li, Z., Sun, Q., Lian, Y., Giusto, D.: An association-based graphical password design resistant to shoulder-surfing attack. In: IEEE Int. Conf. on Multimedia and Expo., pp. 245–248 (2005)
8. Thorpe, J., van Oorschot, P.C.: Towards secure design choices for implementing graphical passwords. In: ACSAC '04: Proc. of the 20th Annual Computer Security Applications Conf., Washington, DC, USA, pp. 50–60. IEEE Computer Society, Los Alamitos (2004)
9. Brostoff, S., Sasse, M.A.: Are passfaces more usable than passwords? a field trial investigation. In: Proc. of HCI (2000)
10. Real User Corporation: The science behind passfaces (September 2001), http://www.realuser.com/published/ScienceBehindPassfaces.pdf (last access January 7, 2010).
11. Dhamija, R., Perrig, A.: Déjà vu: A user study using images for authentication. In: Proc. 9th USENIX Security Symposium, pp. 45–58 (2000)
12. Takada, T., Koike, H.: Awase-e: Image-based authentication for mobile phones using user's favorite images. In: Chittaro, L. (ed.) Mobile HCI 2003. LNCS, vol. 2795, pp. 347–351. Springer, Heidelberg (2003)
13. Li, S., Shum, H.Y.: Secure human-computer identification (interface) systems against peeping attacks: SecHCI. Cryptology ePrint Archive, Report 2005/268 (2005), http://eprint.iacr.org/
14. Rizzolatti, G., Sinigaglia, C.: Mirrors in the brain: How our minds share actions and emotions. Oxford University Press Inc., New York (2006)
15. Breuer, H.: Empathie – Streit um das soziale Hirn. Süddeutsche Zeitung, p.16 (2010) (issue from the 4th of January)
16. Huber, M., Kowalski, S., Nohlberg, M., Tjoa, S.: Towards automating social engineering using social networking sites. In: 2009 Int. Conf. on Computational Science and Engineering (CSE), August 2009, vol. 3, pp. 117–124. IEEE, Los Alamitos (2009)
17. Arnaud, E., Fauvet, B., Memin, E., Bouthemy, P.: A robust and automatic face tracker dedicated to broadcast videos. In: IEEE Int. Conf. on image processing, Genes Italy (2005)
18. An, K., Yoo, D., Chung, M.: An efficient fully automatic face tracking using binary template matching. In: Proc. of The Ninth Int. Symposium on Artificial Life and Robotics, Beppu, Japan, January 28-30, pp. 37–40 (2004)
19. Chen, C.H. (ed.): Handbook of pattern recognition and computer vision, 3rd edn. World Scientific, Singapore (2005)
20. Shannon, C.: A mathematical theory of communication. Bell System Technical Journal 27, 379–423 (1948)
21. Yan, J., Blackwell, A., Anderson, R., Grant, A.: The memorability and security of passwords - some empirical results. Technical Report 500, University of Cambridge, Computer Laboratory (September 2000)
22. Wikipedia Foundation: Web 2.0 (2004), http://en.wikipedia.org/wiki/Web_2.0 (last access January 5, 2010).

Handwriting Biometric Hash Attack: A Genetic Algorithm with User Interaction for Raw Data Reconstruction

Karl Kümmel[1], Claus Vielhauer[1,2], Tobias Scheidat[1,2],
Dirk Franke[2], and Jana Dittmann[2]

[1] Brandenburg University of Applied Sciences, Department of Informatics and Media
Magdeburger Str. 50, 14770 Brandenburg, Germany
[2] University of Magdeburg, Department of Computer Science, Advanced Multimedia and Security Lab
Universitätsplatz 2, 39106 Magdeburg, Germany
{kuemmel,vielhauer,scheidat}@fh-brandenburg.de,
dirk.franke@st.ovgu.de, jana.dittmann@iti.cs.uni-magdeburg.de

Abstract. Biometric Hash algorithms, also called BioHash, are mainly designed to ensure template protection to its biometric raw data. To assure reproducibility, BioHash algorithms provide a certain level of robustness against input variability to ensure high reproduction rates by compensating for intra-class variation of the biometric raw data. This concept can be a potential vulnerability. In this paper, we want to reflect such vulnerability of a specific Biometric Hash algorithm for handwriting, which was introduced in [1], consider and discuss possible attempts to exploit these flaws. We introduce a new reconstruction approach, which exploits this vulnerability; to generate artificial raw data out of a reference BioHash. Motivated by work from Cappelli et al. for fingerprint modality in [6] further studied in [3], where such an artificially generated raw data has the property of producing false positive recognitions, although they may not necessarily be visually similar. Our new approach for handwriting is based on genetic algorithms combined with user interaction in using a design vulnerability of the BioHash with an attack corresponding to cipher-text-only attack with side information as system parameters from BioHash. To show the general validity of our concept, in first experiments we evaluate using 60 raw data sets (5 individuals overall) consisting of two different handwritten semantics (arbitrary Symbol and fixed PIN). Experimental results demonstrate that reconstructed raw data produces an EERreconstr. in the range from 30% to 75%, as compared to non-attacked inter-class EERinter-class of 5% to 10% and handwritten PIN semantic can be better reconstructed than the Symbol semantic using this new technique. The security flaws of the Biometric Hash algorithm are pointed out and possible countermeasures are proposed.

Keywords: Biometric Hashing, Online Handwriting, Vulnerabilities, Reproducibility, Security.

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 178–190, 2010.

1 Introduction and Motivation

A variety of biometric identification and verification systems based on fingerprints, iris, voice etc. were introduced during the last years. In this paper we discuss the dynamic biometric modality handwriting. As for all biometric identification and verification systems, it is crucial to protect the original biometric raw data (templates) in order to prevent all kinds of misuse of individual and personal data. Identity theft is only one example, out of many others, that can be done with eavesdropped information. However, due to the variability of biometric data, templates cannot easily be protected by common cryptographic hash algorithms, like they are used in common password authentication systems comparing two passwords in hash domains. The variability (intra-class variability) has to be taken into account to ensure the reproducibility and protection of the template. One possible method to ensure reproducibility and simple template protection is for example the Biometric Hash algorithm for handwriting, originally introduced in [1] with further discussions in [2]. The goal of this method is to transform intra-subject biometric data, subject to variability, into stable and individual hash vector values; an overview is given in section 2. Although not originally suggested as template protection method in [1], we consider the application of the Biometric Hash scheme for template protection; due to its similar properties as cryptographic hash functions (see section 2).

Motivated by work from [3] and [6] for fingerprint modality, we investigate the generation of raw data based on a reference BioHash, which has the property to produce identical Biometric Hash values as the reference. Our approach is to exploit weaknesses from the intra-class-compensation process within the Biometric Hash generation, which allows different sets of raw data to produce an identical Biometric Hash value as the reference. First experiments on our new approach show the possibility to reconstruct such valid raw data.

Our proposed method is similar to a ciphertext-only-attack on cryptographic hashes, as described in the literature; see e.g. Bishop [5]. The ciphertext-only-attack depicts a scenario where an attacker has access to a collection of hashes and tries to determine the plaintext out of it. In our case we deal with an adaptive ciphertext-only-attack where side information such as system parameters for the BioHash algorithm is given. We further discuss our attack on the Biometric Hash algorithm for handwriting introduced in [1] where the Interval Matrix (IM) represents the side information in terms of system parameters of a BioHash algorithm.

The structure of the paper is composed as follows. Section 2 gives an overview to the BioHash algorithm for handwriting. We discuss potential vulnerabilities of the algorithm and consider possible attack methods based on these flaws. In section 3, the detailed design for the reconstruction of raw data is presented. The experimental evaluation is introduced in section 4, results and discussion are summarized. Finally, we present a conclusion, a comparison to the achieved results for fingerprints in [3] and our future work based on our findings.

2 Biometric Hash for Handwriting

The motivation to create a Biometric Hash algorithm is to develop a method which fulfils a similar task like a cryptographic hash does. However, due to the variability of

the input data a slightly different specification has to be made. The main differences and similarities of properties of both Biometric Hash and cryptographic hash are shown in Table 1. We denote *CH* as a cryptographic hash function, *BH* as a Biometric Hash function, *a* and *a'* as arbitrary digital input data (authentication information), *h* as a cryptographic hash, *b* as a BioHash and *P* and *P'* indicate two different persons.

Table 1. Brief overview of differences and similarities of cryptographic and biometric hashes

Property	Cryptographic Hash	Biometric Hash
a) Reproducibility	The hashes *h* and *h'* of a cryptographic hash function *CH* are identical, if *a* and *a'* are identical. *CH(a)=>h, CH(a')=>h', h=h',* if *a=a'*	The BioHashes *b* and *b'* of a Biometric Hash function *BH* are identical, if *a* and *a'* belong to the same person *P*. *BH(a)=>b, BH(a')=>b', b=b',* if *a* and *a'* belongs to *P*
b) Collision Resistance	It is difficult to find hashes *h* and *h'* of a cryptographic hash function *CH* which are identical, if *a* and *a'* are not identical. *CH(a)=>h, CH(a')=>h', h≠h',* if *a≠a'*	It is difficult to find BioHashes *b* and *b'* of a Biometric Hash function *BH* which are identical, if *a* and *a'* belong to two different persons *P* and *P'* respectively. *BH(a)=>b, BH(a')=>b', b≠b',* if *P≠P'*
c) Non-Reversibility	It should be computably hard to calculate or estimate the input data *a* out of a hash *h=>CH(a),* within a realistic time scale.	It should be computably hard to calculate or estimate the input data *a* out of a BioHash *b=>BH(a)*, within a realistic time scale.
d) Bit Sensitivity	Minor changes within *a* should have a massive effect on *h=>CH(a).*	Changes within *a* should have no effect on *b=>BH(a)*, if it derives from the same person *P*.

The Biometric Hash generation process differs in comparison to the cryptographic hash generation by the characteristics (a) Reproducibility, (b) Collision Resistance and (d) Bit Sensitivity. In consideration of these characteristics the BioHash from [1] and [2] belongs to a class of "Fuzzy commitment schemes", where a certain scope of variability is tolerated to get to the same result (introduced by Al-saggaf et al. in [7]).

The basic idea behind the BioHash algorithm is to extract a set of statistical feature values from actual handwriting samples and to find a parameterized transform function for mapping of these values to a stable hash value space. A workflow on the algorithm for handwriting is shown in Figure 1.

A human handwritten input (e.g. a handwritten signature) is acquired by a sensor which transforms pen positions and pressure into an analogous electrical signal (in the following we briefly summarize the main steps from [1]). This signal is converted by an analog-digital converter into a digital signal *a*, i.e. the digital raw data. Such digital raw data *a* is composed of time-dependent pen positions and corresponding pressure and angle values. Within the Biometric Hash generation process a feature extraction function determines statistical features based on the raw data *a*. Statistical features define characteristics such as *total-write-time*, *total-number-of-event-pixels* or *maximum-pressure* for example; a complete list of all actual used features can be found in table 2.

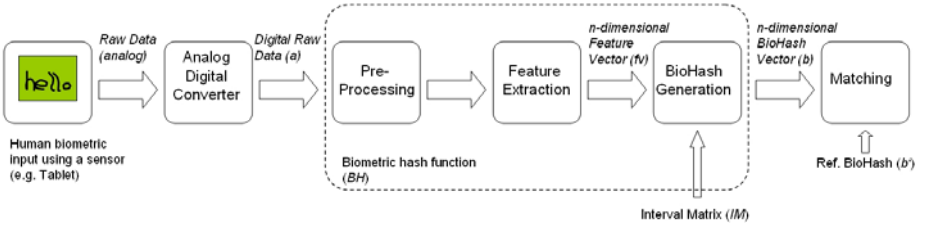


Fig. 1. General workflow of biometric bash generation

All determined features are collected in a feature vector of a dimension of n, whereas n denotes the number of statistical features being used during the extraction process. With help of an Interval Matrix (IM), these statistical features are mapped into a n-dimensional Biometric Hash vector (BioHash). The Interval Matrix is generated during an enrollment process for each subject and consists of an Interval Length Vector $\measuredangle I$ and an Interval Offset Vector Ω: $IM = (\measuredangle I, \Omega)$. Its function is to compensate intra-class variability of a subject by mapping a certain value range into one specific value. Each feature possesses a related pair of Interval Length and Interval Offset, therefore the length of Interval Matrix and feature vector are equal. The mapping for each feature element fv_i within a feature vector fv into a BioHash element b_i based on the Interval Matrix (Interval Length Vector and Offset Vector) is described in the following Equation 1, whereas i denote the index from 1 to n.

$$b_i = \left\lfloor \frac{fv_i - \Omega_i}{\Delta I_i} \right\rfloor \tag{1}$$

The result of a Biometric Hash generation process is an n-dimensional Biometric Hash vector b (BioHash). In verification mode this BioHash b is compared to a reference BioHash b_{ref}. The matching can be done, for example, by calculating the Hamming distance (amount of equal vector elements) between b and b_{ref}. Reference BioHash b_{ref} and Interval Matrix (IM) is stored for each user in a database.

A more detailed description on the BioHash algorithm is given in [1] and a further discussion in [2]. The BioHash cannot only be generated out of personal handwriting such as signatures, it is also possible or even advised to use pass phrases, pseudonyms, symbols or Personal Identification Numbers (PIN). These alternative handwriting samples are called semantics. It has been observed in [1] that these kinds of semantics produce similar recognition accuracy as compared to handwriting signatures, without disclosing the true identity of the writer. General comment: In the original design of the BioHash neither of the aspects of irreversibility nor attack scenarios has been considered. Therefore we address these aspect in this paper.

2.1 Potential Design Vulnerabilities of the BioHash for Handwriting

Our idea is to consider the Biometric Hash scheme as a method for template protection and to analyze the vulnerabilities for the reconstruction of raw data. We do so based on the assumption that an attacker has compromised a biometric based verification

Table 2. List of all features used during the BioHash generation process. Note: our attack classifies features into *cb – calculated basic feature, ib – interactive basic feature* and *gc – genetically calculated features* which are explained in the following sections.

fv_i: i=	Parameter Description	fv_i: i=	Parameter Description
1 (cb)	Total writing time in ms	53(gc)	Numeric Integration of Y values for 3rd one-fifth time period
2 (cb)	Total number of event pixels	54 (gc)	Numeric Integration of Y values for 4th one-fifth time period
3 (gc)	Image Width * 1000 DIV Height	55 (gc)	Numeric Integration of Y values for 4th one-fifth time period
4 (cb)	Average velocity in x direction in 1000 * pixels / ms	56 (gc)	Average Pen Down Pressure normalized to 1 * 1000
5 (cb)	Average velocity in y direction in 1000 * pixels / ms	57 (gc)	Average PenUp Pressure normalized to 1 * 1000
6 (ib)	Number of consecutive pen-down segments	58 (gc)	Baseline Angle of the Sample
7 (gc)	Minimum absolute x-velocity during sample	59 (gc)	Histogram of Y for Zone 1 in % * 100
8 (gc)	Maximum absolute x-velocity during sample	60 (gc)	Histogram of Y for Zone 2 in % * 100
9 (gc)	Minimum absolute y-velocity during sample	61 (gc)	Histogram of Y for Zone 3 in % * 100
10 (gc)	Maximum absolute y-velocity during sample	62 (gc)	Area(ConvexHull) vs. Area(BoundingBox) * 1000
11 (gc)	Centroid of horizontal pen position in bounding box	63 (gc)	Area(ConvexHull(Segments)) vs. Area(ConvexHull(Sample)) *
12 (gc)	Centroid of vertical pen position in bounding box	64 (gc)	Area(ConvexHull(Segments)) vs. Area(BoundingBox) * 1000
13 (gc)	Distance of Centroid from origin	65 (gc)	PathLength(ConvexHull) vs. PathLength(BoundingBox) * 1000
14 (ib)	Maximum absolute pressure occurred during writing	66 (gc)	PathLength(ConvexHull(Seg.)) vs.
15 (gc)	Centroid of horizontal pen position	67 (gc)	PathLength(ConvexHull(Seg.)) vs. PathLength(BoundingBox)
16 (gc)	Centroid of vertical pen position	68 (gc)	Histogram of X for left in % * 100
17 (gc)	Distance of Centroid from origin	69 (gc)	Histogram of X for right in % * 100
18 (gc)	Horizontal azimuth of centroid from origin	70 (gc)	Amount of maxima in X direction
19 (ib)	Maximum absolute altitude of pen occurred	71 (gc)	Amount of minima in X direction
20 (ib)	Minimum absolute altitude of pen occurred	72 (gc)	Amount of maxima in Y direction
21 (ib)	Maximum absolute azimuth of pen occurred	73 (gc)	Amount of minima in Y direction
22 (ib)	Minimum absolute azimuth of pen occurred	74 (gc)	Ratio of maxima in X direction vs. maxima in Y directions
23 (ib)	Average Writing Pressure relative to MaxPressure	75 (gc)	Ratio of minima X direction vs. minima directions
24 (ib)	Average Azimuth of pen projected on writing plane	76 (gc)	Amount of crossing points (intersections)
25 (ib)	Average Altitude of pen above the writing plane	77 (gc)	Amount of intersections with line at 1st quarter of X
26 (gc)	Normalized Average velocity in x direction in pixels	78 (gc)	Amount of intersections with line at 2nd quarter of X
27 (gc)	Normalized Average velocity in y direction in pixels	79 (gc)	Amount of intersections with line at 3rd quarter of X
28 (ib)	Absolute cumulated Pen-up time in ms	80 (gc)	Amount of intersections with line at 4th quarter of X
29 (gc)	Ratio of Pen-ups by total write time * 1000	81 (gc)	Amount of intersections with line at 1st quarter of Y
30 (gc)	Total Number of Sample Values	82 (gc)	Amount of intersections with line at 2nd quarter of Y
31 (gc)	Total absolute Path Length in Pixels	83 (gc)	Amount of intersections with line at 3rd quarter of Y
32 (gc)	Number of pixels in first row, first column	84 (gc)	Amount of intersections with diagonal line (up. left to bottom right)
33 (gc)	Number of pixels in first row, second column	85 (gc)	Amount of intersections with diagonal line (up. right to bottom left)
34 (gc)	Number of pixels in first row, third column	86 (gc)	Ratio of distance start/end to path length
35 (gc)	Number of pixels in first row, fourth column	87 (gc)	Ratio of distance min(X)/max(X) to path length * 1000
36 (gc)	Number of pixels in second row, first column	88 (gc)	Ratio of distance min(Y)/max(Y) to path length * 1000
37 (gc)	Number of pixels in second row, second column	89 (gc)	Ratio of distance start-centroid to end-centroid * 1000
38 (gc)	Number of pixels in second row, third column	90 (gc)	Mapping of maxima/minima in X to a value
39 (gc)	Number of pixels in second row, fourth column	91 (gc)	Mapping of maxima/minima in Y to a value
40 (gc)	Number of pixels in third row, first column	92 (gc)	Mapping of maxima/minima in P to a value
41 (gc)	Number of pixels in third row, second column	93 (gc)	Mapping of maxima/minima in A to a value
42 (gc)	Number of pixels in third row, third column	94 (gc)	Range of all stroke points
43 (gc)	Number of pixels in third row, fourth column	95 (gc)	Pixels inside radius 1/3*Boundingbox to point with least
44 (gc)	Numeric Integration of normalized X values	96 (gc)	Pixels inside radius 1/3*Boundingbox to point with most neighbour
45 (gc)	Numeric Integration of normalized Y values	97 (gc)	Pixels inside radius 1/3*Boundingbox to point with average
46 (gc)	Numeric Integration of X values for 1st one-fifth tp.	98 (gc)	Average angle of all cross-point-angles between 0-30°
47 (gc)	Numeric Integration of X values for 2nd one-fifth tp.	99 (gc)	Average angle of all cross-point-angles between 31-60°
48 (gc)	Numeric Integration of X values for 3rd one-fifth tp.	100 (gc)	Average angle of all cross-point-angles between 61-90°
49 (gc)	Numeric Integration of X values for 4th one-fifth tp.	101 (gc)	Angle count of all cross-point-angles between 0-30°
50 (gc)	Numeric Integration of X values for 4th one-fifth tp.	102 (gc)	Angle count of all cross-point-angles between 31-60°
51 (gc)	Numeric Integration of Y values for 1st one-fifth tp.	103 (gc)	Angle count of all cross-point-angles between 61-90°
52 (gc)	Numeric Integration of Y values for 2nd one-fifth tp.		

system and has access and knowledge to username, reference BioHash b_{Ref}, and Interval Matrix (*IM*) for each registered individual. The operating principle of the BioHash algorithm is published and is accessible for everyone who is interested in (Kerkhoff principles).

The first thing that attracts our attention is the Interval Matrix and the mapping function, which maps the feature vector into a BioHash. When BioHash b_{Ref} and corresponding Interval Matrix *IM* are given, we can perform a reverse mapping to create a feature vector fv_{calc}. If we convert Equation 1 according to fv_i, we calculate the lower limit of a value range which can be mapped to b_i. By adding the half of $\measuredangle I_i$ to it, we

compute the middle of the value range (see Figure 2) in feature space. Equation 2 formulates this approach, whereby fv_i is replaced with $fv_{calc,i}$ because they are not necessarily equal, related to the rounding in equation 1.

$$fv_{calc_i} = b_{ref_i} \cdot \Delta I_i + \Omega_i + \frac{\Delta I_i}{2} \tag{2}$$

Figure 2 gives a visual point of view on the reverse mapping done in Equation 2. Using this method, it is possible to calculate a complete feature vector fv_{calc}. Due to the fact that fv_{calc} is determined from b_{Ref} and corresponding *IM*, it can be mapped with help of *IM* to b_{Ref} again and therefore be used to reconstruct raw data, based on it.

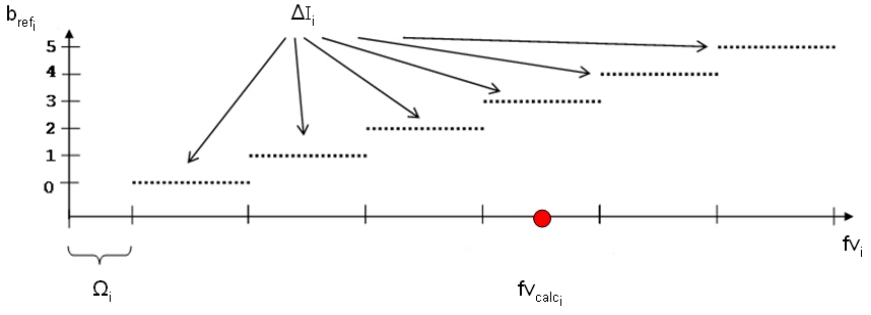


Fig. 2. Example of backward calculation (reverse mapping) of a feature vector element $fv_{calc,i}$ based on the corresponding reference BioHash $b_{ref,i}$, Interval Length ΔI and Interval Offset Ω_i

If an attacker takes advantage of this vulnerability (reverse mapping) he can reduce his work on reconstructing raw data based on that calculated feature vector fv_{calc}, i.e. in feature space rather than on the BioHash. The next section shows a possible method to achieve this.

2.2 Attack Considerations

By analyzing the feature vector and it's consisting statistical feature it turns out that some features can be used to build a basic structure to reconstruct raw data. For that reason it is necessary to calculate or determine some fundamental data. The idea is now to find fundamental data that helps to form the basic structure by looking into *total-write-time*, *total-number-of-event-pixels*, *maximum-value-in-x-direction* and *maximum-value-in-y-direction*. The first two values can easily be determined by reading the corresponding values out of the feature vector fv_{calc}. Maximum value in x and y direction can be calculated out of the features *total-write-time* and *average-velocity-in-x-direction* respectively *average-velocity-in-y-direction*. These four mentioned features, which we refer to as *calculated basic features*, form the basic raw data structure (see Table 2 marked with *cb*).

Based on this basic structure it is now possible to manually implement other features as well. This can be done as follows: an experienced user chooses a specific feature fv_i he wants to implement from fv_{calc}, he changes the basic raw data structure corresponding to that feature fv_i then he determines a temporary feature vector fv_{temp}

for this new set of raw data and compares feature fv_i from fv_{temp} with feature fv_i of feature vector fv_{calc}. If they are not the same, he changes the basic raw data structure again, until they match. This procedure is time-consuming and needs a lot of experience in interpreting and manipulating raw data in a way to fulfill a certain feature. However, by determining these features manually, we expect that it leads to more natural results within the artificial raw data. To perform a first test to determine the success tendency of this approach under controlled conditions, we introduce a time limit for this interactive process in our experiments as described in section 4.1.

Initial considerations about which features can be set manually revealed that pressure and angle based features are our first choices. That is because they are independent from horizontal and vertical (x/y coordinates) based features, which make up more than *80%* of all features used and described in [1]. Once the pressure and angle based features are implemented inside the raw data they can be locked, so that they cannot be changed anymore in the further reconstruction process. We call these pressure based, angle based and all other features that can be implemented manually within a raw data structure *interactive basic features* (see Table 2 marked with *ib*). All *calculated* and *interactive basic features* are additional summarized in Table 3.

Table 3. Classification of features based on the feature list in Table 2

Feature class	Description	Dedicated features
Calculated Basic Features (fv_{cb})	Features that form the basis for a raw data structure by calculating and determine fundamental data	$fv_{cb} = \{fv_1, fv_2, fv_4, fv_5\}$
Interactive Basic Features (fv_{ib})	Features that are implemented into the basic raw data structure manually	$fv_{ib} = \{fv_6, fv_{14}, fv_{19}, fv_{20}, fv_{21}, fv_{22}, fv_{23}, fv_{24}, fv_{25}, fv_{28}\}$
Genetically Calculated Features (fv_{gc})	All features that are implemented into raw data by a genetic algorithm.	$fv_{gc} = fv_{all} \setminus fv_{cb} \setminus fv_{ib}$
Overall features (fv_{all})	All features used during the BioHash determination process	$fv_{all} = \{fv_1, \ldots, fv_{103}\}$

Obviously, if *calculated* and *interactive basic features* are implemented into a set of raw data then all the remaining features have to be implemented as well to generate a feature vector that matches fv_{calc}. This can be done in many different ways (e.g. brute-force attack). Our first idea is to implement the remaining features into the raw data by using a genetic algorithm, first presented by Holland in 1975 [4]. We define features which are calculated this way as *genetically calculated features* (see also marked in Table 2 and listed in Table 3).

In the next subsection we focus on the description of our reconstruction approach based on the mentioned attack consideration by involving genetic algorithms.

3 Design Approach for Reconstruction

In this section we give a detailed description on our first design approach for reconstructing raw data out of a given BioHash and corresponding Interval Matrix, based on the vulnerability and feature classification, discussed in section 2.

Our approach can be divided into four major steps. The first step is to calculate the feature vector fv_{calc} as described in section 2.1. During the next step, we build a basic raw data structure based on the *calculated basic features* as introduced in section 2.2. The third step implies the implementation of *interactive basic features* into the basic raw data structure as presented in section 2.2. Finally, our last and fourth step is to determine all remaining features based on the raw data structure, by using a genetic algorithm, which is detailed later on in this section. These four steps build our new design approach for reconstructing raw data and are depicted in Figure 3.

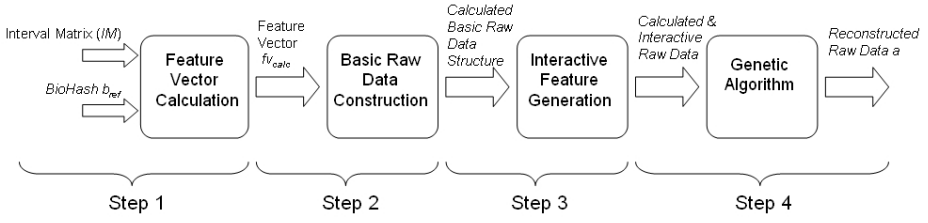


Fig. 3. Illustration of our new approach to reconstruct raw data out of BioHash b_{ref} and corresponding Interval Matrix (*IM*)

Step 1: With help of given BioHash and corresponding Interval Matrix (*IM*) we calculate the feature vector fv_{calc} based on equation 2. As already mentioned in section 2.1, fv_{calc} is not necessarily, or even likely to be equal to the original feature vector, but leads to the same BioHash b_{ref}, when mapped with the corresponding Interval Matrix. The feature vector fv_{calc} forms the basis for our reconstruction process described in steps 2 to 4.

Step 2: Based on fv_{calc} we build a basic raw data structure using the *calculated basic features*. This helps to implement additional features since a basic raw data structure is now given and has only to be modified.

Step 3: A user tries to implement all *interactive basic features* into the basic raw data structure, in a defined limited period of time to have a first evaluation with controlled time constrains. Depending on the experience of a user and constellation of fv_{calc} it is possible that not all *interactive basic features* can be implemented due to this time constant.

Step 4: All remaining features fv_{gc} are transcribed into the raw data structure by a genetic algorithm (GA), which is described as follows. A start population is composed of individuals, which are randomly generated. In order to generate individuals that represent a realistic signature, they have to be build of continuous horizontal and vertical signal components. Motivated by Galbally et al. in [8], where synthetic signatures are generated based on spectral analysis; we created an algorithm which generates different, almost realistic signatures. Due to the fact that we already generated pressure and angle values (implemented with the *calculated* and *interactive basic features*), only horizontal and vertical raw data values needs to be added. All individuals now possess an implementation of the same basic features and randomly generated continuous horizontal and vertical signals. The fitness function for each individual is based on the method “survival of the fittest”. To determine the fittest

individuals, a BioHash is calculated for each individual (based on given Interval Matrix) and compared, using the normalized Hamming distance, to the reference BioHash b_{ref}. Obviously, in our case, the normalized Hamming distance between any 2 BioHash vectors is defined by the number of non-equal vector components divided by the vector's dimension. Individuals which achieve the highest scores are defined as the fittest. Only the fittest are taken into the next round combined with a defined survival rate. We utilize the BioHash Hamming distance as fitness function, since this is calculated based on the original feature vector values, whereas fv_{calc} is just estimation. We use the genetic operators' mutation and crossover to create new generations and modify raw data in such a way that the before mentioned *calculated* and *interactive basic features* are preserved. Mutations are not always applied to individuals during the generation creation; it is randomly controlled using a mutation rate. The mutation operator changes only a part of an individual when it occurs. We only use a *one-point-crossover* genetic operator during a creation of a new generation. It swaps sub-sections of two individuals, whereby these sub-sections are at the same position within the two individuals. The genetic algorithm terminates if one of the following conditions is reached: (1) returned BioHash and reference BioHash are equal (or a specific threshold is reached), (2) individuals do not change any more in a positive way (higher matching scores) after multiple generation cycles or (3) a specific defined amount of generation cycles has been passed (e.g. x=100 iterations). An overview on the GA workflow is shown in Figure 4.

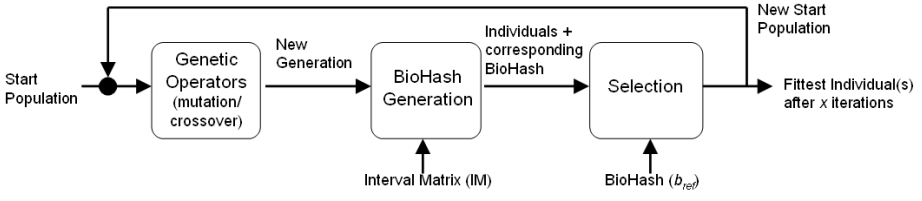


Fig. 4. Basic workflow of the used genetic algorithm during raw data reconstruction

After all four steps are performed; one or more raw data sets were reconstructed, depending on the GA result (two or more individuals achieve the highest fitness score). In order to evaluate our new design approach, we perform experiments, as described in the next section.

4 Experimental Evaluation

In this chapter we summarize our first experiments to evaluate the proposed reconstruction attack. Our goal is to see how successfully we can generate artificial biometric handwriting raw data on given Interval Matrix and corresponding BioHash by measuring the archived $FAR_{reconstr.}$ with the reconstructed raw data. First we define the experimental settings. Secondly we introduce our methodology to provide a comparative study of the achieved results to the general verification performance. Thirdly, results are presented and discussed.

4.1 Overall Settings

The biometric database of our initial tests consists of *5* subjects, which each have donated *6* handwriting samples for two different semantics (PIN and Symbol). The given PIN is a sequence of the five digits *77993*. Using this semantic, the individual kind of writing plays a more important role than the content to recognize a person as its self or distinguish him/her from other users. The freely chosen Symbol is based on individual creative characteristics and provides a knowledge based component in form of the sketched object (e.g. order of single strokes to create the symbol). In order to create the reference data, an Interval Matrix and a reference BioHash are calculated for each person using the first five handwriting samples. The remaining sixth sample is applied for verification.

In the next paragraphs we provide details about the settings for our experiments during the reconstruction process for step 3 and 4. In step 1 and 2 no parameter needs to be set. The dimension of Interval Matrix is 2x103 and thus the size of the BioHash vector 103; therefore 103 features are used during the BioHash determination and raw data reconstruction in our tests.

During the third step (setting interactive feature) in the reconstruction process, in our first test one human attacker and forger tries to implement the *interactive basic features* into a basic raw data structure, for this procedure we define a maximum processing time of *10* minutes.

Within step four of the reconstruction process, the settings for the genetic algorithm is as follows: start population of *100* individuals, *90%* survival rate, the recombination rate and mutation rate is *10%* (*10%* of all individuals are recombined or mutated). The genetic algorithm terminates if a *100%* matching rate is accomplished or the matching score does not change any more in a positive way after two full cycles (all features). The matching to determine the fittest individual is accomplished by calculating the Hamming distance as described in section 3. The fittest individual represents the reconstructed raw data and is used during verification together with genuine raw data. In Table 4 are all settings summarized.

Table 4. Overview of all overall settings

General Settings		Step 3: Interactive feature generation	
Number of b_{ref} per user:	10 (5 per semantic class PIN/Symbol)	Forger:	One computer science student (age: 26) as interactive attacker
SizeOf b_{ref}:	103	Forgery time:	≤ 10 min
SizeOf fv_{cb}:	4 features	Step 4: Genetic Algorithm	
SizeOf fv_{ib}:	10	Start population	100 individuals
SizeOf fv_{ge}:	89	Survival rate	90%
SizeOf fv_{all}:	103	Recombination rate	10%
		Mutation rate	10%
		Termination criteria	100% match or no improvement of fitness function after 2 generations

4.2 Evaluation Methodology and Measurements

In order to compare the performance of a verification using the reconstructed raw data with the verification using genuine raw data, biometric error rates FRR/FAR and EER are calculated.

The FRR (false rejection rate) describes the ratio between the number of false rejections of authentic persons and the total number of tests. Generally, the FAR (false acceptance rate) is the ratio between number of false acceptances of non-authentic persons and the entire number of authentication attempts. In our evaluation, we perform two kinds of false acceptance tests: firstly, interclass FAR ($FAR_{inter\text{-}class}$) errors are calculated by comparing BioHash values of all subjects against each other and analyzing false acceptances against normalized Hamming distance thresholds. Secondly, we determined false acceptances generated by the reconstructed raw data against the same threshold, in the following denoted as $FAR_{reconstr.}$.

For a comparative analysis of verification performance, the EER (equal error rate) is a common measurement in biometrics. EER denotes the point in error characteristics, where FRR and FAR yield identical value. In the further discussion of our experimental results, we analyze error rate diagrams, which consists of error rates graphs for $FAR_{inter\text{-}class}$, $FAR_{reconstr.}$, and FRR for each writing semantics to illustrate $EER_{inter\text{-}class}$ respectively $EER_{reconstr.}$.

4.3 Results and Discussion

The results of our test with different semantic classes are displayed in Figure 4. In average the reconstructed raw data produces an $EER_{reconstr.}$ of *75%* for the semantic PIN, whereas the original user based raw data leads to an $EER_{inter\text{-}class}$ of only *10%* (Figure 5) for inter-class verification. This means, that random forgeries (interclass tests) cause lower false acceptance than our achieved reconstructed raw data.

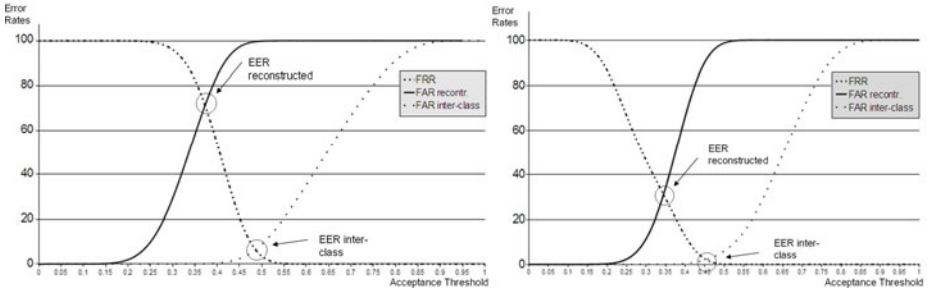


Fig. 5. FRR, $FAR_{inter\text{-}class}$ and $FAR_{reconstr.}$ for PIN (left) and Symbol (right)

For semantic class *symbol* the reconstructed raw data leads to an $EER_{reconstr.}$ of *30%* and an interclass $EER_{inter\text{-}class}$ of approximately *5%* (see Figure 4 right). It shows that the semantic class *Symbol* is more resistant to our attack based on reconstructed raw data then the semantic class *PIN*. The results reflect the first attempt to generate biometric raw data corresponding to a given Interval Matrix and corresponding reference BioHash. Please note that this is only a first attempt of producing reconstructed raw data and can just point into a direction because of the relatively limited amount of semantics and users during the test. However, these results show that in our case the reconstructed raw data creates a significantly higher FAR then the original user based raw data, which allows attackers to reproduce BioHash values in *75%* of all trails.

5 Conclusion and Future work

In this paper we have suggested a method for reconstructing biometric raw data from given BioHash values, by using features derived from user interaction and genetic algorithm for approximation. Our work reveals some vulnerabilities of the Biometric Hash algorithm for handwriting, introduced in [1]. Based on these vulnerabilities and motivated by earlier work e.g. for generating artificial forgeries [6], it is possible to design and implement an attack to generate raw data based on *calculated* and *interactive basic feature* determination, with an additional genetic algorithm. Our first experiment shows that such generated forgeries produce an $EER_{reconstr.}$ in the range of 30% to *75%* as compared to non-attack inter-class $EER_{inter\text{-}class}$ of *5%* to *10%*. If we compare our results with the one made in [3] by Galbally et al., where a fake fingertip was created from an image reconstructed from a minutiae template, we recognize similarities. The test set is in comparison to 5 Symbols and 5 PINs limited to 10 different fingerprints. They are using an attack rate to present the evaluation results, instead of an EER, because the FAR for all 5 considered thresholds are *0%*. By using the same measurement methodology we achieve similar results. The success-attack-rate in [3] is higher (*30%-100%*) compared to our attack rates (*0%-70%*) as we expected, due to the fact that fingerprint based recognition systems are more accurate in distinguishing different fingerprint samples.

In order to eliminate the vulnerability to the BioHash algorithm we suggest to not use *calculated basic features* during the BioHash generation process. It is more difficult to reconstruct raw data using our new approach if *calculated basic features* are nonexistent or derive to complex additional features.

In our future work we will run tests with more semantic classes and users to extend our first results. We also plan to examine all features being used during the BioHash generation to find more that can be used as *calculated basic features* or *interactive basic features*. The development of a more advanced genetic algorithm is also planned in the future. Another consideration is the development of an automatic approach, which makes an interactive interference needless.

Acknowledgements

This work is supported by the German Federal Ministry of Education and Research (BMBF), project “OptiBioHashEmbedded”) under grant number 17N3109). The content of this document is under the sole responsibility of the authors.

References

1. Vielhauer, C.: Biometric User Authentication for IT Security: From Fundamentals to Handwriting, Springer, New York (2006)
2. Vielhauer, C. and Steinmetz, R. and Mayerhoefer, A.: Biometric Hash based on Statistical Features of Online Signatures: In: Proceedings of the IEEE International Conference on Pattern Recognition (ICPR), vol.1, pp.123-126, (2002)

3. J. Galbally, R. Cappelli, A. Lumini, D. Maltoni and J. Fierrez: Fake Fingertip Generation from a Minutiae Template, in *Proc. Intl. Conf. on Pattern Recognition, ICPR*, Tampa, USA, (2008)
4. Holland, J.H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology. Control and Artificial Intelligence: MIT Press, 1995, First Published by University of Michigan Press (1975)
5. Bishop, M.: Computer Security, Addison-Wesley, Boston, U.S.A, ISBN 0-201-44099-7, (2003)
6. R. Cappelli, A. Erol, D. Maio and D. Maltoni: Synthetic Fingerprint-image Generation, in proceedings 15th International Conference on Pattern Recognition (ICPR2000), Barcelona, vol.3, pp.475-478, (2000)
7. Alawi A. Al-saggaf, Acharya H. S.: A Fuzzy Commitment Scheme, In: Proc. IEEE International Conference on Advances in Computer Vision and Information Technology, India (2007)
8. Galbally, J., Fierrez, J., Martinez-Diaz M. and Ortega-Garcia, J.: Synthetic Generation of Handwritten Signatures Based on Spectral Analysis, in Defense and Security Symposium, Biometric Technologies for Human Identification, BTHI, Proc. SPIE, Orlando, USA (2009)

Privacy Preserving Key Generation for Iris Biometrics*

Christian Rathgeb and Andreas Uhl

Multimedia Signal Processing and Security Lab
Department of Computer Sciences
University of Salzburg, A-5020 Salzburg, Austria
{crathgeb,uhl}@cosy.sbg.ac.at

Abstract. In this work we present a new technique for generating cryptographic keys out of iris textures implementing a key-generation scheme. In contrast to existing approaches to iris-biometric cryptosystems the proposed scheme does not store any biometric data, neither in raw nor in encrypted form, providing high secrecy in terms of template protection. The proposed approach is tested on a widely used database revealing key generation rates above 95%.

1 Introduction

Generic key management systems perform key release based on alternative authentication – passwords or PINs [1]. Hence, cryptographic keys as well as encrypted information are only as secure as the passwords or PINs used to release these. As a consequence, biometric authentication has been introduced to cryptographic systems. So-called biometric cryptosystems aim at extracting or binding cryptographic keys out of or with biometric traits. With respect to iris biometrics [2] several approaches have been proposed [3,4,5]. Yet, most existing approaches (based on key-binding schemes) are constrained to store biometric data bound with cryptographic keys involving the application of error correction codes [6,7]. Since biometric information, especially within iriscodes, is not distributed uniformly random and error correction codes underlie specific structures, stored templates are found to suffer from low entropy [8]. This means, potential imposters could get to compromise cryptographic keys and, furthermore, reconstruct biometric templates or decrypt any kind of crucial information.

The contribution of this work is the proposal of a new technique for generating cryptographic keys out of iris textures. Our system does not require the storage of biometric data, neither in raw nor in encrypted form. Thus, we provide high security in terms of template protection in contrast to existing approaches. Generated biometric keys are long enough to be applied in generic cryptographic systems (e. g. AES) and, in addition, these are fully revocable.

* This work has been supported by the Austrian Science Fund, project no. L554-N15 and the Austrian Grid Project II is gratefully acknowledged.

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 191–200, 2010.

The remainder of this paper is organized as follows: an overview of iris-biometric cryptosystems is given in Section 2. In Section 3 our proposed system is described in detail. Experimental results are presented in Section 4. Section 5 concludes.

2 Previous Work

In the past years several approaches to biometric cryptosystems have been proposed applying different biometric modalities (key ideas are summarized in [1]). Focusing on iris biometrics, most approaches aim at binding constant features with cryptographic keys where biometric variance is overcome by means of error correcting codes this work.

Davida *et al.* [6] were the first to propose a biometric cryptosystem applied to iris biometrics, which they refer to as "private template scheme". In their private template scheme a representative feature vector is concatenated with an error correction code. Additionally, a hash value of the feature vector and personal information is stored in the template. During authentication, personal information is verified and error correction information is used to reconstruct the original feature vector, serving as cryptographic key. Experimental results are omitted.

In their "fuzzy commitment scheme" Juels and Wattenberg [7] provided a theoretical basis for binding and retrieving cryptographic keys in a fuzzy manner. Extracted binary biometric features are XORed with a cryptographic key prepared with error correction information. The resulting bitstream, which is expected to be secure, is stored in a database together with a hash of the key. Biometric features which are "close" enough to that presented at registration are capable of reconstruction the cryptographic key. This is done by XORing features with the stored template and applying error correction decoding, where the resulting key is hashed and tested against the previously stored one. Hoa *et al.* [3] applied the fuzzy commitment scheme to 2048-bit iriscodes to bind and retrieve 140-bit keys. For 70 persons a FRR of 0.47% and zero FAR is reported. In previous work [9] we provide a systematic approach to the construction of fuzzy commitment schemes based on iris biometrics. Experimental results provide a FRR of 4.64% and 6.57% according to a zero FAR for adopting the fuzzy commitment scheme to two different iris recognition algorithms. In other work [10] we propose a context-based texture analysis in order to detect the most reliable parts in iris textures out of which cryptographic keys are constructed. Applying error correction codes to overcome remaining variance we obtained a FRR of 6.53% at a zero FAR for the extraction of 128-bit keys. Bringer *et al.* [11] apply two-dimensional iterative min-sum decoding in a fuzzy commitment scheme for the binding of 128-bit cryptographic keys. The authors report a FRR of 9.1%. In general, iris-biometric cryptosystems utilize binary iriscodes where biometric variance is suppressed applying error correction codes. However, iriscodes must not be expected to be uniformly random and error correction codes exhibit distinct structures. Due to these facts biometric templates have been found to suffer from low entropy [8].

The proposed system is based on a different concept that has been applied to online signature [12], face biometrics [13] and iris [14], which we refer to as "interval-mapping scheme". The key idea of an interval-mapping scheme is to extract several real-valued features at the time of enrollment and construct intervals for each extracted feature, using several enrollment samples. These intervals are encoded and cryptographic keys are extracted out of a another biometric measurement by mapping features into intervals. Additionally, genuine intervals are hidden by adding fake intervals in appropriate ranges. Interval-mapping schemes represent key-generation schemes in which stored intervals (which are secured by adding fake ones) are applied as helper data to generate keys. While interval-mapping schemes provide high security in terms of template protection reported results are found unpractical so far, for example, in [14] we achieved a FRR of 36.5% for the generation of 128-bit iris-biometric keys.

3 System Architecture

Based on ideas presented in [12,13,14] we construct an interval-mapping scheme for biometric key generation based on iris biometrics. In order to obtain a real valued feature vector, which is essential for the construction of an interval-mapping scheme, texture analysis based on pixel-blocks is applied. Subsequently, image quality measurement techniques are utilized to construct adequate intervals for the extracted features. The most reliable pixel-blocks are detected through context-based texture analysis [10]. Encoded intervals of each feature element and a position mask of the most reliable pixel-blocks, arranged in a two-dimensional array, are stored as helper data. In Figure 1 an schematical illustration of the proposed system is shown. In the following subsections the key components of our scheme are described in detail.

In contrast to our existing work [14], in which we construct an interval-mapping scheme based on iris biometrics, we do not employ an existing feature extraction method, but construct one which is more suitable for biometric key generation. Additionally, we apply a context-based reliable component detection prior to the key generation process.

3.1 Preprocessing

The preprocessing procedure of the proposed system is based on the standard approach described in [15]. First of all the pupil of an eye is detected and the inner and outer boundaries of the iris are approximated. Subsequently, the pixels of the resulting iris (in form of a ring) are mapped from polar coordinates to cartesian coordinates in order to generate a rectangular iris texture of fixed length. Parts of iris textures which mostly comprise eyelashes or eyelids are discarded (315^o to 45^o and 135^o to 225^o). Finally, the resulting iris texture is enhanced by applying a global histogram stretching method to obtain a well-distributed texture. Figure 2 (a) shows an example of a preprocessed iris texture.

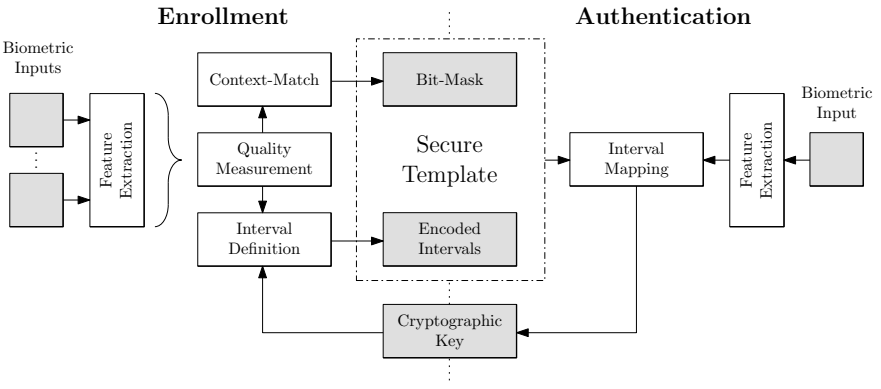


Fig. 1. Proposed System (Enrollment and Authentication): several biometric measurements are applied to detect the most reliable pixel-blocks in iris textures and intervals are constructed for these. At authentication these reliable features are extracted and mapped into intervals to generate a key.

3.2 Archetype and Reliable Component Selection

We calculate a so-called "archetype" which represents a preprocessed iris texture as a set of real-valued features. For this purpose the iris texture of the i-th user is divided into n rectangular $x \times y$ pixel-blocks. The resulting pixel-blocks are quantized by assigning the mean value of all pixel values to the whole block (note that the feature space depends on the number of possible grayscale values). In Figure 2 an example of this process is illustrated. That is, the archetype of an iris texture of the i-th user, denoted by A_i, is a set of real-valued features, which can be defined as,

$$A_i := \{f_{i1}, f_{i2}, ..., f_{in}\}. \tag{1}$$

During the enrollment process of the i-th user several iris images $I_{i1}, I_{i2}, ..., I_{ik}$ are captured where I_{i1} is used to generate the archetype A_i. For each $x \times y$ pixel-block of the remaining iris textures $(I_{i2}, I_{i3}, ..., I_{ik})$ the peak signal-to-noise ratio ($PSNR$) with respect to the according pixel-block of the archetype A_i is calculated (note that the remaining iris textures are not quantized). With the following two equations the $PSNR$ is defined:

$$MSE = \frac{1}{nm} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} ||I_1(i,j) - I_2(i,j)||^2 \tag{2}$$

where MSE denotes the mean squared error between two images I_1 and I_2 (in the proposed system the MSE is calculated between pixel-blocks). Subsequently, the $PSNR$ of the images I_1 and I_2 is calculated as,

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX_I^2}{MSE} \right) \tag{3}$$

(a) (b)

Fig. 2. Construction of the Archetype: (a) example of a preprocessed iris texture (b) the according archetype for 8×3 pixel blocks. The mean grayscale value of each 8×3 pixel block is assigned to the whole block.

where MAX_I is the maximum possible pixel value of both images. The $PSNR$ is commonly used to estimate the quality of a reconstructed image compared with an original image. This means, we interpret the archetype A_i as original image and estimate deviations of $x \times y$ pixel-blocks of each iris texture $I_{i2}, I_{i3}, ..., I_{ik}$, considered as noisy reconstructions of the pixel blocks of A_i. That is, for each pixel-block of A_i we obtain a number of $k-1$ $PSNR$ values, denoted by $p_{i1}, p_{i2}, ..., p_{ik-1}$. In order to gain a representative average $PSNR$ the mean of all $PSNR$ values calculated for one pixel-block, denoted by $\overline{p}_i$, is estimated as the $PSNR$ value of the pixel-block. In summary, for the features $f_{i1}, f_{i2}, ..., f_{in}$ of the n pixel blocks of the archetype A_i, $PSNR$ values $\overline{p}_{i1}, \overline{p}_{i2}, ..., \overline{p}_{in}$ are calculated. We found that the $PSNR$ of each pixel-block is a suitable measurement for the deviation of the feature value assigned to this block (we will verify this claim in our experimental result).

In the next step we determine the most reliable components of the first iris texture, hence the most stable features f_{ij}, of the archetype A_i. Obviously, those features which exhibit high $PSNR$ values are suitable. We go one step further and select those features for which surrounding features exhibit high $PSNR$ values, too. That is, a so-called context-based selection (as described in [10]) is performed. For this purpose the mean of the eight adjacent $PSNR$ values of each feature are assigned as new $PSNR$ value to the currently processed feature. In Figure 3 the neighboring $PSNR$ values which are taken into account are illustrated. As a result of the reliable component selection we store a two-dimensional bit-mask as first part of the template. This bit-mask points at the l most reliable features, where l is a predefined parameter. Thus, the l most stable features (those which exhibit the highest context-based $PSNR$ values) contribute to the key generation process, while features which underlie high variations are discarded.

3.3 Interval Construction and Key Generation

Once the most reliable features are extracted based on the assigned PSNR values, we calculate intervals for these. Since the $PSNR$ of two pixel-blocks is higher if those blocks tend to be very similar, the left and right border, $b_{L_{ij}}$ and $b_{R_{ij}}$, of the feature value f_{ij} are defined as,

$$b_{L_{ij},R_{ij}} := f_{ij} \pm 2^{\frac{c}{\overline{p}_{ij}}} \cdot d \tag{4}$$

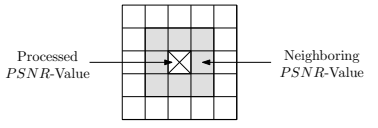


Fig. 3. Reliable Component Selection: context-based selection is performed to find the stable features

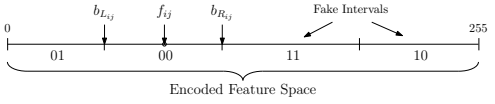


Fig. 4. Interval Construction: the feature space is divided into several encoded intervals of appropriate size

where c and d are predefined parameters for all users and have to be adjusted according to the applied block dimension and the feature space. In order to hide genuine intervals the whole feature space is filled (up) with fake intervals of appropriate sizes. Subsequently, features are encoded with one or several bits, depending on the feature space and on the range of calculated intervals. For example, if the average interval range would be 16 and 256 possible grayscale values are assigned to each pixel-block, $\log_2(256/16) - 1 = 3$ bits could be used to encode the genuine as well as the fake intervals. As second part of the template, genuine intervals together with fake intervals and the encoding of these are stored. In Figure 4 an illustration of a constructed interval for a single feature is shown. The concatenation of all bits used to encode the l genuine intervals form the cryptographic key associated with the i-th user. This means, by applying an appropriate encoding, the encoding of intervals can be adopted to any previously chosen cryptographic keys. In summary, the first part of biometric template of the i-th user consists of a bit-mask which points at locations of the l most reliable features f_{ij} of this user. Intervals of these l features as well as additionally added fake intervals form the second part of the template.

At the time of authentication another preprocessed iris texture is used to generate a cryptographic key. We first extract the according features $\hat{f}_1, \hat{f}_2, ..., \hat{f}_n$ in the same manner we calculated features of archetypes in the enrollment procedure. We then apply the stored bit-masks to locate the positions of valuable features (note that bit masks are distinct for each user). Subsequently, the selected features are mapped into intervals where according codewords are returned. By concatenating the bits of all returned codewords a biometric key is constructed. In order to provide rotation invariance for small head tilts we implement a circular pixel-wise shift of iris textures and perform key generation several times. If applicable the extracted key could be hashed and tested against a previously stored hash of the key in order to suppress the release of errored keys.

3.4 Cancellable Cryptographic Keys

The concept of "cancellable biometrics" was introduced by Ratha *et al.* [16]. If original biometric data is compromised it becomes useless because it can not be modified ex post. The key idea of cancellable biometrics is to store a transformed version of biometric data and, furthermore, perform the matching procedure in the transformed space. If transformed biometric data is compromised, transform

functions are changed, that is, the biometric template is updated. We do not store any biometric data, however, since extracted biometric keys are dependent on biometric features cancellable biometric keys have to be provided.

In our system cancellable keys are achieved by modifying the encoding of constructed intervals. As mentioned earlier, the encoding of intervals can be chosen at random during enrollment. Thus, different keys can be used for different applications. In case a cryptographic key is stolen, the encoding of intervals is changed, hence, the key is updated. To enhance security, a permutation of stored bit-masks could be introduced in order to hide positions of reliable components. Parameters of the inverse permutation could be stored on an additional token (e.g. a smart-card). Thus, the helper data of the proposed interval-mapping scheme provides the generation of fully revocable cryptographic keys.

4 Experimental Results

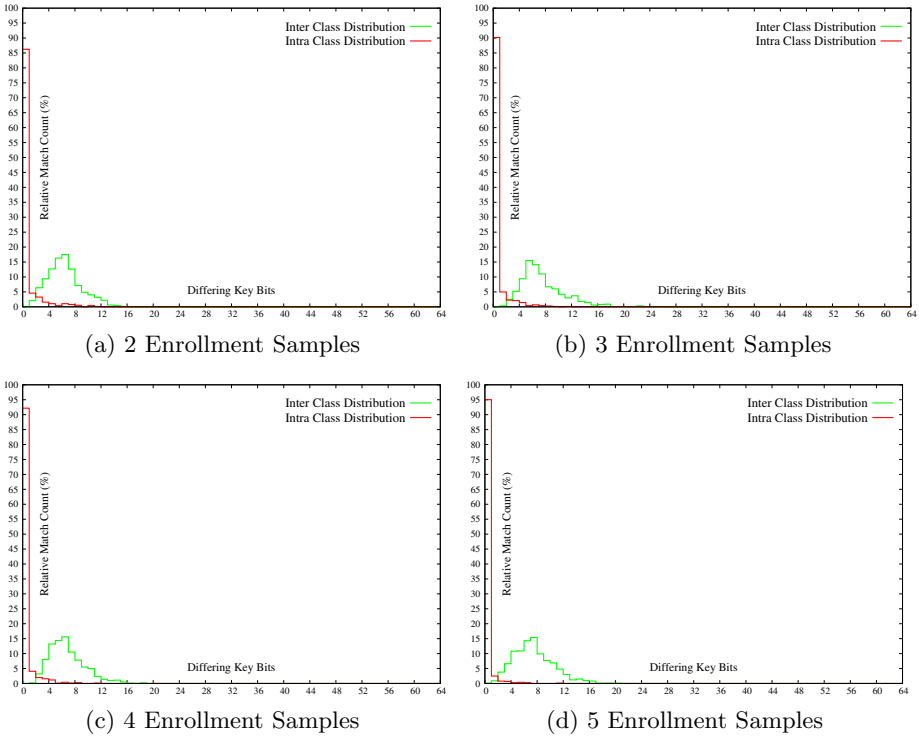
Experiments are carried out using the CASIAv3-Interval iris database[1] a test set of iris images of over two hundred persons. As previously mentioned, the top and bottom quarter of the iris are often hidden by eyelashes or eyelids, preprocessed iris textures are slitted from the right side (45^o to 315^o) and the left side (135^o to 225^o) resulting in textures of size 256×64 pixels possessing 256 grayscale values (see Figure 2 (a)). Hence, feature space of extracted features ranges from 0 to 255. For the construction of archetypes we use 12×6 pixels-blocks (best results were achieved for this dimension). It is found that the according PSNR values tend to range from 2.8 dB to 3.6 dB. For the above equation we set c=15 and d=3.5 so that

$$b_{L_{ij},R_{ij}} := f_{ij} \pm 2^{\frac{15}{P_{ij}}} \cdot 3.5. \tag{5}$$

For this setting of c and d the average width of intervals ranges around 64. Since the feature space ranges from 0 to 255, two bits are appropriate to encode one interval. In order to generate keys which are long enough to be used in generic cryptosystems (at least 128 bits), l is set to 64, according to the method described above. In this work we do not focus on generating biometric keys longer than 128 bits. At the time of authentication circular shifts are performed five pixels to the left and to the right.

The performance of the proposed system is measured in terms of key generation rates. The KGR of a biometric key-generation scheme defines the percentage of correctly generated keys for genuine users where no correct keys are released to non-genuine users. Since no error correction is involved, only hundred percent correct cryptographic keys are acceptable. In other words, the number of error key bits for genuine users must be zero. We tested our system for the aforementioned parameter setting using different numbers of enrollment samples. Distributions of genuine and non-genuine users are shown in Figure 5(a)-(d). For this evaluation

[1] The Center of Biometrics and Security Research, CASIA Iris Image Database, http://www.sinobiometrics.com



(a) 2 Enrollment Samples (b) 3 Enrollment Samples

(c) 4 Enrollment Samples (d) 5 Enrollment Samples

Fig. 5. Distribution of Genuine and Non-Genuine Biometric Keys: the inter-class and intra-class distributions of incorrect bits within cryptographic keys for the generation of 128-bit cryptographic keys for the CASIAv3-Interval iris database.

no user-specific encoding parameters have been used (all users use the same encoding). With respect to the evaluation of approaches to cancellable biometric this experimental setting is equal to the stolen-token scenario. As performance results we achieve a key generation rate of 86.23%, 90.17%, 92.21%, and 95.09% for two, three, four and five enrollment samples, respectively. In comparison to existing approaches to biometric key generation (e.g. [13], [12], [14])these are promising results. By analogy to generic biometric systems, performance rates increase the more enrollment samples are acquired.

4.1 Security Analysis

In comparison to existing iris-biometric cryptosystems our system offers significant advantages in terms of template protection. Most biometric cryptosystems based on iris biometrics bind arbitrary chosen keys with biometric data in so-called key binding schemes [3,9,11]. Though practical performance rates are achieved, these systems tend to generate low entropy templates. For example, the approach of Hao *et al.* [3] which, to our knowledge, reveals the best results

for binding and retrieving 140 bits was found to exhibit a key entropy of only 44 bits [8]. This is because error correction codes underly distinct structures and binary iriscodes are not distributed uniformly random. If cryptographic keys are compromised, templates may be decrypted and iriscodes could be stolen. This is a very critical issue regarding template protection.

In our system stored templates do not contain any data of original biometrics, neither in plain nor in encrypted form. This means, if an imposter gets to compromise the biometric key of a user, reconstructing the original biometric feature vector of this user is not feasible. This is due to the fact that stored bit-masks do not reveal information about feature values, only about locations of these. Furthermore, according fake intervals hide the genuine intervals used to generate correct keys. The detection of reliable components represents a non-invertible transform of the original biometric data, since parts of the iris texture are discarded. Addionally, we provide fully revocable cryptographic keys since intervals are encoded at random at the time of registration.

5 Conclusion

In this work we presented a new approach to iris-biometric key-generation. While the majority of biometric cryptosystems based on iris biometrics are implemented as key-binding systems our technique is based on the so-called interval-mapping concept. Regarding iris biometrics interval-mapping schemes represent an entirely different approach and existing systems do not provide practical key generation rates [14]. In this paper we propose an iris-biometric interval-mapping scheme in which we obtain a key generation rate of more than 95% which are promising results with respect to iris biometric key generation.

Since interval-mapping schemes do not require the storage of biometric data, neither in raw nor in encrypted form (as it is done in key-binding schemes), our system provides higher security in terms of template protection preserving the privacy of registered users. In case a potential imposter gains access to the stored helper data of a user it is impossible to reconstruct the complete original biometric template, unlike in conventional key-binding schemes (e.g. in [7]). Besides these significant advantages in terms of template protection the proposed scheme is easily adopted to provide fully revocable cryptographic keys.

References

1. Uludag, U., Pankanti, S., Prabhakar, S., Jain, A.K.: Biometric cryptosystems: issues and challenges. Proceedings of the IEEE 92(6), 948–960 (2004)
2. Bowyer, K., Hollingsworth, K., Flynn, P.: Image understanding for iris biometrics: a survey. Computer Vision and Image Understanding 110, 281–307 (2008)
3. Hao, F., Anderson, R., Daugman, J.: Combining Cryptography with Biometrics Effectively. IEEE Transactions on Computers 55(9), 1081–1088 (2006)
4. Wu, X., Qi, N., Wang, K., Zhang, D.: A Novel Cryptosystem based on Iris Key Generation. In: Fourth International Conference on Natural Computation (ICNC'08), pp. 53–56 (2008)

5. Bringer, J., Chabanne, H., Cohen, G., Kindarji, B., Zémor, G.: Theoretical and practical boundaries of binary secure sketches. IEEE Transactions on Information Forensics and Security 3, 673–683 (2008)
6. Davida, G., Frankel, Y., Matt, B.: On enabling secure applications through off-line biometric identification. In: Proc. of IEEE, Symp. on Security and Privacy, pp. 148–157 (1998)
7. Juels, A., Wattenberg, M.: A fuzzy commitment scheme. In: Sixth ACM Conference on Computer and Communications Security, pp. 28–36 (1999)
8. Buhan, I.R., Doumen, J.M., Hartel, P.H., Veldhuis, R.N.J.: Fuzzy extractors for continuous distributions. Technical report, University of Twente (2006)
9. Rathgeb, C., Uhl, A.: Systematic construction of iris-based fuzzy commitment schemes. In: Tistarelli, M., Nixon, M.S. (eds.) ICB 2009. LNCS, vol. 5558, pp. 947–956. Springer, Heidelberg (2009)
10. Rathgeb, C., Uhl, A.: Context-based texture analysis for secure revocable iris-biometric key generation. In: Proceedings of the 3rd International Conference on Imaging for Crime Detection and Prevention, ICDP '09, London, UK (2009)
11. Bringer, J., Chabanne, H., Cohen, G., Kindarji, B., Zémor, G.: Optimal iris fuzzy sketches. In: Proc. 1st IEEE International Conference on Biometrics: Theory, Applications, and Systems, pp. 1–6 (2007)
12. Vielhauer, C., Steinmetz, R., Mayerhöfer, A.: Biometric hash based on statistical features of online signatures. In: ICPR '02: Proceedings of the 16 th International Conference on Pattern Recognition (ICPR'02), vol. 1, pp. 100–123 (2002)
13. Sutcu, Y., Sencar, H.T., Memon, N.: A secure biometric authentication scheme based on robust hashing. In: MMSec '05: Proceedings of the 7th Workshop on Multimedia and Security, pp. 111–116 (2005)
14. Rathgeb, C., Uhl, A.: An iris-based interval-mapping scheme for biometric key generation. In: Proceedings of the 6th International Symposium on Image and Signal Processing and Analysis, ISPA '09, Salzburg, Austria (September 2009)
15. Daugman, J.: How Iris Recognition Works. IEEE Trans. CSVT 14(1), 21–30 (2004)
16. Ratha, N.K., Connell, J.H., Bolle, R.M.: Enhancing security and privacy in biometrics-based authentication systems. IBM Systems Journal 40, 614–634 (2001)

Generalizations and Extensions of Redactable Signatures with Applications to Electronic Healthcare

Daniel Slamanig[1] and Stefan Rass[2]

[1] Carinthia University of Applied Sciences, 9020 Klagenfurt Austria
d.slamanig@cuas.at
[2] Klagenfurt University, System Security Group, 9020 Klagenfurt Austria
stefan.rass@syssec.at

Abstract. Redactable signatures allow for altering signed documents, retaining the validity of the signature without interaction with the original signer. In their plain form, such schemes are designed for documents having an unspecific structure, i.e. documents are simply considered as binary strings. In this work, we generalize the concept of redactable signatures towards documents that inherently provide a structure and investigate the security of our construction. Furthermore, we present extensions to our scheme, adding features not commonly provided by other redactable signature schemes. Additionally, various applications in healthcare are discussed, supporting the applicability and usability of our construction.

1 Introduction

A standard digital signature links an identity to a document and thereby provides an authentication mechanism, as well as integrity assurance. Consequently, no alteration of a signed document is possible once the signature has been attached. However, there are scenarios where it would be valuable to have the possibility to replace or remove (specified) parts of a message after signature creation such that the original signature stays valid. Such signatures are called *redactable* (we remark that the naming is not consistent throughout the literature). An example application is the anonymization (removal of identifying information) of medical documents or general XML-documents.

Redactable signature schemes were independently introduced by *Steinfeld et al.* [1] and *Johnson et al.* [2] in 2001 and 2002 respectively. Loosely spoken, such signature schemes are constructed from standard signature schemes using the hash-then-sign paradigm by virtue of modifying the construction of the hash value. Merkle-trees [3] are a convenient tool for that matter: if a message is hashed via a Merkle-tree, then a message part at a leaf of that tree can be redacted by replacing it with an empty-symbol in the message, along with the hash-value of the (now missing) message part. For signature verification, the

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 201–213, 2010.

remaining hash-tree can be constructed as usual, thus creating a signature with redaction capabilities. One such scheme is described in section 4.

Contribution: The contributions of this paper are manifold: while standard redactable signatures are applicable to documents providing no specific structure, i.e. which are simply interpreted as binary strings, and organized as leafs of binary trees, we generalize the whole concept towards documents providing an inherent structure. We present efficient constructions with provable security and present various extensions, adding valuable features to the given construction.

2 Applications in Electronic Healthcare

Although there are several different applications of signature schemes supporting redaction, we will focus on their application in electronic healthcare (eHealth) here and thus briefly motivate the field and their potential benefits.

In recent years eHealth and ubiquitous healthcare have gained significant attention in the scientific community as well as among practitioners, since these concepts hold great potential to improve medical treatment processes. In order to guarantee an adequate quality of data, it is important that authenticity and integrity of involved data can be checked at any time. Hence, digital signatures seem to be an invaluable tool for that matter. Redacting parts of documents after signing while preserving the signature validity can be valuable, when information of a medical document needs to be removed (e.g. the document is anonymized). This spares the need to contact the original signer to re-create the signature and enables more time-efficient medical processes with guaranteed signature validity. More importantly, *source-authenticity* of the original data source is ensured.

Achieving k-Anonymity: Person related health data are in general very sensitive information and consequently must be protected appropriately. Especially, when using these data for secondary use, which includes medical research, clinical studies and second opinions, the protection of the privacy of patients is obligatory. Hence, it is necessary to prepare data such that the patients cannot be identified uniquely anymore, before passing it to another party for secondary use. This process of preparation focuses primarily on attributes that directly identify the patients and secondarily on attributes that indirectly identify the patients. Redactable signatures allow for removing certain identifying attributes in order to anonymize a document whilst retaining its authenticity. Suppose that a combination of non-identifying attributes uniquely identifies an individual (a study [4] from the year 2000 estimated that 63% of the population of the United States can be uniquely identified based only on gender, date of birth, and 5-digit zip code). Assume that upon removing a subset of those attributes (*suppression*), identification becomes ambiguous, meaning that these attributes can equally well be related to k different individuals. The same effect could be achieved by replacing the attribute values by more general ones (such as telling only

the first 2 digits of a ZIP-code for instance) This process is called *generalization* (see [5] for details). Either way, we get *k-anonymity* of the individual, as introduced in [6,7]. Removing or replacing attributes to achieve k-anonymity is a natural application of redactable signatures. We revisit this in section 6.

Privacy Preserving and Unbiased Second Opinions: Redacting certain parts of an authentic (signed) medical document may as well be desirable if second opinions for some medical problem are sought. Either a medical expert or the patient could call for anonymity of the patient (privacy), or could redact certain parts of the medical document to make sure that the other expert remains unbiased by otherwise visible information, e.g. the diagnosis, while having authentic access to relevant information, e.g. anamnesis, lab-results, etc.

3 Related Work

Signature Schemes: Sanitizable signatures [8,9] permit a designated party (*sanitizer*) to remove or replace desginated parts of a document. The signer can exactly mark parts as (not) redactable. However, the signer cannot specify how this redaction may look like, i.e. the redactor can replace a redactable part with an arbitrary string. The redaction capability is achieved by computing chameleon hashes (trapdoor commitments) for all the parts of a message that can be redacted. Only the sanitizer, in possession of the trapdoor information, is able to compute collisions for the chameleon hashes and hence can change the respective parts of the message arbitrarily. Nevertheless, for a verifier it is not noticeable whether the original message has been modified. The concept of sanitizable signatures was further extended to trapdoor sanitizable signatures in [10], which allow to give the power of sanitization to possibly several entities. Recently, there have also been proposed some extensions that restrict the choice of replaced strings [11], which seems to be very important and is also discussed in this paper in the context of redactable signatures.

In content extraction signatures [1], which, besides other constructions, use a hash-tree based scheme similar to redactable signatures, everybody can remove substrings. Furthermore, the parts that can be removed can be specified by means of a policy named content extraction access structure. This policy, which specifies the set of indices of redactable document parts, is signed along with the message and needs to be evaluated by the signature verification algorithm. If any redaction has taken place that is not specified in the policy, the signature validation algorithm rejects the signature.

A straightforward approach to redactable signatures is by splitting the document into pieces, each of which is signed separately. This makes removing a part simple but computationally expensive and not very flexible. An efficient construction is possible by using the pairing-based signature of [12], which saves bandwidth by aggregating all single signatures into one signature. However, all signatures corresponding to message parts that can be redacted need to be available to the redactor in addition. Using this construction, it is possible to remove blocks from the document and to remove the corresponding signatures from

the aggregated signature without the verifier being able to realize whether the message has been redacted or not (invisible redaction) [13]. This concept was recently extended to also provide non-invisible redaction [14], which can also be achieved for the aforementioned construction by including indexes into the single message parts. Another scheme, based on aggregating single signatures using batch RSA signatures, was introduced in [1] (prior to the aforementioned schemes). Although, their scheme also requires a number of RSA signatures that is linear in the number of message blocks, once a redaction has taken place, solely a single RSA signature (as the product of single signatures) needs to be transmitted to the verifier.

Furthermore, [15] have recently extended the approach of [2], such that a signer can specify which parts of a document are allowed to be redacted. Their approach is similar to the one of [1]. Another approach by [16] extends the tree-based construction of [2] such that the length of the redacted strings can be hidden. Finally, [17] have proposed a redactable signature scheme that does not rely on random oracles and is provably secure in the standard model.

Medical Applications: *Ateniese et al.* [9] mention alongside that sanitizable signatures can accommodate different level of data de-identification, supporting the "minimum necessary" disclosure standard of HIPAA Privacy Rule, but do not consider further details. The authors of [18] have recently proposed a concept to apply redactable signatures in context of Personal Health Records (PHRs) to realize minimum disclosure of personal information and to achieve cryptographic enforcement of dependencies between parts containing personal information. Another recent work developed independently from ours by *Haber et al.* [15] can be considered as closest to the ideas presented in this paper. They present redactable signatures that can be used for pseudonymization and de-identification of data, although they mainly focus on relational, i.e. 2-dimensional, data tables. In contrast, our work focuses on applying the aforementioned and additional mechanisms on structured documents.

4 Redactable Signatures

As mentioned in the introduction, redactable signatures permit removing parts of a signed document whilst retaining the validity of the signature. Technically, this amounts to an efficient update of the signature that requires no interaction with the original signer. Our proposed scheme will build upon the scheme of *Johnson et al.* [2], which we will briefly describe. We assume the reader to be familiar with the concept of a (binary) Merkle-tree, as this is used extensively in the following. The second core ingredient for our construction are pseudorandom generators (PRGs) based on GGM-trees [19].

Johnson-Molnar-Song-Wagner Redactable Signature Scheme: To construct a redactable signature for a document D, the authors of [2] divide the document into $N = \left\lceil \frac{|D|}{n} \right\rceil$ blocks of size n, so that $D = D[1], \ldots, D[N]$. Next,

they build a complete binary tree with N leafs, having the message blocks assigned to them in the correct order. Consequently, one is able to build the hash of the root node h_R of the tree by hashing up the tree from the bottom (i.e. simply construct a Merkle-tree) and sign h_R by using a conventional signature scheme, i.e. set $\sigma = S_{SK}(h_R)$.

To remove the i-th block, which can be seen as a replacement with the empty symbol $\perp$, one may pass the document $D' = D[1], \ldots, D[i-1], \perp, D[i+1], \ldots, D[N]$ to a verifier, along with the hash value $h_{D[i]} = H(D[i])$ of the removed message block (leaf) $D[i]$. The missing message block causes no problem in reconstructing the Merkle-tree, because its hash-value is still available. The updated signature is simply $\sigma' = (\sigma, h_{D[i]})$, where σ is the original signature. One problem instantly arises if the block size is too small to prevent brute-force determination of the missing part $D[i]$. Therefore, a GGM-tree with its shape identical to the Merkle-tree is used to generate randomizers that are attached to each intermediate value (or message block) prior to hashing it. Technically, one generates a randomizer r_{v_i} for every node v_i of the tree by means of a GGM-tree, starting with a randomly chosen seed r_{v_R} at the root node v_R. Instead of solely hashing the values (message parts denoted as x), as within the Merkle-tree, the subsequent rules (1) are applied.

$$\phi(p) = \begin{cases} H(\phi(c_l)||\phi(c_r)) & \text{if } p \text{ has two children,} \\ H(\phi(c_l)) & \text{if } p \text{ has one child,} \\ H(r_p||x) & \text{if } p \text{ is a leaf.} \end{cases} \quad (1)$$

Intuitively, the values of the leafs are hashed along with a randomizer, which allows to hide the values of removed parts of the document. Technically, this can also be seen as a commitment to a message part, such that the commitment hides the message part as long as the randomizer is unknown, i.e. the hiding of redacted message parts is based on the hiding property of the used commitment scheme. It must be mentioned, that one may also use other commitment schemes. However, commitments based on hash functions [20] are usually much more efficient.

In order to be able to build the Merkle-tree to verify the signature for document D, besides the standard signature, the updated signature needs to additionally include the seed of the GGM-tree, i.e. $\sigma_{RS} = (\sigma, r_{v_R})$. The signature verification is straightforward by constructing the GGM-tree by means of the seed r_{v_R}, building the Merkle-tree from the document and verifying the standard signature σ of the hash of the root node. In order to redact block i from D, one deletes the block from D as already mentioned above and updates the signature to be $\sigma'_{RS} = (\sigma, h_{D[i]}, \mathcal{A}_{v_{D[i]}})$ and additionally includes the indices of the redacted parts (which is omitted here). Thereby, the signature contains the randomized hash of the leaf, which does not reveal any information on the removed part and the authentication path $\mathcal{A}_{v_{D[i]}}$ that contains the set of randomizers, which is necessary to reconstruct the remaining part of the GGM-tree. But, the computation of the randomizer $r_{v_{D[i]}}$ for the removed part $D[i]$ is impossible, as long as the used PRG is secure. For notational convenience, we will

subsequently denote the set of hash values (commitments) that result from redacted document parts as C and the set of randomizers (in the respective authentication paths) as R. In the example discussed before, our updated signature will be $\sigma'_{RS} = (\sigma, C, R)$, where $C = \{h_{D[i]}\}$ and $R = \mathcal{A}_{v_{D[i]}}$.

5 Generalized Redactable Signatures

Redactable signatures [2] organize the content of an (unstructured) document as leafs of a complete binary tree. For XML documents, however, one can beneficially (cf. section 6) exploit the XML-induced N-ary tree structure to avoid cumbersome assignments to a binary tree *not* resembling the natural structure of the document. For instance, one can construct subtrees from the respective XML-attributes and values.

5.1 Construction

Generalized redactable signatures are obtained by replacing the binary tree construction with a general tree construction. While a Merkle-tree can be defined using general trees rather than binary trees (see equation (2)), the generalization is less obvious for the GGM-tree. This one, in particular the proof of security regarding the outputs, hinges on the binary tree construction.

Our construction that enjoys provable security (since the security is implied by the standard GGM-tree construction) can be described as follows: Suppose we have a tree T, and let v denote an inner node with label r_v and $N > 2$ children. For every child c, we seek to derive a randomizer r_c (to be used in equation (2)) from v's label r_v, such that the children's labels (randomizer) do not provide any information about each other. For the leaves of a (binary) GGM-tree, this has been proven (see [19]). The idea is to simply construct an auxiliary GGM-tree with root v, and (at least) N leafs resembling the children of v. The so-constructed randomizers r for the children of v will enjoy the desired properties, because the leafs of GGM-trees do. Such an auxiliary GGM-tree can be constructed for each inner node of the (general) tree T. The arising additional costs can be considered negligible, since many standard GGM-tree implementations are extremely efficient.

Taking the construction in section 4 as a template, we will devise our scheme for generalized redactable signatures on arbitrary documents D being represented as an N-ary tree. Thereby, we assume that we have already applied a transformation to a document D and for simplicity we will denote a document and its parts as $D = D[1], \ldots, D[n]$, whereas we assume that this is a representation of a N-ary tree. The assignment ϕ for the labels of the nodes of the modified Merkle-tree is adapted to (2), with randomizers r_p created using the generalized GGM-tree construction outlined above. Note, that now also inner nodes hold parts of messages (denoted as x). Note that $C(m, r)$ represents a commitment to message m using randomizer r, e.g. using hash functions [20].

$$\phi(p) = \begin{cases} H(x||\phi(c_0)||\ldots||\phi(c_k)) & \text{if } p = \text{root and has } k{+}1 \text{ children,} \\ C((x, \phi(c_0), \ldots, \phi(c_k)), r_p) & \text{if } p \neq \text{root and has } k{+}1 \text{ children,} \\ C(x, r_p) & \text{if } p \text{ is a leaf.} \end{cases} \quad (2)$$

Basically, the computation of the commitments of leave nodes is identical to the Merkle-tree construction and the remaining computations are adapted to N-ary trees. A generalized redactable signature consists of the signed root value $\phi(p)$ where p is root and the seed r_{v_R} of the root node. More precisely, the signature $\sigma' = (\sigma, C, R)$, where σ is the signature for $\phi(p)$, $C = \emptyset$ and $R = \{r_{v_R}\}$. When redacting a message part, we preceded as in section 4 and if the redacted node is an inner node, the randomizers of all child nodes need to be included.

5.2 Security of Generalized Redactable Signatures

Essentially, the security of generalized redactable signatures has to satisfy two properties, namely *unforgeability* and *hiding*. Hiding refers to the adversary's inability to reconstruct redacted parts of the document. Unforgeability is more involved and requires some definitions [2,15]: given a document $D = D[1], \ldots, D[n]$ composed of n blocks, a redacted document is $D' = D'[1], \ldots, D'[n]$ with $D'[i] \in \{D[i], \perp\}$ for all $i = 1, 2, \ldots, n$. The more blocks are redacted, the less related the document becomes to its original parent. This is formally modeled through a partial order relation:

Definition 1. *Let $D = D[1], \ldots, D[n]$ and $D' = D'[1], \ldots, D'[n]$ be two (possibly) redacted versions of a document $\mathbf{D}$. We define a partial order $\prec$ on redacted documents such that $D \prec D'$ holds if and only if the following properties are satisfied for all $1 \leq i \leq n$:*

1. *if $D[i] \neq \perp$, then $D'[i] \neq \perp$.*
2. *if $D[i] = \perp$, then either $D'[i] = \mathbf{D}[i]$ or $D'[i] = \perp$.*

Roughly speaking, unforgeability prevents the adversary from finding a signature for a document D such that $D' \prec D$ for a given and signed document D'. In contrast to security definitions of standard signature schemes [21], we are not focusing on existential unforgeability against chosen message attacks (eUF-CMA). Recall that in the game based proof for eUF-CMA security, an adversary can issue signature queries for adaptively chosen messages of his choice, and finally needs to output a valid signature for a message that has not been issued to the sign oracle during the game. But we will base the security on a game capturing the aforementioned unforgeability and hiding properties at the same time and furthermore relate the signature forgery to a redacted document of the adversary's choice. Nevertheless, is should be noted that we require the standard signature scheme used to sign the root of the modified Merkle-tree to provide eUF-CMA security. Thereby, our game is based on the security model of [15]. Basically, an adversary who manages to win this game with non-negligible

probability, will be able to breach the security of some of the used cryptographic primitives with non-negligible probability. Our definition of secure generalized redactable signature schemes follows the same ideas and is as well based on a game in the aforementioned sense.

Definition 2. *A generalized redactable signature scheme is said to be secure, if no probabilistic polynomial-time adversary has a non-negligible advantage against the challenger in game 5.2.*

Game 5.2. The following oracles are available to the adversary:

1. commit: A commit query is defined as issuing a document D to the oracle, which chooses a random seed r_{v_R}, derives randomizers for all nodes of the N-ary tree (assume that the tree has n nodes), computes the commitments h_{v_i}, $1 \leq i \leq n$, for all nodes and returns the tuple $(r_1, \ldots, r_n, h_{v_1}, \ldots, h_{v_n})$.
2. sign: A sign query is defined as issuing the hash value of the root node h_{v_R} to the oracle, which uses the private-key SK of a eUF-CMA secure digital signature scheme, computes a signature $\sigma = S_{SK}(h_{v_R})$ and returns σ.
3. redact: A redact query is defined as issuing a generalized redactable signature σ_{GRS}, a document D and a set of indices (which defines the document parts to be redacted) to the oracle, which computes the redaction and returns the redacted document D' along with an adapted generalized redactable signature σ'_{GRS}.

The adversary plays the following game for breaking the signature scheme:
Setup: The challenger takes a security parameter, chooses a secure digital signature scheme, generates a key-pair (SK, PK) for this scheme, chooses a cryptographic hash function H as well as a commitment scheme C for the tree computation and chooses a PRG G to generate the randomizers, gives (PK, H, C, G) and the signature scheme description to the adversary and keeps SK secret.
Phase 1: The adversary issues queries $q_1, \ldots, q_m$, where $q_i \in \{\mathsf{commit}, \mathsf{sign}, \mathsf{redact}\}$ and these queries may be asked adaptively, that is, each query q_i may depend on the replies to $q_1, \ldots, q_{i-1}$.
Challenge: The adversary may want to attack the hiding property (figure out redacted parts of the document) by providing two documents D_0 and D_1 of equal length along with a list of parts to be redacted to the challenger, which differ in exactly one document part, i.e. for exactly one i it holds that $D_0[i] \neq D_1[i]$. The challenger chooses a random bit b, produces a generalized redactable signature for D_b and redacts the document (the i'th document part must be redacted).
Phase 2: The adversary issues additional queries $q_{m+1}, \ldots, q_n$ where query $q_i \in \{\mathsf{commit}, \mathsf{sign}, \mathsf{redact}\}$ and these queries may be asked adaptively. Thereby, the adversary is not allowed to ask any sign or redact queries for any $D_b \prec D$.
Guess: The adversary outputs a guess for either attacking the hiding or the unforgeability property.
Hiding: The adversary outputs a bit b' and wins the game if $b = b'$ holds. The adversary's advantage is defined as $|\mathsf{Pr}[b = b'] - \frac{1}{2}|$.

Unforgeability: The adversary outputs a generalized redactable signature $\sigma' = (\sigma, C, R)$ for a document D, whereas no sign or redact query has been issued to any document D' such that $D \prec D'$ or $D_b \prec D'$. The adversary's advantage is defined as the probability that the verification of the signature succeeds when the two aforementioned conditions hold.

Theorem 1 (Proof Appears in Appendix A). *If the signature schemes provides eUF-CMA security, H is a collision-resistant cryptographic hash function, C a secure commitment scheme and G a secure PRG, then the generalized redactable signature scheme is a secure one.*

6 Extensions to Generalized Redactable Signatures

In this section we present extensions to realize *controlled replacement* and *fine-grained redaction*. Redactable signatures in the sense above allow for "deleting" parts of the document by replacing them with the empty symbol $\perp$. However, it might be interesting for the signer to specify a set of permitted replacements for certain parts of the document. An authentic (signed) questionnaire would be an example of such a scenario. Sanitizable signatures permit replacements of *designated* parts by *designated* redactors, who are free to substitute any string they like. Recently, [11] introduced *controlled* replacements. These limit the redactors substitution options to a set S that can be specified by the verifier. The idea is to replace the empty symbol $\perp$ by an element $x \in S$ along with a compact representation of the set S of permissible substitutions. Bloom-Filters [22] or accumulators [23] are appropriate for that matter. Testing for a given substitution is easy by querying the Bloom-Filter or the accumulator (notice that a small probability for getting a false-positive answer is inevitable in case of Bloom-filters). Both approaches are deterministic and as such allow for brute-force identification of the set S, if it is sufficiently small.

We sketch two remedies for this drawback: *randomized accumulators* and *generalization hierarchies*. The secure RSA-based accumulator, specified in [16], employs hash-functions. Replacing those hash-values by message-authentication codes with secret random keys, creates a *randomized accumulator* from a standard (deterministic) accumulator. Simply replace the hash-function $H(s)$ by a randomized commitment for s based on hash functions, i.e. by $H(s, r_s)$, where r_s is a randomizer that is *specific* for s. Providing this randomizer in addition permits querying whether s is contained in the accumulator, but prevents brute-force searching for other elements s', because their specific randomizers (keys) $r_{s'}$ remain unknown. A randomizer r_s for s can be derived using GGM-trees.

Generalization hierarchies are particularly suited for k-anonymity and generalization sets. As an example, consider an attribute ZIP= 22047 and its associated generalization set $S = \{22047, 2204*, 220**, 22***\}$. The ZIP-code shall only be replacable by elements of S. Simply change this block into the root of a subtree with leafs being the members of S. For the example, the child nodes would have labels 22, 0, 4 and 7. Redactions work in the obvious way.

7 Conclusion

In this paper we have presented a generalization of redactable signature schemes and several extensions. In contrast to standard redactable signatures, our scheme takes advantage of the inherent structure of structured documents like XML-documents. This is on the one hand a more natural way to access this problem and on the other hand beneficial for extensions like generalization hierarchies. Furthermore, the generalization induces only little additional computational overhead, which we attempt to confirm upon implementations. We believe that this concept is a valuable tool in the context of electronic healthcare, since it combines redaction capability with verification of the authenticity of remaining data while improving the efficiency of medical treatment processes. Future research includes further development of the proposed extensions, along with respective implementations (including an analysis of how much the XML structure affects the scheme's efficiency). A performance study to evaluate the quantitative benefit in terms of storage space and computational effort is part of ongoing work.

References

1. Steinfeld, R., Bull, L., Zheng, Y.: Content Extraction Signatures. In: Kim, K.-c. (ed.) ICISC 2001. LNCS, vol. 2288, pp. 285–304. Springer, Heidelberg (2002)
2. Johnson, R., Molnar, D., Song, D., Wagner, D.: Homomorphic Signature Schemes. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 244–262. Springer, Heidelberg (2002)
3. Merkle, R.C.: A Certified Digital Signature. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 218–238. Springer, Heidelberg (1990)
4. Golle, P.: Revisiting the Uniqueness of Simple Demographics in the US Population. In: WPES 2006, pp. 77–80. ACM, New York (2006)
5. Ciriani, V., di Vimercati, S.D.C., Foresti, S., Samarati, P.: Theory of Privacy and Anonymity. In: Algorithms and Theory of Computation Handbook. CRC Press, Boca Raton (2009)
6. Samarati, P.: Protecting Respondents' Identities in Microdata Release. IEEE Trans. Knowl. Data Eng. 13(6), 1010–1027 (2001)
7. Sweeney, L.: k-Anonymity: a Model for Protecting Privacy. Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 10(5), 557–570 (2002)
8. Brzuska, C., Fischlin, M., Freudenreich, T., Lehmann, A., Page, M., Schelbert, J., Schroeder, D., Volk, F.: Security of Sanitizable Signatures Revisited. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443. Springer, Heidelberg (2009)
9. Ateniese, G., Chou, D.H., de Medeiros, B., Tsudik, G.: Sanitizable signatures. In: di Vimercati, S.d.C., Syverson, P.F., Gollmann, D. (eds.) ESORICS 2005. LNCS, vol. 3679, pp. 159–177. Springer, Heidelberg (2005)
10. Canard, S., Laguillaumie, F., Milhau, M.: Trapdoor Sanitizable Signatures and Their Application to Content Protection. In: Bellovin, S.M., Gennaro, R., Keromytis, A.D., Yung, M. (eds.) ACNS 2008. LNCS, vol. 5037, pp. 258–276. Springer, Heidelberg (2008)
11. Klonowski, M., Lauks, A.: Extended Sanitizable Signatures. In: Rhee, M.S., Lee, B. (eds.) ICISC 2006. LNCS, vol. 4296, pp. 343–355. Springer, Heidelberg (2006)

12. Boneh, D., Gentry, C., Lynn, B., Shacham, H.: Aggregate and Verifiably Encrypted Signatures from Bilinear Maps. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 416–432. Springer, Heidelberg (2003)
13. Miyazaki, K., Hanaoka, G., Imai, H.: Digitally Signed Document Sanitizing Scheme Based on Bilinear Maps. In: Proc. of the 2006 ACM Symp. on Information, Computer and Communications Security, ASIACCS 2006, pp. 343–354. ACM, New York (2006)
14. Izu, T., Kunihiro, N., Ohta, K., Sano, M., Takenaka, M.: Yet Another Sanitizable Signature from Bilinear Maps. In: ARES 2009, pp. 941–946. IEEE Computer Society, Los Alamitos (2009)
15. Haber, S., Hatano, Y., Honda, Y., Horne, W., Miyazaki, K., Sander, T., Tezoku, S., Yao, D.: Efficient Signature Schemes Supporting Redaction, Pseudonymization, and Data Deidentification. In: Proc. of the 2008 ACM Symp. on Information, Computer and Communications Security, ASIACCS 2008, pp. 353–362. ACM, New York (2008)
16. Chang, E.C., Lim, C., Xu, J.: Short Redactable Signatures Using Random Trees. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 133–147. Springer, Heidelberg (2009)
17. Nojima, R., Tamura, J., Kadobayashi, Y., Kikuchi, H.: A Storage Efficient Redactable Signature in the Standard Model. In: Samarati, P., Yung, M., Martinelli, F., Ardagna, C.A. (eds.) ISC 2009. LNCS, vol. 5735, pp. 326–337. Springer, Heidelberg (2009)
18. Bauer, D., Blough, D., Mohan, A.: Redactable Signatures on Data with Dependencies and their Application to Personal Health Records. In: Proc. of the 8th ACM Workshop on Privacy in the Electronic Society, WPES '09, pp. 91–100. ACM Press, New York (2009)
19. Goldreich, O., Goldwasser, S., Micali, S.: How to Construct Random Functions. J. ACM 33(4), 792–807 (1986)
20. Halevi, S., Micali, S.: Practical and Provably-Secure Commitment Schemes from Collision-Free Hashing. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 201–215. Springer, Heidelberg (1996)
21. Goldwasser, S., Micali, S., Rivest, R.: A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. SIAM J. on Computing 17(2), 281–308 (1988)
22. Bloom, B.: Space/Time Trade-offs in Hash Coding with Allowable Errors. Commun. ACM 13(7), 422–426 (1970)
23. Benaloh, J., de Mare, M.: One-Way Accumulators: A Decentralized Alternative to Digital Sinatures (Extended Abstract). In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 274–285. Springer, Heidelberg (1994)

A Proof of Theorem 1

For the proof of Theorem 1 we use collision-resistant hash functions H, a secure commitment scheme C, a secure PRF G and a $\mathsf{eUF\text{-}CMA}$ secure digital signature scheme. We omit the formal definitions here, but note that for collision-resistant hash functions the probability of finding two messages $m \neq m'$ with $H(m) = H(m')$ is negligible and, for a secure commitment scheme only with negligible probability, one will be able to figure out m from $C(m, r)$ and to open $C(m, r)$ to some $m' \neq m$.

Basically, the proof follows the reduction to contradiction paradigm, i.e. we construct an adversary $\mathcal{B}$ that uses the adversary in game 5.2 (called $\mathcal{A}$) whereas $\mathcal{B}$ plays the role of the challenger of $\mathcal{A}$ (using $\mathcal{A}$'s output as his own) and can break one of the used cryptographic primitives with non-negligible advantage.

Proof (sketch): Subsequently, we sketch how $\mathcal{B}$ acts when using adversary $\mathcal{A}$:
Setup: $\mathcal{B}$'s challenger runs the setup, gives (PK, H, C, G) as well as the description of the signature scheme to $\mathcal{B}$. $\mathcal{B}$ gives all the information to $\mathcal{A}$ (note that $\mathcal{A}$ and $\mathcal{B}$ do not know SK).
Phase 1: $\mathcal{A}$ is run and issues queries $q_1, \ldots, q_m$ as in game 5.2, but issues his queries $q_i \in \{\mathsf{commit}, \mathsf{sign}, \mathsf{redact}\}$ to $\mathcal{B}$. Except for queries of type sign, $\mathcal{B}$ acts as defined in section 5.2. However, since $\mathcal{B}$ does not know SK he forwards these types of queries to his challenger.
Challenge: $\mathcal{A}$ may want to attack the hiding property and outputs two equal length documents D_0 and D_1, which differ in exactly one document part, i.e. for exactly one i it holds that $D_0[i] \neq D_1[i]$, along with a list of parts to be redacted. Now, $\mathcal{B}$ will use the advantage of $\mathcal{A}$'s guess (in the guess phase) to break the hiding property of the commitment scheme C. He proceeds as follows: He takes D_0 and D_1 as the two messages for breaking the hiding property of C and gives them along with i to his challenger. The challenger chooses at random a bit b and computes a commitment $h_{D_b[i]}$ for $D_b[i]$ (note that if $D_b[i]$ is an inner node, the challenger needs to compute the hash of the root of this subtree by means of a commit query). $\mathcal{B}$ uses $h_{D_b[i]}$ and computes the Merkle-tree using the remaining document (note that the remaining parts of D_0 and D_1 are identical). Now $\mathcal{B}$ has embedded his challenge (to attack C) into $\mathcal{A}$'s challenge and issues a sign query on the root hash to his challenger. Hence, he obtains a generalized redactable signature, redacts at least the i'th subdocument and gives the result back to $\mathcal{A}$.
Phase 2: $\mathcal{A}$ issues queries $q_{m+1}, \ldots, q_n$ as in phase 1 to $\mathcal{B}$ (with the same restrictions as defined in section 5.2).
Guess: $\mathcal{A}$ outputs a guess for either attacking the hiding or the unforgeability property and $\mathcal{B}$ uses this guess for breaking one of the primitives (signature scheme, hash function, commitment scheme, PRG).
Hiding: Adversary $\mathcal{A}$ outputs a guess b which is used by $\mathcal{B}$. Clearly, if $\mathcal{A}$ wins, then $\mathcal{B}$ is able to break the hiding property of the commitment scheme C. Consequently, if $\mathcal{A}$'s advantage in winning is non-negligible, $\mathcal{B}$ will break the commitment scheme with non-negligible probability, which contradicts our assumption.
Unforgeability: Adversary $\mathcal{A}$ outputs a generalized redactable signature (as defined in game 5.2). Now we need to investigate all possible cases: Let us assume that this signature is not equal to any of the signatures returned by $\mathcal{A}$ during his game, i.e. $\mathcal{A}$ has broken the unforgeability property. But, now $\mathcal{B}$ could construct the Merkle-tree for the corresponding document and the root hash will be $h_{v_R} = H(x||\phi(c_0)||\ldots||\phi(c_k))$ which is a hash of a possible label x of the root node and a concatenation of the values of the $k+1$ children (as commitments to their subtrees). But the signature constitutes a signature of exactly these values, which can be used by $\mathcal{B}$ to break the existential unforgeability property of the

respective signature scheme. The remaining case is a signature that is identical to a signature which was output by $\mathcal{A}$ during one of the phases. Firstly, either the two Merkle-trees constructed for the respective documents differ in their roots, then $\mathcal{B}$ has found a signature forgery for the digital signature scheme. Otherwise, the two trees differ anywhere below the root node and $\mathcal{B}$ has found a collision for the hash function H. Secondly, $\mathcal{B}$ investigates the construction of auxiliary GGM-tree (note that the security of this construction is implied by that of the standard GGM-tree). If there exists an index j where both documents differ, then he has found two document parts which can be used to break the hiding property of C. Finally, we need to look at the restriction, whereas we denote the two documents as D and D'. We have required that $D \prec D'$ needs to hold. But if $\mathcal{A}$ outputs as a guess a signature for a document D that contains some position that has been redacted in D' (in one of the two phases) and is present in D, then $\mathcal{B}$ can use this to break the hiding property of C or the PRG G underlying the auxiliary GGM-tree construction. The proof is complete.

Chosen-Ciphertext Secure Certificateless Proxy Re-Encryption

Chul Sur[1], Chae Duk Jung[2], Youngho Park[2], and Kyung Hyune Rhee[3]

[1] Department of Computer Science, Pukyong National University,
599-1, Daeyeon3-Dong, Nam-Gu, Busan 608-737, Republic of Korea
kahlil@pknu.ac.kr
[2] Department of Information Security, Pukyong National University
{jcd0205,pyhoya}@pknu.ac.kr
[3] Department of IT Convergence and Application Engineering,
Pukyong National University
khrhee@pknu.ac.kr

Abstract. In this paper we introduce the notion of certificateless proxy re-encryption and also give precise definitions for secure certificateless proxy re-encryption schemes. We present a concrete scheme based on bilinear pairing, which enjoys the advantages of certificateless public key cryptography while providing the functionalities of proxy re-encryption. Moreover, the proposed scheme is unidirectional and compatible with current certificateless encryption deployments. Finally, we show that our scheme has chosen ciphertext security in the random oracle model.

Keywords: Certificateless Public Key Encryption, Proxy Re-Encryption, Chosen Ciphertext Security, Bilinear Pairing.

1 Introduction

In a proxy re-encryption (PRE) scheme, a semi-trusted proxy is allowed to transform a ciphertext under Alice's public key into a new ciphertext under Bob's public key on the same message. However, the proxy cannot learn any information about the messages encrypted under the public key of either Alice or Bob. PRE has received plenty of attention from the research community as fundamental cryptographic function to solve key management problems in various practical applications such as distributed file systems, secure email forwarding, and interoperable DRM systems in recent years [3,4,10].

In [7], Blaze *et al.* introduced the notion of PRE and proposed a concrete PRE scheme, where the message and secret keys are kept hidden from the proxy. However, their scheme is bidirectional. That is, the information released to divert ciphertexts from Alice to Bob can also be used to transform ciphertexts in the opposite direction [3,10]. Note that, it is obvious that unidirectional PRE is more powerful than bidirectional one since any unidirectional scheme can be easily transformed to a bidirectional one. Moreover, their scheme suffers from collusion attack that Alice (Bob) can collude with the proxy to reveal Bob's (Alice's) secret

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 214–232, 2010.

key. Ateniese *et al.* [3] presented the first constructions of unidirectional proxy re-encryption based on bilinear pairing. The schemes prevent the proxy from colluding with Bob to expose Alice's secret key. However, their schemes only achieve chosen plaintext attack (CPA) security.

As pointed out in [10], chosen plaintext security is clearly not enough for many applications that require security against chosen ciphertext attack (CCA). Canetti and Hohenberger [10] addressed the problem of obtaining a PRE scheme that is secure against chosen ciphertext attack. They then provided a formal security model for secure PRE schemes and presented a construction that has chosen ciphertext security in the standard model. However, their construction is still bidirectional and vulnerable to collusion attack like previous bidirectional PRE schemes. Recently, Libert and Vergnaud [15] presented the first construction of unidirectional proxy re-encryption that has chosen ciphertext security in the standard model. In [13], Green and Ateniese addressed the problem of identity-based PRE and presented a unidirectional scheme that has chosen ciphertext security in the random oracle model [6]. However, the scheme suffers from the collusion attack owing to the key sharing technique employed in the scheme. That is, Bob can collude with the proxy to reveal Alice's secret key.

Even though a number of PRE schemes have been proposed in the literature, all prior PRE schemes are constructed based on either traditional public key encryption (PKE) [3,7,10,15,16] or identity-based encryption (IBE) [13]. However, it is well recognized that traditional PKE suffers from the issues associated with certificate management such as revocation and IBE has inherent key escrow problem. To alleviate the aforementioned problems in traditional PKE and IBE, the concept of certificateless public key cryptography (CL-PKC), which combines the best aspects of traditional PKE (i.e., key escrow free) and of IBE (i.e., implicit certification), was introduced in [1]. The topic of CL-PKC has undergone quite rapid development with many schemes being proposed for encryption, signature, authenticated key agreement, and so on.

Our Contribution. In this paper, we introduce the notion of certificateless proxy re-encryption that enjoys the best aspects of traditional PKE and of IBE while providing the functionalities of proxy re-encryption. Upon taking into consideration of both security notions of certificateless encryption and proxy re-encryption, we provide a formal security model for constructing secure certificateless proxy re-encryption schemes and present a concrete scheme based on bilinear pairing. The proposed scheme is unidirectional and compatible with existing certificateless encryption deployments. Finally, we show that our scheme is secure against adaptive chosen ciphertext attack in the random oracle model.

The rest of the paper is organized as follows. The next section gives precise definition and security model for certificateless proxy re-encryption schemes. In Section 3, we present the first construction of certificateless proxy re-encryption based on bilinear pairing and then prove that the proposed scheme has chosen ciphertext security in the random oracle model under reasonable complexity assumption. Finally, we conclude the paper in Section 4.

2 Certificateless Proxy Re-Encryption

In this section, we present a definition of a certificateless proxy re-encryption (CL-PRE) scheme and provide a formal security model so as to construct secure CL-PRE schemes.

2.1 Definition

We incorporate certificateless encryption (CLE) [1,2] into proxy re-encryption (PRE) [10,13] for constructing a new certificateless proxy re-encryption scheme. A model for certificateless proxy re-encryption is specified as follows:

Definition 1 (CL-PRE). *A unidirectional certificateless proxy re-encryption scheme is a 9-tuple of algorithms which are the following*:

- **Setup(k):** Takes a security parameter k and returns a randomly chosen master key mk and a list of public parameters params.
- **Partial-Private-Key-Extract(params,mk,ID_A):** Takes a list of public parameters params, an identifier for user A, $\mathsf{ID}_A \in \{0,1\}^*$ and the master key mk as inputs, returns a partial private key d_A.
- **Set-Secret-Value(params,ID_A):** Given a list of public parameters params and an identifier for user A, returns a randomly chosen secret value x_A for that user.
- **Set-Private-Key(params,d_A,x_A):** Given a list of public parameters params, a user's partial private key d_A and secret value x_A, outputs a private key sk_A.
- **Set-Public-Key(params,x_A):** Takes a list of public parameters params and a user's secret value x_A as inputs, outputs a corresponding public key pk_A.
- **Encrypt(m,params,ID_A,pk_A):** Takes a message m, a list of public parameters params, an identifier ID_A of the receiver, and a corresponding public key pk_A as inputs. It outputs a ciphertext C_A or a distinguished symbol $\perp$.
- **Set-Proxy-Re-Encryption-Key(params,ID_A,pk_A,sk_A,ID_B,pk_B):** Takes a list of public parameters params, a user A's identifier ID_A and a public/private key pair (pk_A, sk_A), a user B's identifier ID_B and a public key pk_B as inputs. It outputs a unidirectional re-encryption key $rk_{A\to B}$.
- **Re-Encrypt(params,$rk_{A\to B}$,C_A):** Takes a list of public parameters params, a re-encryption key $rk_{A\to B}$, and a ciphertext C_A as inputs. It outputs a re-encrypted ciphertext C_B or a distinguished symbol $\perp$.
- **Decrypt(params,sk_{ID},C_{ID}):** Given a list of public parameters params, a ciphertext C_{ID}, and the receiver ID's private key sk_{ID}, it outputs a message m or a distinguished symbol $\perp$.

For completeness, it is obviously required that the following two propositions must hold for any message m in the message space $\mathcal{M}$:

- **Decrypt(params,sk_A,Encrypt(m, params, ID_A, pk_A))** $= m$.
- **Decrypt(params,sk_B, Re-Encrypt(params,$rk_{A\to B}$,C_A))** $= m$.

where $sk_{ID} \leftarrow$ **Set-Private-Key(params,d_{ID},x_{ID})** and $rk_{A\to B} \leftarrow$ **Set-Proxy-Re-Encryption-Key(params,ID_A,pk_A,sk_A,ID_B,pk_B)**.

2.2 Security Model

Here we provide a security model to prove the adaptive chosen ciphertext security of our CL-PRE scheme. To give a precise security notion for the proposed scheme, we take into account both security notions of certificateless encryption [1,2] and proxy re-encryption [10,16]. Following the security model in [1,2], there are two kinds of adversaries, named Type I and Type II adversaries, that represent a malicious third party and an honest-but-curious key generation center (KGC) in our security model, respectively. In addition, we consider the notion of derivative ciphertexts [10,16] to prevent the adversaries from breaking our CL-PRE scheme by means of re-encryption queries and re-encryption key queries corresponding to the challenge identity ID^* and the challenge ciphertext C^*.

We then define two different types of "indistinguishability of encryptions under chosen ciphertext attack" (IND-CCA) games against the Type I and the Type II adversaries, respectively. The first game between the Type I adversary (denoted by $\mathcal{A}_I$) and the challenger is defined as follows:

Setup: The challenger takes a security parameter k and runs the Setup algorithm. It gives the resulting public parameters params to $\mathcal{A}_I$ and keeps the master key mk to itself.

Phase 1: $\mathcal{A}_I$ issues queries $q_1, \ldots, q_m$ adaptively where query q_i is one of:

- **Extraction query** on ID_i. The challenger responds by running algorithm Partial-Private-Key-Extraction to generate the partial private key d_{ID_i} for ID_i.
- **Private Key query** on ID_i. If the public key for ID_i has not been replaced, the challenger responds by running algorithm Set-Private-Key to generate the private key sk_{ID_i} for ID_i.
- **Public Key query** on ID_i. The challenger responds by running algorithm Set-Public-Key to generate the public key pk_{ID_i} for ID_i.
- **Replace Public Key query** on the public key for ID_i. $\mathcal{A}_I$ can repeatedly replace the public key pk_{ID_i} for ID_i with any value pk'_{ID_i} of its choice.
- **Re-Encryption Key query** on (ID_i,ID_j). The challenger responds by running algorithm Set-Proxy-Re-Encryption-Key to generate the re-encryption key $rk_{i \to j}$.
- **Re-Encryption query** on (ID_i,ID_j,C_{ID_i}). The challenger responds by running algorithm Re-Encrypt to transform the ciphertext C_{ID_i} into the re-encrypted ciphertext C_{ID_j} using the re-encryption key $rk_{i \to j}$.
- **Decryption query** on (ID_i,C_{ID_i}). The challenger responds by running algorithm Decrypt to decrypt the ciphertext C_{ID_i} using the private key sk_{ID_i}. Even though the public key for ID_i may be replaced, the challenger is forced to respond with a correct answer as in [1,2].

Challenge: Once $\mathcal{A}_I$ decides that Phase 1 is over, it outputs two equal length plaintexts $m_0, m_1 \in \mathcal{M}$ and an identity ID^* of uncorrupted private key, on which it wishes to be challenged. In particular, ID^* may not have been submitted to both Replace Public Key and Extraction queries. Moreover, $\mathcal{A}_I$ is restricted to the choice of ID^* such that trivial decryption is not possible

using keys extracted in Phase 1, e.g., by using re-encryption keys to transform C_{ID^*} into C_{ID_i} for which $\mathcal{A}_I$ holds a decryption key. The challenger picks a random bit $b \in \{0,1\}$ and computes the challenge ciphertext $C^* = \mathsf{Encrypt}(m_b,\mathsf{params},\mathsf{ID}^*,pk_{ID^*})$, where pk_{ID^*} is the public key currently associated with ID^*. It sends C^* as the challenge to $\mathcal{A}_I$.

Phase 2: $\mathcal{A}_I$ issues more queries $q_{m+1}, \ldots, q_n$ adaptively where q_i is one of:

- **Decryption query** on (ID_i,C_{ID_i}). If (ID_i,C_{ID_i}) is not a *challenge derivative*, the challenger responds as in Phase 1. The definition of challenge derivative is as follows [10,16]:
  1. (ID^*,C^*) is a challenge derivative of itself.
  2. If (ID_i,C_{ID_i}) is a challenge derivative, $\mathcal{A}_I$ has issued the Re-Encryption query on $(\mathsf{ID}_i,\mathsf{ID}_j,C_{ID_i})$ to receive a new ciphertext C_{ID_j}, then $(\mathsf{ID}_j, C_{ID_j})$ is a challenge derivative.
  3. If (ID_i,C_{ID_i}) is a challenge derivative, $\mathcal{A}_I$ has issued the Re-Encryption Key query on $(\mathsf{ID}_i,\mathsf{ID}_j)$ to obtain a re-encryption key $rk_{i\to j}$, and $C_{ID_j} = \mathsf{Re\text{-}Encrypt}(\mathsf{params},rk_{i\to j},C_{ID_i})$, then (ID_j,C_{ID_j}) is a challenge derivative.
- **Extraction query** on ID_i. The challenger responds as in Phase 1 except that $\mathcal{A}_I$ has issued the Replace Public Key query on ID_i and a challenge derivative (ID_i,C_{ID_i}) exists.
- **Private Key query** on $\mathsf{ID}_i \neq \mathsf{ID}^*$ where a challenge derivative (ID_i,C_{ID_i}) does not exist. The challenger responds as in Phase 1.
- **Public Key query** on ID_i. The challenger responds as in Phase 1.
- **Replace Public Key query** on the public key for ID_i. The challenger responds as in Phase 1.
- **Re-Encryption Key query** on $(\mathsf{ID}_i,\mathsf{ID}_j)$. The challenger responds as in Phase 1 except that $\mathsf{ID}_i = \mathsf{ID}^*$ and $\mathcal{A}_I$ has issued the Private Key query on ID_j.
- **Re-Encryption query** on $(\mathsf{ID}_i,\mathsf{ID}_j,C_{ID_i})$. The challenger responds as in Phase 1 except that $\mathcal{A}_I$ has issued the Private Key query on ID_j and (ID_i,C_{ID_i}) is a challenge derivative.

Guess: Finally, $\mathcal{A}_I$ outputs a guess $b' \in \{0,1\}$. $\mathcal{A}_I$ wins if $b' = b$.

We define the advantage of $\mathcal{A}_I$ in breaking the CL-PRE scheme as

$$\mathrm{Adv}(\mathcal{A}_I) = \left|\mathsf{Pr}[b = b'] - \frac{1}{2}\right|$$

Definition 2. *We say that a unidirectional CL-PRE scheme is (t,ϵ)-adaptive chosen ciphertext (IND-CCA) secure against the Type I adversary if for any t-time adversary $\mathcal{A}_I$ we have that $Adv(\mathcal{A}_I) < \epsilon$.*

We now define the second game between the Type II adversary (denoted by $\mathcal{A}_{II}$) and the challenger as follows:

Setup: The challenger takes a security parameter k and runs the Setup algorithm. It gives the resulting public parameters params and the master key mk to $\mathcal{A}_{II}$.

Phase 1: $\mathcal{A}_{II}$ issues queries $q_1, \ldots, q_m$ adaptively where query q_i is one of:

- **Private Key query** on ID_i. The challenger responds by running algorithm Set-Private-Key to generate the private key sk_{ID_i} for ID_i.
- **Public Key query** on ID_i. The challenger responds by running algorithm Set-Public-Key to generate the public key pk_{ID_i} for ID_i.
- **Re-Encryption Key query** on (ID_i,ID_j). The challenger responds by running algorithm Set-Proxy-Re-Encryption-Key to generate the re-encryption key $rk_{i \to j}$.
- **Re-Encryption query** on (ID_i,ID_j,C_{ID_i}). The challenger responds by running algorithm Re-Encrypt to transform the ciphertext C_{ID_i} into the re-encrypted ciphertext C_{ID_j} using the re-encryption key $rk_{i \to j}$.
- **Decryption query** on (ID_i,C_{ID_i}). The challenger responds by running algorithm Decrypt to decrypt the ciphertext C_{ID_i} using the private key sk_{ID_i}.

Challenge: Once $\mathcal{A}_{II}$ decides that Phase 1 is over, it outputs two equal length plaintexts $m_0, m_1 \in \mathcal{M}$ and an identity ID^* of uncorrupted private key, on which it wishes to be challenged. In particular, $\mathcal{A}_{II}$ is restricted to the choice of ID^* such that trivial decryption is not possible using keys extracted in Phase 1, e.g., by using re-encryption keys to transform C_{ID^*} into C_{ID_i} for which $\mathcal{A}_{II}$ holds a decryption key. The challenger picks a random bit $b \in \{0,1\}$ and computes the challenge ciphertext $C^* = \mathsf{Encrypt}(m_b,\mathsf{params},\mathsf{ID}^*,pk_{ID^*})$. It sends C^* as the challenge to $\mathcal{A}_{II}$.

Phase 2: $\mathcal{A}_{II}$ issues more queries $q_{m+1}, \ldots, q_n$ adaptively where q_i is one of:

- **Decryption query** on (ID_i,C_{ID_i}). If (ID_i,C_{ID_i}) is not a challenge derivative, the challenger responds as in Phase 1.
- **Private Key query** on $\mathsf{ID}_i \neq \mathsf{ID}^*$. If a challenge derivative (ID_i,C_{ID_i}) does not exist. The challenger responds as in Phase 1.
- **Public Key query** on ID_i. The challenger responds as in Phase 1.
- **Re-Encryption Key query** on (ID_i,ID_j). The challenger responds as in Phase 1 except that $\mathsf{ID}_i = \mathsf{ID}^*$ and $\mathcal{A}_{II}$ has issued the Private Key query on ID_j.
- **Re-Encryption query** on (ID_i,ID_j,C_{ID_i}). The challenger responds as in Phase 1 except that $\mathcal{A}_{II}$ has issued the Private Key query on ID_j and (ID_i,C_{ID_i}) is a challenge derivative.

Guess: Finally, $\mathcal{A}_{II}$ outputs a guess $b' \in \{0,1\}$. $\mathcal{A}_{II}$ wins if $b' = b$.

We define the advantage of $\mathcal{A}_{II}$ in breaking the CL-PRE scheme as

$$\mathrm{Adv}(\mathcal{A}_{II}) = \left| \mathsf{Pr}[b = b'] - \frac{1}{2} \right|$$

Definition 3. *We say that a unidirectional CL-PRE scheme is (t, ϵ)-adaptive chosen ciphertext (IND-CCA) secure against the Type II adversary if for any t-time adversary $\mathcal{A}_{II}$ we have that $Adv(\mathcal{A}_{II}) < \epsilon$.*

3 Chosen-Ciphertext Secure CL-PRE Scheme

In this section, we present the first construction of CL-PRE based on bilinear pairing, which gets rid of key escrow problem inherent in identity-based PRE schemes as well as the certificate revocation problem in traditional public key based PRE ones. We apply the transformation technique of Libert and Quisquater [14], which is a modification of Fujisaki-Okamoto conversion [12] by hashing the recipient's identity and public key along with the message and the random string in the encryption algorithm, to our construction. As pointed out in [14], Fujisaki-Okamoto conversion is not enough to prevent an attacker from breaking the scheme by using public key replacement oracles in the certificateless setting. We then prove that the proposed scheme is secure against an adaptive chosen ciphertext attack (IND-CCA) in the random oracle model.

3.1 Bilinear Pairing and Complexity Assumption

We start by providing a brief overview of bilinear pairing and related computational problem on which our CL-PRE scheme is based. We use the following standard notation [8,9] to describe bilinear pairing:

1. $\mathbb{G}_1$ and $\mathbb{G}_2$ are two (multiplicative) cyclic groups of prime order q.
2. g is a generator of $\mathbb{G}_1$.
3. $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ is a bilinear map.

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two groups as above. A bilinear map is a map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ with the following properties:

1. Bilinear : for all $u, v \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q^*$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
2. Non-degenerate : $e(g, g) \neq 1_{\mathbb{G}_2}$.

Note that $e(,)$ is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

The security of our scheme relies on the intractability of the following problem [8]. The p-Bilinear Diffie Hellman Inversion (p-BDHI) problem is defined as follows: given a tuple $(g, g^{\alpha}, \ldots, g^{\alpha^p}) \in \mathbb{G}_1^{p+1}$ as input, compute $e(g, g)^{1/\alpha} \in \mathbb{G}_2$. An algorithm $\mathcal{B}$ has advantage ϵ in solving the p-BDHI problem if

$$\mathsf{Pr}\big[\mathcal{B}(g, g^{\alpha}, \ldots, g^{\alpha^p}) = e(g, g)^{1/\alpha}\big] \geq \epsilon$$

Where the probability is over the random choice of generator $g \in \mathbb{G}_1$, the random choice of $\alpha \in \mathbb{Z}_q^*$, and the random bits of $\mathcal{B}$.

Definition 4. *We say that the (t, p, ϵ)-BDHI assumption holds in $\mathbb{G}_1$ if no t-time algorithm has advantage at least ϵ in solving the p-BDHI problem in $\mathbb{G}_1$.*

3.2 Construction

The detailed description of the scheme is as follows:

Setup: Given security parameters k and k_0, this algorithm performs as follows:

1. Choose a k-bit prime number q, bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of order q, and random generators $g, h \in \mathbb{G}_1$.
2. Pick a KGC's master secret key $\alpha \in \mathbb{Z}_q^*$ uniformly at random and compute a public key $g_1 = g^\alpha \in \mathbb{G}_1$.
3. Choose cryptographic hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$, $H_2 : \mathbb{G}_2^2 \to \{0,1\}^{n+k_0}$, $H_3 : \{0,1\}^* \to \mathbb{Z}_q^*$, $H_4 : \{0,1\}^* \to \mathbb{Z}_q^*$, $H_5 : \{0,1\}^* \to \mathbb{G}_1$, and compute the group elements $g_2 = e(g,g), g_3 = e(g,h) \in \mathbb{G}_2$.

The public parameters are

$$\mathsf{params} := \{\mathbb{G}_1, \mathbb{G}_2, e, g, h, g_1, g_2, g_3, H_1, H_2, H_3, H_4, H_5\}$$

where message space is $\mathcal{M} := \{0,1\}^n$.

Partial-Private-Key-Extract: This algorithm takes params and an identifier ID_A for a user A as inputs, computes $h_A = H_1(\mathsf{ID}_A) \in \mathbb{Z}_q^*$ and extracts A's partial private key $d_A = g^{1/(\alpha + h_A)} \in \mathbb{G}_1$.

Set-Secret-Value: Given params and ID_A as inputs, this algorithm selects $a_1, a_2 \in \mathbb{Z}_q^*$ uniformly at random and sets $x_A = (a_1, a_2) \in \mathbb{Z}_q^{*2}$ which is returned as user A's secret value.

Set-Private-Key: Given params, user A's partial private key d_A and secret value x_A, this algorithm returns the pair $sk_A = (d_A, x_A) \in \mathbb{G}_1 \times \mathbb{Z}_q^{*2}$ as a private key.

Set-Public-Key: This algorithm takes params and user A's secret value x_A as inputs, and produces user A's public key $pk_A = (g_3^{a_1}, g^{a_2}) \in \mathbb{G}_2 \times \mathbb{G}_1$.

Encrypt: To encrypt a message $m \in \mathcal{M}$ using the identifier ID_A and the public key $pk_A = (g_3^{a_1}, g^{a_2})$, the sender performs the following steps:

1. Check that pk_A is in $\mathbb{G}_1$ and $\mathbb{G}_2$, if not output $\perp$.
2. Compute $h_A = H_1(\mathsf{ID}_A) \in \mathbb{Z}_q^*$.
3. Choose a random $\sigma \in \{0,1\}^{k_0}$ and compute $r = H_4(m||\sigma||\mathsf{ID}_A||pk_A) \in \mathbb{Z}_q^*$.
4. Compute the ciphertext $C = (C_1, C_2, C_3, C_4)$:

$$C = ((g^{h_A} \cdot g_1)^r,\ h^r,\ (m||\sigma) \oplus H_2(g_2^r||(g_3^{a_1})^r),\ u^r)$$

where $u = H_5(\mathsf{ID}_A||pk_A||C_1||C_2||C_3)$.

Set-Proxy-Re-Encryption-Key: Given user B's identifier ID_B and public key $pk_B = (g_3^{b_1}, g^{b_2})$, and user A's identifier ID_A and public/private key pair (pk_A, sk_A), this algorithm performs as follows:

1. Choose a random $s \in \mathbb{Z}_q^*$ and compute $\mu = H_3(g_2^s||\mathsf{ID}_A||pk_A||\mathsf{ID}_B||pk_B) \in \mathbb{Z}_q^*$.
2. Compute $h_A = H_1(\mathsf{ID}_A) \in \mathbb{Z}_q^*$ and $h_B = H_1(\mathsf{ID}_B) \in \mathbb{Z}_q^*$.
3. Set the proxy re-encryption key $rk_{A\to B} = (rk_{A\to B}^{(1)}, rk_{A\to B}^{(2)}, rk_{A\to B}^{(3)}) = (g^{\mu/(\alpha+h_A)},\ (g^{h_B}\cdot g_1)^s,\ (g^{b_2})^{a_1})$.

Re-Encrypt: Given a re-encryption key $rk_{A\to B}$ and a ciphertext $C = (C_1, C_2, C_3, C_4)$ under the identifier ID_A and the public key pk_A, the proxy performs the following steps:

1. Set $u = H_5(\mathsf{ID}_A||pk_A||C_1||C_2||C_3)$ and compute $h_A = H_1(\mathsf{ID}_A) \in \mathbb{Z}_q^*$.
2. Check that $e(C_1, u) \stackrel{?}{=} e((g^{h_A}\cdot g_1), C_4)$ and $e(C_2, u) \stackrel{?}{=} e(h, C_4)$, if not output $\perp$.
3. Compute $C_1' = e(C_1, rk_{A\to B}^{(1)})$ and set $C_1'' = rk_{A\to B}^{(2)}$.
4. Compute $C_2' = e(C_2, rk_{A\to B}^{(3)})$.
5. Set the new ciphertext $C' = (C_1', C_1'', C_2', C_3, \mathsf{ID}_A, pk_A)$.

Decrypt

- To decrypt non re-encrypted ciphertext C, the receiver A performs as follows:
  1. Parse C as (C_1, C_2, C_3, C_4).
  2. Compute $h_A = H_1(\mathsf{ID}_A) \in \mathbb{Z}_q^*$.
  3. Compute $\omega = e(C_1, d_A)||e(g, C_2)^{a_1}$ and then $(m||\sigma) = C_3 \oplus H_2(\omega)$.
  4. If $C_1 = (g^{h_A}\cdot g_1)^r$ and $C_2 = h^r$, where $r = H_4(m||\sigma||\mathsf{ID}_A||pk_A)$, return m as the message. Otherwise, output $\perp$.

- To decrypt re-encrypted ciphertext C', the receiver B performs as follows:
  1. Parse C' as $(C_1', C_1'', C_2', C_3, \mathsf{ID}_A, pk_A)$.
  2. Compute $\kappa = e(C_1'', d_B)$ and $\mu = H_3(\kappa||\mathsf{ID}_A||pk_A||\mathsf{ID}_B||pk_B)$.
  3. Compute $\omega = {C_1'}^{1/\mu}||{C_2'}^{1/b_2}$ and then $(m||\sigma) = C_3 \oplus H_2(\omega)$.
  4. If $C_1' = g_2^{\mu\cdot r}$ and $C_2' = (g_3^{a_1})^{b_2\cdot r}$, where $r = H_4(m||\sigma||\mathsf{ID}_A||pk_A)$, return m as the message. Otherwise, output $\perp$.

The consistency of the construction is easy to check as follows:

- For the receiver A, we have

$$\begin{aligned} e(C_1, d_A)||e(g, C_2)^{a_1} &= e((g^{h_A}\cdot g^{\alpha})^r, g^{1/(\alpha+h_A)})||e(g, h^r)^{a_1} \\ &= e(g^{(h_A+\alpha)\cdot r}, g^{1/(\alpha+h_A)})||e(g, h)^{a_1\cdot r} \\ &= e(g, g)^r||e(g, h)^{a_1\cdot r} \\ &= g_2^r||(g_3^{a_1})^r \end{aligned}$$

- For the receiver B, we have

$$\begin{aligned} {C_1'}^{1/\mu}||{C_2'}^{1/b_2} &= e(g, g)^{\mu\cdot r/\mu}||e(g, h)^{a_1\cdot b_2\cdot r/b_2} \\ &= e(g, g)^r||e(g, h)^{a_1\cdot r} \\ &= g_2^r||(g_3^{a_1})^r \end{aligned}$$

3.3 Security Analysis

We prove the security of our CL-PRE scheme under the p-BDHI assumption described in Section 3.1.

Theorem 1. *Suppose the hash functions H_i ($i = 1, 2, 3, 4, 5$) are modeled as random oracles, the above CL-PRE scheme is secure in the sense of IND-CCA described in Section 2.2 under the p-BDHI assumption.*

To prove the theorem, the proof separately considers both kinds of adversaries to establish the chosen ciphertext security of the above CL-PRE scheme. We now prove Theorem 1 by combining the following two lemmas.

Lemma 1. *Suppose that a Type I IND-CCA adversary $\mathcal{A}_I$ has advantage ϵ over our CL-PRE scheme when running time in a time τ, asking at most q_{H_i} queries (H_i: i=1,2,3,4,5), at most q_{EX} extraction queries, at most q_{SK} private key queries, at most q_{PK} public key queries, at most q_R replace queries, at most q_{RK} re-encryption key queries, at most q_{RE} re-encryption queries, and at most q_D decryption queries. There exists an algorithm $\mathcal{B}$ to solve the p-BDHI problem with advantage*

$$\epsilon' \geq \frac{1}{q_{H_2}}\Big(\frac{\epsilon}{2q_{H_1}} - \frac{q_{H_4} + (q_{H_2} + q_{H_4})q_D}{2^{k_0}} - \frac{q_{RE} + 2q_D}{q}\Big)$$

The proof of Lemma 1 is found in Appendix A.

Lemma 2. *Suppose that a Type II IND-CCA adversary $\mathcal{A}_{II}$ has advantage ϵ over our CL-PRE scheme when running time in a time τ, asking at most q_{H_i} queries (H_i: i=1,2,3,4,5), at most q_{SK} private key queries, at most q_{PK} public key queries, at most q_{RK} re-encryption key queries, at most q_{RE} re-encryption queries, and at most q_D decryption queries. There exists an algorithm $\mathcal{B}$ to solve the p-BDHI problem with advantage*

$$\epsilon' \geq \frac{1}{q_{H_2}}\Big(\frac{\epsilon}{q_{H_1}} - \frac{q_{H_4} + (q_{H_2} + q_{H_4})q_D}{2^{k_0}} - \frac{q_{RE} + 2q_D}{q}\Big)$$

The proof of Lemma 2 will be found in the full version of the paper due to the space limitation.

4 Conclusion

In this paper, we have introduced the notion of certificateless proxy re-encryption and also provided precise definitions for constructing secure certificateless proxy re-encryption schemes. We have presented a concrete scheme based on bilinear pairing, which enjoys the advantages of certificateless public key cryptography while providing the functionalities of proxy re-encryption. The proposed scheme is unidirectional and compatible with existing certificateless encryption deployments.

Acknowledgements

This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD, Basic Research Promotion Fund) (KRF-2008-521-D00454).

References

1. Al-Riyami, S.S., Paterson, K.: Certificateless public key cryptography. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 452–473. Springer, Heidelberg (2003)
2. Al-Riyami, S.S., Paterson, K.: CBE from CL-PKE: A generic construction and efficient scheme. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 398–415. Springer, Heidelberg (2005)
3. Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. ACM Transactions on Information and System Security (TISSEC) 9(1), 1–30 (2006)
4. Ateniese, G., Benson, K., Hohenberger, S.: Key-Private Proxy Re-Encryption, Cryptography ePrint Archive, Report 2008/463 (2008)
5. Baek, J., Safavi-Naini, R., Susilo, W.: Certificateless public key encryption without pairing. In: Zhou, J., López, J., Deng, R.H., Bao, F. (eds.) ISC 2005. LNCS, vol. 3650, pp. 134–148. Springer, Heidelberg (2005)
6. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: ACM CCS' 93, pp. 62–73 (1993)
7. Blaze, M., Bleumer, G., Strauss, M.: Divertible protocols and atomic proxy cryptography. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 127–144. Springer, Heidelberg (1998)
8. Boneh, D., Boyen, X.: Efficient selective-id secure identity based encryption without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)
9. Boneh, D., Franklin, M.: Identity-based encryption from the weil paring. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
10. Canetti, R., Hohenberger, S.: Chosen-ciphertext secure proxy re-encryption, Cryptography ePrint Archive, Report 2007/171 (2007)
11. Deng, R.H., Weng, J., Liu, S., Chen, K.: Chosen-ciphertext secure proxy re-encryption without pairings. In: Franklin, M.K., Hui, L.C.K., Wong, D.S. (eds.) CANS 2008. LNCS, vol. 5339, pp. 1–17. Springer, Heidelberg (2008)
12. Fujisaki, E., Okamoto, T.: How to enhance the security of public-key encryption at minimum cost. In: Imai, H., Zheng, Y. (eds.) PKC 1999. LNCS, vol. 1560, pp. 53–68. Springer, Heidelberg (1999)
13. Green, M., Ateniese, G.: Identity-based proxy re-encryption, Cryptography ePrint Archive, Report 2006/473 (2006)
14. Libert, B., Quisquater, J.: On constructing certificateless cryptosystem from identity based encryption. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T.G. (eds.) PKC 2006. LNCS, vol. 3958, pp. 474–490. Springer, Heidelberg (2006)
15. Libert, B., Vergnaud, D.: Unidirectional chosen-ciphertext secure proxy re-encryption. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 360–379. Springer, Heidelberg (2008)
16. Shao, J., Cao, Z.: CCA-secure proxy re-encryption without pairings. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 357–376. Springer, Heidelberg (2009)

A Proof of Lemma 1

Proof. Let $\mathcal{A}_I$ be an adversary which has advantage ϵ in attacking the IND-CCA security of our CL-PRE scheme. We show how to build an algorithm $\mathcal{B}$ that uses $\mathcal{A}_I$ to solve the p-BDHI problem in $\mathbb{G}_1$. Given as input a random tuple $(g, g^{\alpha}, \ldots, g^{\alpha^p}) \in \mathbb{G}_1^{p+1}$, $\mathcal{B}$'s goal is to extract $e(g,g)^{1/\alpha}$ for some unknown α. As in [1,14], we start by distinguishing two kinds of Type I adversaries:

- Type I-A adversary : may replace the public key for the challenge identity ID^* as some point, but cannot ask for the corresponding partial private key.
- Type I-B adversary : may ask for the partial private key of the challenge identity ID^* at some point, but cannot replace the corresponding public key.

Therefore, since $\mathcal{B}$ should guess which kind of Type I adversary will be, it chooses a random bit $c \in \{0,1\}$. If $c = 0$, $\mathcal{B}$ bets on a Type I-A adversary. Otherwise, bets on a Type I-B adversary. $\mathcal{B}$ selects an index ℓ with $1 \leq \ell \leq q_{H_1}$ and an element $\psi_\ell \in \mathbb{Z}_q^*$ uniformly at random. $\mathcal{B}$ also chooses $w_1, \ldots, w_{\ell-1}, w_{\ell+1}, \ldots, w_{q_{H_1}} \in \mathbb{Z}_q^*$ and computes $\psi_i = \psi_\ell - w_i$ for $i = 1, \ldots, \ell-1, \ell+1, \ldots, q_{H_1}$.

Depending on the value of c, $\mathcal{B}$ now works by interacting with $\mathcal{A}_I$ as follows:

Setup: To generate public parameters params, $\mathcal{B}$ does the followings:

- Case $c = 0$. $\mathcal{B}$ builds a generator $h \in \mathbb{G}_1$ for which it knows $p-1$ pairs of the form $(\psi_i, h^{1/(\psi_i+\beta)})$ for random $i \neq \ell$. This work is done as follows:
  1. Let $f(z)$ be the polynomial $f(z) = \prod_{i=1, i\neq\ell}^{p}(z + w_i)$. Expand the terms of $f(z)$ to get $f(z) = \sum_{j=0}^{p-1} c_j z^j$. The constant term c_0 is non-zero.
  2. Compute $h = \prod_{j=0}^{p-1}(g^{(\alpha^j)})^{c_j} = g^{f(\alpha)}$ and $u = \prod_{j=1}^{p}(g^{(\alpha)^j})^{c_{j-1}} = g^{\alpha f(\alpha)}$. Note that $u = h^{\alpha}$.
  3. Construct $p-1$ pairs $(w_i, h_i = h^{1/(\alpha+w_i)})$ for any $i \in \{1, \ldots, p\} \setminus \{\ell\}$. To see this, write $f_i(z) = f(z)/(z + w_i) = \sum_{i=0}^{p-2} d_i z^i$. Then $h_i = h^{1/(\alpha+w_i)} = g^{f_i(\alpha)} = \prod_{i=0}^{p-2}(g^{(\alpha)^i})^{d_i}$.
  4. Compute the public key of the KGC as $g_1 = u^{-1} \cdot h^{-\psi_\ell} = h^{-\alpha-\psi_\ell}$ for unknown master key $\mathsf{mk} := \beta = -\alpha - \psi_\ell \in \mathbb{Z}_q^*$, and set $v = g^{\alpha}$ as another generator.
  5. Set $(\psi_i, h^{1/(\psi_i+\beta)}) = (\psi_i, h_i^{-1})$ for all $i \in \{1, \ldots, p\} \setminus \{\ell\}$.
- Case $c = 1$. $\mathcal{B}$ builds generators $h = g^{\alpha^{p-1}}, v = g^{\alpha} \in \mathbb{G}_1$. Then, it chooses a random $\beta \in \mathbb{Z}_q^*$ as the KGC's master key and computes the corresponding public key $g_1 = h^{\beta} \in \mathbb{Z}_q^*$.

Finally, $\mathcal{B}$ generates public parameters $\mathsf{params} = (\mathbb{G}_1, \mathbb{G}_2, e, h, v, g_1, e(h,h), e(h,v), H_1, H_2, H_3, H_4, H_5)$ and gives params to $\mathcal{A}_I$. Here $H_i (i = 1,2,3,4,5)$ are random oracles controlled by $\mathcal{B}$ as described below:

H_1-queries: For query on input ID_i, $\mathcal{B}$ responds with ψ_i and increments an index i.

H_2-queries: For query on input $\gamma_i \in \mathbb{G}_2^2$ of the form $(\gamma_{i,1} || \gamma_{i,2})$, $\mathcal{B}$ maintains a H_2-list of tuples (γ_i, ζ_i) as below:

1. If γ_i appears in the H_2-list, then $\mathcal{B}$ responds with ζ_i.

2. Otherwise, $\mathcal{B}$ chooses $\zeta_i \in \{0,1\}^{n+k_0}$ uniformly at random and adds (γ_i, ζ_i) to the H_2-list and returns ζ_i to $\mathcal{A}_I$.

H_3-queries: For query on input $\tau_{i,j} \in \{0,1\}^*$ of the form $(\kappa_{i,j}||\mathsf{ID}_i||pk_i||\mathsf{ID}_j||pk_j)$, $\mathcal{B}$ maintains a H_3-list of tuples $(\tau_{i,j}, \mu_{i,j})$ as below:

1. If $\tau_{i,j}$ appears in the H_3-list, then $\mathcal{B}$ responds with $\mu_{i,j}$.
2. Otherwise, $\mathcal{B}$ chooses $\mu_{i,j} \in \mathbb{Z}_q^*$ uniformly at random and adds $(\tau_{i,j}, \mu_{i,j})$ to the H_3-list and returns $\mu_{i,j}$ to $\mathcal{A}_I$.

H_4-queries: For query on input $\lambda_i \in \{0,1\}^*$ of the form $(m_i||\sigma_i||\mathsf{ID}_i||pk_i)$, $\mathcal{B}$ maintains a H_4-list of tuples $(m_i, \sigma_i, \mathsf{ID}_i, pk_i, r_i)$ as below:

1. If λ_i appears in the H_4-list, then $\mathcal{B}$ responds with r_i.
2. Otherwise, $\mathcal{B}$ chooses $r_i \in \mathbb{Z}_q^*$ uniformly at random and adds $(m_i, \sigma_i, \mathsf{ID}_i, pk_i, r_i)$ to the H_4-list and returns r_i to $\mathcal{A}_I$.

H_5-queries: For query on input $\eta_i \in \{0,1\}^*$ of the form $(\mathsf{ID}||pk||C_1||C_2||C_3)$, $\mathcal{B}$ maintains a H_5-list of tuples (η_i, y_i, ρ_i) as below:

1. If η_i appears in the H_5-list, then $\mathcal{B}$ responds with ρ_i.
2. Otherwise, $\mathcal{B}$ chooses a random $y_i \in \mathbb{Z}_q^*$ and adds $(\eta_i, y_i, \rho_i = v^{y_i})$ to the H_5-list and returns ρ_i to $\mathcal{A}_I$.

Phase 1: $\mathcal{A}_I$ issues queries $q_1, \ldots, q_m$ where query q_i is one of:

- Extraction queries. $\mathcal{B}$ responds to $\mathcal{A}_I$'s query on ID_i as follows:
  - Case $c = 0$. If $i = \ell$, $\mathcal{B}$ stops the simulation and outputs "failure". Otherwise, it responds with $h^{1/(\psi_i+\beta)}$ corresponding to $H_1(\mathsf{ID}_i) = \psi_i$.
  - Case $c = 1$. $\mathcal{B}$ responds with d_i by running the Partial-Private-Key-Extract algorithm as it knows the master key.
- Public key queries. For a public key query on ID_i, $\mathcal{B}$ checks whether the pk-list contains a tuple $(\mathsf{ID}_i, x_i = (x_{i,1}, x_{i,2}), pk_i)$ for this input.
  1. If it does, the previously defined value is returned.
  2. Otherwise, $\mathcal{B}$ does the followings:
     - Case $c = 0$. $\mathcal{B}$ picks random $x_{i,1}, x_{i,2} \in \mathbb{Z}_q^*$ and computes $pk_i = (e(h,v)^{x_{i,1}}, h^{x_{i,2}})$, then returns it to $\mathcal{A}_I$.
     - Case $c = 1$. If $i = \ell$, $\mathcal{B}$ picks a random $\delta \in \mathbb{G}_1$ and responds with $pk_i = (e(g,g), \delta)$ which is equal to $(e(h,v)^{1/\alpha^p}, h^x)$ for unknown $(1/\alpha^p, x)$. Otherwise, $\mathcal{B}$ responds with pk_i as in the simulation $c = 0$.
  3. $\mathcal{B}$ then adds $(\mathsf{ID}_i, x_i, pk_i)$ to the pk-list (If $i = \ell$ and $c = 1$, x_i is $\perp$ as an unknown value).
- Private key queries. For query on input ID_i, $\mathcal{B}$ performs as follows:
  1. If $i = \ell$, $\mathcal{B}$ stops the simulation and outputs "failure".
  2. If $i \neq \ell$, $\mathcal{B}$ recovers the corresponding tuple from the pk-list. If the pk-list contain the tuple $(\mathsf{ID}_i, x_i, pk_i)$,
     - Case $c = 0$. $\mathcal{B}$ sets $sk_i = (d_i, x_i)$ as it knows the partial private key and sends it to $\mathcal{A}_I$.
     - Case $c = 1$. $\mathcal{B}$ extracts the partial private key itself using the master key and returns $sk_i = (d_i, x_i)$ to $\mathcal{A}_I$.
  3. Otherwise, $\mathcal{B}$ makes a public key query on ID_i, then simulates as the above process.

- Replace public key queries. When $\mathcal{A}_I$ queries on input (ID_i, pk_i'), $\mathcal{B}$ does the followings:
  1. If $i = \ell$ and $c = 1$, $\mathcal{B}$ stops the simulation and outputs "failure".
  2. Otherwise, $\mathcal{B}$ checks that pk_i' is in $\mathbb{G}_1$ and $\mathbb{G}_2$. If does, replaces the corresponding tuple in the pk-list with $(\mathsf{ID}_i, \perp, pk_i')$.
- Re-encryption key queries. $\mathcal{B}$ responds to $\mathcal{A}_I$'s query on $(\mathsf{ID}_i,\mathsf{ID}_j)$ as follows:
  1. If $i = \ell$, $\mathcal{B}$ stops the simulation and outputs "failure".
  2. If $i \neq \ell$, $\mathcal{B}$ checks whether the rk-list contains a tuple $(\mathsf{ID}_i, \mathsf{ID}_j, rk_{i\to j}, s_{i,j})$ for this input.
     - If it does, the previously defined value is returned.
     - Otherwise, $\mathcal{B}$ does followings:
     (a) Recover the corresponding tuples $(\mathsf{ID}_i, x_i = (x_{i,1}, x_{i,2}), pk_i)$ and $(\mathsf{ID}_j, x_j = (x_{j,1}, x_{j,2}), pk_j)$ from the pk-list.
     (b) Choose a random $s_{i,j} \in \mathbb{Z}_q^*$ and evaluate $H_3(e(h,h)^{s_{i,j}} || \mathsf{ID}_i || pk_i || \mathsf{ID}_j || pk_j)$ to obtain $\mu_{i,j}$.
     (c) Simulate an extraction query on ID_i to obtain the corresponding partial private key $d_i = h^{1/(\beta+\psi_i)}$.
     (d) If $j = \ell$ and $c = 1$, set the re-encryption key $rk_{i\to j} = (rk_{i\to j}^{(1)}, rk_{i\to j}^{(2)}, rk_{i\to j}^{(3)}) = (h^{\mu_{i,j}/(\beta+\psi_i)}, (h^{\psi_j} \cdot g_1)^{s_{i,j}}, \delta^{x_{i,1}})$. Otherwise, set the re-encryption key $rk_{i\to j} = (rk_{i\to j}^{(1)}, rk_{i\to j}^{(2)}, rk_{i\to j}^{(3)}) = (h^{\mu_{i,j}/(\beta+\psi_i)}, (h^{\psi_j} \cdot g_1)^{s_{i,j}}, (h^{x_{j,2}})^{x_{i,1}})$.
     (e) Return the re-encryption key $rk_{i\to j}$ to $\mathcal{A}_I$ and add $(\mathsf{ID}_i,\mathsf{ID}_j, rk_{i\to j}, s_{i,j})$ to the rk-list.
- Re-encryption queries. For query on input $(\mathsf{ID}_i, \mathsf{ID}_j, C)$, $\mathcal{B}$ parses C as (C_1, C_2, C_3, C_4). Then, $\mathcal{B}$ recovers the corresponding tuples $(\mathsf{ID}_i, x_i, pk_i)$ and $(\mathsf{ID}_j, x_j, pk_j)$ from the pk-list and evaluates $H_5(\mathsf{ID}_i || pk_i || C_1 || C_2 || C_3)$ to obtain ρ_i. $\mathcal{B}$ checks that $e(C_1, \rho_i) \stackrel{?}{=} e((h^{\psi_i} \cdot g_1), C_4)$ and $e(C_2, \rho_i) \stackrel{?}{=} e(v, C_4)$. If not, $\mathcal{B}$ returns $\perp$. Otherwise,
  1. If $i = \ell$, $\mathcal{B}$ does the followings:
     - Look up the corresponding tuple $(m, \sigma, \mathsf{ID}_i, pk_i, r)$ in the H_4-list such that $C_1 = (h^{\psi_i} \cdot g_1)^r$ and $C_2 = v^r$. If there exists no such tuple, return $\perp$.
     - Choose a random $s \in \mathbb{Z}_q^*$ and evaluate $H_3(e(h,h)^s || \mathsf{ID}_i || pk_i || \mathsf{ID}_j || pk_j)$ to obtain $\mu_{i,j}$.
     - Set $C_1' = e(h,h)^{\mu_{i,j} \cdot r}$, $C_1'' = (h^{\psi_j} \cdot g_1)^s$, and $C_2' = e(h,v)^{x_{i,1} \cdot x_{j,2} \cdot r}$ in case $c = 0$ or $C_2' = e(g,g)^{x_{j,2} \cdot r}$ in case $c = 1$.
     - Output $C' = (C_1', C_1'', C_2', C_3, \mathsf{ID}_i, pk_i)$ to $\mathcal{A}_I$.
  2. If $i \neq \ell$, $\mathcal{B}$ simulates a re-encryption key query on $(\mathsf{ID}_i, \mathsf{ID}_j)$ to obtain $rk_{i\to j}$, then outputs $C' = \mathsf{Re\text{-}Encrypt}(params, rk_{i\to j}, C)$ to $\mathcal{A}_I$.
- Decryption queries. $\mathcal{B}$ responds to $\mathcal{A}_I$'s query on (ID_i, C) as below:
  1. If $i = \ell$, $\mathcal{B}$ parses C,
     - Case $C = (C_1, C_2, C_3, C_4)$. $\mathcal{B}$ does followings:

(a) Look up the corresponding tuples $(m, \sigma, \mathsf{ID}_i, pk_i, r)$ and (γ, ζ) in the H_4-list and the H_2-list, respectively, such that

$$C_1 = (h^{\psi_i} \cdot g_1)^r,\ C_2 = v^r,\ C_3 = (m||\sigma) \oplus \zeta$$

If there exist no such tuples, return $\perp$.

(b) Check that $C_4 \stackrel{?}{=} H_5(\mathsf{ID}_i||pk_i||C_1||C_2||C_3)^r$. If not return $\perp$, otherwise output m to $\mathcal{A}_I$.

- $C = (C_1', C_1'', C_2', C_3, \mathsf{ID}_j, pk_j)$. $\mathcal{B}$ does followings:

(a) Recover the corresponding tuple $(\mathsf{ID}_j, \mathsf{ID}_i, rk_{j\to i}, s_{j,i})$ from the rk-list.

(b) Check that $C_1'' \stackrel{?}{=} rk_{j\to i}^{(2)}$. If not return $\perp$.

(c) Recover the corresponding tuples $(\mathsf{ID}_j, x_j, pk_j)$ and $(\mathsf{ID}_i, x_i, pk_i)$ from the pk-list, then evaluate $H_3(e(h,h)^{s_{j,i}}||\mathsf{ID}_j||pk_j||\mathsf{ID}_i||pk_i)$ to obtain μ.

(d) Look up the corresponding tuples $(m, \sigma, \mathsf{ID}_j, pk_j, r)$ and (γ, ζ) in the H_4-list and the H_2-list, respectively, such that

$$C_1' = e(h,h)^{\mu \cdot r},\ C_2' = e(h,v)^{x_{j,1} \cdot x_{i,2} \cdot r},\ C_3 = (m||\sigma) \oplus \zeta$$

in case $c = 0$ or

$$C_1' = e(h,h)^{\mu \cdot r},\ C_2' = e(\delta,v)^{x_{j,1} \cdot r},\ C_3 = (m||\sigma) \oplus \zeta$$

in case $c = 1$. If there exist no such tuples, return $\perp$. Otherwise, output m to $\mathcal{A}_I$.

2. If $i \neq \ell$, $\mathcal{B}$ simulates a private key query on ID_i to obtain sk_i, then outputs $m = \mathsf{Decrypt}(params, sk_i, C)$ to $\mathcal{A}_I$.

Challenge: Once $\mathcal{A}_I$ decides that Phase 1 is over, it outputs two messages $m_0, m_1 \in \mathcal{M}$ and ID^* on which it wishes to be challenged. If $\mathsf{ID}^* \neq \mathsf{ID}_\ell$, $\mathcal{B}$ stops the simulation and outputs "failure". Otherwise, $\mathcal{B}$ picks a random bit $b \in \{0,1\}$ and a random $\sigma^* \in \{0,1\}^{k_0}$. Finally, $\mathcal{B}$ builds a challenge ciphertext for ID^* and the current public key pk_{ID^*} according to the value of c.

– Case $c = 0$. $\mathcal{B}$ picks a random $l \in \mathbb{Z}_q^*$ and a random string $\xi \in \{0,1\}^{n+k_0}$. $\mathcal{B}$ then defines the challenged ciphertext to be

$$C^* = (C_1^*, C_2^*, C_3^*, C_4^*) = (h^{-l},\ g^l,\ \xi,\ (g^y)^l)$$

where y is obtained from evaluating $H_5(\mathsf{ID}^*||pk_{ID^*}||C_1^*||C_2^*||C_3^*)$. Define $r = l/\alpha$. Since $\beta = -\alpha - \psi_\ell$, we have that

$$C_1^* = h^{-l} = (h^{-\alpha})^{l/\alpha} = h^{(\psi_\ell + \beta) \cdot r} = h^{\psi_\ell \cdot r} \cdot h^{\beta \cdot r} = (h^{H_1(ID^*)} \cdot g_1)^r,$$
$$C_2^* = g^l = (g^\alpha)^{l/\alpha} = v^r,$$
$$C_4^* = (g^y)^l = (g^{\alpha \cdot y})^{l/\alpha} = (v^y)^{l/\alpha} = \rho^r$$

Without issuing a H_2-query on the input $(e(h,h)^r||e(h,v)^{x_{\ell,1} \cdot r})$, $\mathcal{A}_I$ is unable to recognize that C^* is not an encryption of m_0 or m_1 and such an event would provide $\mathcal{B}$ with the searched p-BDHI solution.

– Case $c = 1$. $\mathcal{B}$ picks a random $l \in \mathbb{Z}_q^*$ and a random string $\xi \in \{0,1\}^{n+k_0}$. $\mathcal{B}$ then defines the challenged ciphertext to be

$$C^* = (C_1^*, C_2^*, C_3^*, C_4^*) = ((g^{\alpha^{p-2}})^{\psi_\ell \cdot l} \cdot (g^{\alpha^{p-2}})^{\beta \cdot l},\ g^l,\ \xi,\ (g^y)^l)$$

where y is obtained from evaluating $H_5(\mathsf{ID}^*||pk_{ID^*}||C_1^*||C_2^*||C_3^*)$. Define $r = l/\alpha$. Then, we have that

$$C_1^* = (g^{\alpha^{p-2}})^{\psi_\ell \cdot l} \cdot (g^{\alpha^{p-2}})^{\beta \cdot l} = (h^{\psi_\ell})^{l/\alpha} \cdot (h^{\beta})^{l/\alpha} = (h^{H_1(ID^*)} \cdot g_1)^r,$$
$$C_2^* = g^l = (g^\alpha)^{l/\alpha} = v^r,$$
$$C_4^* = (g^y)^l = (g^{\alpha \cdot y})^{l/\alpha} = (v^y)^{l/\alpha} = \rho^r$$

Without issuing a H_2-query on the input $(e(h,h)^r||e(g,g)^r)$, $\mathcal{A}_I$ is unable to recognize that C^* is not an encryption of m_0 or m_1 and such an event would provide $\mathcal{B}$ with the searched p-BDHI solution.

Finally, $\mathcal{B}$ returns C^* as the challenge ciphertext to $\mathcal{A}_I$.

Phase 2: $\mathcal{A}_I$ issues more queries. $\mathcal{B}$ responds as Phase 1 with the restrictions described in Section 2.2.

Guess: $\mathcal{A}_I$ outputs its guess $b' \in \{0,1\}$ which is ignored.

To produce a result, $\mathcal{B}$ chooses a random tuple $(\gamma_1||\gamma_2, \zeta)$ from the H_2-list with probability $1/q_{H_2}$. We distinguish two cases according to the value of c.

– Case $c = 0$. We have $\gamma_1 = e(h,h)^r = e(g,g)^{f(\alpha)^2 \cdot l/\alpha}$ and $\mathcal{B}$ can extract the p-BDHI solution by noting that, if $\gamma^* = e(g,g)^{1/\alpha}$, then

$$\gamma_1 = e(g,g)^{f(\alpha)^2 \cdot l/\alpha} = \Big(\gamma^{*(c_0^2)} \cdot e(\prod_{i=0}^{p-2} g^{c_{i+1} \cdot \alpha^i}, \prod_{j=0}^{p-1} g^{c_j \cdot \alpha^j}) \cdot e(\prod_{i=0}^{p-2} g^{c_{i+1} \cdot \alpha^i}, g^{c_0})\Big)^l$$

– Case $c = 1$. We have $\gamma_2 = e(g,g)^r = e(g,g)^{l/\alpha}$ and $\mathcal{B}$ can extract the p-BDHI solution by noting that $\gamma^* = \gamma_2^{1/l}$.

Let us analyze the simulation. The main idea of our analysis is borrowed from [5,11]. We first evaluate the simulations of the random oracles. From the constructions of H_1, H_3 and H_5, It is obvious that the simulations of H_1, H_3 and H_5 are perfect. Let AskH_2^* be the event that $(e(h,h)^r||e(h,v)^{x_{\ell,1} \cdot r})$ in case $c = 0$ or $(e(h,h)^r||e(g,g)^r)$ in case $c = 1$ has been queried to H_2, and AskH_4^* be the event that $(m_b||\sigma^*||\mathsf{ID}^*||pk_{ID^*})$ has been queried to H_4. The simulations of H_2 and H_4 are also perfect, as long as AskH_2^* and AskH_4^* did not occur, where b and σ^* are chosen by $\mathcal{B}$ in the Challenge phase.

The simulated challenge ciphertext is identically distributed as the real one from the construction.

Next, we analyze the simulation of the re-encryption oracle. This simulation is also perfect, unless $\mathcal{A}_I$ can submit valid original ciphertexts without querying

hash function H_4. Let $\mathsf{ReEncErr}$ denote this event. However, since H_4 acts as a random oracle and $\mathcal{A}_I$ issues at most q_{RE} re-encryption queries, we have $\mathsf{Pr}[\mathsf{ReEncErr}] \leq \frac{q_{RE}}{q}$.

The simulation of the decryption oracle is perfect, with the exception that simulation errors may occur in rejecting some valid ciphertexts. However, these errors are not significant as shown below: Suppose a ciphertext C has been queried to the decryption oracle. Even if C is valid, there is a possibility that C can be produced without querying $(g_2^r||(g_3^{x_1})^r)$ to H_2, where $r = H_4(m||\sigma||\mathsf{ID}||pk)$. Let Valid be an event that C is valid. Let AskH_2 and AskH_4 be the events that $(g_2^r||(g_3^{x_1})^r)$ has been queried to H_2 and $(m||\sigma||\mathsf{ID}||pk)$ has been queried to H_4, respectively. We then have that

$$\begin{aligned}\mathsf{Pr}[\mathsf{Valid}|\neg\mathsf{AskH}_2] &= \mathsf{Pr}[\mathsf{Valid} \wedge \mathsf{AskH}_4|\neg\mathsf{AskH}_2] + \mathsf{Pr}[\mathsf{Valid} \wedge \neg\mathsf{AskH}_4|\neg\mathsf{AskH}_2] \\ &\leq \mathsf{Pr}[\mathsf{AskH}_4|\neg\mathsf{AskH}_2] + \mathsf{Pr}[\mathsf{Valid}|\neg\mathsf{AskH}_4 \wedge \neg\mathsf{AskH}_2] \\ &\leq \frac{q_{H_4}}{2^{k_0}} + \frac{1}{q}\end{aligned}$$

and similarly we have $\mathsf{Pr}[\mathsf{Valid}|\neg\mathsf{AskH}_4] \leq \frac{q_{H_2}}{2^{k_0}} + \frac{1}{q}$. Therefore, we have that

$$\begin{aligned}\mathsf{Pr}[\mathsf{Valid}|(\neg\mathsf{AskH}_2 \vee \neg\mathsf{AskH}_4)] &\leq \mathsf{Pr}[\mathsf{Valid}|\neg\mathsf{AskH}_2] + \mathsf{Pr}[\mathsf{Valid}|\neg\mathsf{AskH}_4] \\ &\leq \frac{q_{H_2} + q_{H_4}}{2^{k_0}} + \frac{2}{q}\end{aligned}$$

Let DecErr be the event that $\mathsf{Pr}[\mathsf{Valid}|(\neg\mathsf{AskH}_2 \vee \neg\mathsf{AskH}_4)]$ happens during the entire simulation. Then, since $\mathcal{A}_I$ issues at most q_D decryption queries, we have

$$\mathsf{Pr}[\mathsf{DecErr}] \leq \frac{(q_{H_2} + q_{H_4})q_D}{2^{k_0}} + \frac{2q_D}{q}$$

If $\mathcal{B}$ does not abort during the simulation, it is clear that the simulations of the other oracles are perfect. Now, let us calculate the probability that $\mathcal{B}$ does not abort during the simulation. Let $\neg\mathsf{Abort}$ denote this event. As the proof technique in [1,14], we define the following events:

- $\mathcal{H}$: $\mathcal{A}_I$ chooses ID_ℓ as the challenge identity ID^*.
- $\mathcal{F}_0$: $\mathcal{A}_I$ extracts the partial private key for ID_ℓ at some point.
- $\mathcal{F}_1$: $\mathcal{A}_I$ replaces the public key of ID_ℓ at some point.

Following the above events, $\mathcal{B}$ could abort for one of the following reasons:

1. Because $c = 0$ and the event $\mathcal{F}_0$ occurs during the simulation.
2. Because $c = 1$ and the event $\mathcal{F}_1$ occurs during the simulation.
3. Because of a private key query for the identity ID_ℓ. Let $\mathcal{F}_2$ denote this event.
4. Because of a re-encryption key query for $(\mathsf{ID}_\ell, \mathsf{ID}_j)$. Let $\mathcal{F}_3$ denote this event.
5. Because $\mathcal{A}_I$ chooses a challenge identity $\mathsf{ID}^* \neq \mathsf{ID}_\ell$. Let $\neg\mathcal{H}$ denote this event.

We also name the event $(c = i) \wedge \mathcal{F}_i$ as $\mathcal{H}_i$ for $i = 0, 1$. It is obvious that the event $\mathcal{H}$ implies the events $\neg\mathcal{F}_2$ and $\neg\mathcal{F}_3$. Therefore, the probability that $\mathcal{B}$ does not abort during the simulation is

$$\begin{aligned}\mathsf{Pr}[\neg\mathsf{Abort}] &= \mathsf{Pr}[\neg\mathcal{H}_0 \wedge \neg\mathcal{H}_1 \wedge \neg\mathcal{F}_2 \wedge \neg\mathcal{F}_3 \wedge \mathcal{H}] \\ &= \mathsf{Pr}[\neg\mathcal{H}_0 \wedge \neg\mathcal{H}_1 \wedge \mathcal{H}] = \mathsf{Pr}[\neg\mathcal{H}_0 \wedge \neg\mathcal{H}_1 | \mathcal{H}] \cdot \mathsf{Pr}[\mathcal{H}] \\ &= \frac{1}{q_{H_1}} \mathsf{Pr}[\neg\mathcal{H}_0 \wedge \neg\mathcal{H}_1 | \mathcal{H}]\end{aligned}$$

Since the events $\mathcal{H}_0$ and $\mathcal{H}_1$ are mutually exclusive, we have that

$$\mathsf{Pr}[\neg\mathcal{H}_0 \wedge \neg\mathcal{H}_1 | \mathcal{H}] = 1 - \mathsf{Pr}[\mathcal{H}_0 | \mathcal{H}] - \mathsf{Pr}[\mathcal{H}_1 | \mathcal{H}]$$

On the other hand, we have that

$$\mathsf{Pr}[\mathcal{H}_i | \mathcal{H}] = \mathsf{Pr}[(c = i) \wedge \mathcal{F}_i | \mathcal{H}] = \frac{1}{2}\mathsf{Pr}[\mathcal{F}_i | \mathcal{H}]$$

because the event $\mathcal{F}_i | \mathcal{H}$ is independent of the event $c = i$ for $i = 0, 1$. Finally, as $\mathsf{Pr}[\mathcal{F}_0 \wedge \mathcal{F}_1 | \mathcal{H}] = 0$, this implies that $\mathsf{Pr}[\mathcal{F}_0 | \mathcal{H}] + \mathsf{Pr}[\mathcal{F}_1 | \mathcal{H}] \leq 1$. Therefore, we have that

$$\begin{aligned}\mathsf{Pr}[\neg\mathsf{Abort}] &= \frac{1}{q_{H_1}} \Big(1 - \frac{1}{2}\mathsf{Pr}[\mathcal{F}_0 | \mathcal{H}] - \frac{1}{2}\mathsf{Pr}[\mathcal{F}_1 | \mathcal{H}]\Big) \\ &\geq \frac{1}{2q_{H_1}}\end{aligned}$$

Let Good denote the event $(\mathsf{AskH}_2^* \vee (\mathsf{AskH}_4^* | \neg\mathsf{AskH}_2^*) \vee \mathsf{ReEncErr} \vee \mathsf{DecErr}) | \neg\mathsf{Abort}$. If Good does not happen, due to the randomness of the output of the random oracle H_2, it is obvious that $\mathcal{A}_I$ cannot gain any advantage greater than $1/2$ in guessing b. Namely, we have $\mathsf{Pr}[b = b' | \neg\mathsf{Good}] = \frac{1}{2}$. Hence, by splitting $\mathsf{Pr}[b = b']$, we have that

$$\begin{aligned}\mathsf{Pr}[b = b'] &= \mathsf{Pr}[b = b' | \neg\mathsf{Good}]\mathsf{Pr}[\neg\mathsf{Good}] + \mathsf{Pr}[b = b' | \mathsf{Good}]\mathsf{Pr}[\mathsf{Good}] \\ &\leq \frac{1}{2}\mathsf{Pr}[\neg\mathsf{Good}] + \mathsf{Pr}[\mathsf{Good}] = \frac{1}{2} + \frac{1}{2}\mathsf{Pr}[\mathsf{Good}]\end{aligned}$$

and

$$\mathsf{Pr}[b = b'] \geq \mathsf{Pr}[b = b' | \neg\mathsf{Good}]\mathsf{Pr}[\neg\mathsf{Good}] = \frac{1}{2} - \frac{1}{2}\mathsf{Pr}[\mathsf{Good}]$$

We then have that

$$\Big|\mathsf{Pr}[b = b'] - \frac{1}{2}\Big| \leq \frac{1}{2}\mathsf{Pr}[\mathsf{Good}]$$

By definition of the advantage for the $\mathsf{IND\text{-}CCA}$ adversary $\mathcal{A}_I$, we have that

$$\begin{aligned}\epsilon &= |2 \times \mathsf{Pr}[b = b'] - 1| \\ &\leq \mathsf{Pr}[\mathsf{Good}] = \mathsf{Pr}[(\mathsf{AskH}_2^* \vee (\mathsf{AskH}_4^* | \neg\mathsf{AskH}_2^*) \vee \mathsf{ReEncErr} \vee \mathsf{DecErr}) | \neg\mathsf{Abort}] \\ &= \frac{(\mathsf{Pr}[\mathsf{AskH}_2^*] + \mathsf{Pr}[\mathsf{AskH}_4^* | \neg\mathsf{AskH}_2^*] + \mathsf{Pr}[\mathsf{ReEncErr}] + \mathsf{Pr}[\mathsf{DecErr}])}{\mathsf{Pr}[\neg\mathsf{Abort}]}\end{aligned}$$

Since $\mathsf{Pr}[\mathsf{AskH}_4^*|\neg\mathsf{AskH}_2^*] \leq \frac{q_{H_4}}{2^{k_0}}$, $\mathsf{Pr}[\mathsf{ReEncErr}] \leq \frac{q_{RE}}{q}$, $\mathsf{Pr}[\mathsf{DecErr}] \leq \frac{(q_{H_2}+q_{H_4})q_D}{2^{k_0}} + \frac{2q_D}{q}$ and $\mathsf{Pr}[\neg\mathsf{Abort}] \geq \frac{1}{2q_{H_1}}$, we obtain

$$\begin{aligned}\mathsf{Pr}[\mathsf{AskH}_2^*] &\geq \epsilon \cdot \mathsf{Pr}[\neg\mathsf{Abort}] - \mathsf{Pr}[\mathsf{AskH}_4^*|\neg\mathsf{AskH}_2^*] - \mathsf{Pr}[\mathsf{ReEncErr}] - \mathsf{Pr}[\mathsf{DecErr}] \\ &\geq \frac{\epsilon}{2q_{H_1}} - \frac{q_{H_4}}{2^{k_0}} - \frac{q_{RE}}{q} - \frac{(q_{H_2}+q_{H_4})q_D}{2^{k_0}} - \frac{2q_D}{q} \\ &\geq \frac{\epsilon}{2q_{H_1}} - \frac{q_{H_4}+(q_{H_2}+q_{H_4})q_D}{2^{k_0}} - \frac{q_{RE}+2q_D}{q}\end{aligned}$$

Finally, notice that $\mathcal{B}$ solves the p-BDHI problem with probability $1/q_{H_2}$ if AskH_2^* happens. Consequently, we obtain

$$\epsilon' \geq \frac{1}{q_{H_2}}\mathsf{Pr}[\mathsf{AskH}_2^*] \geq \frac{1}{q_{H_2}}\Big(\frac{\epsilon}{2q_{H_1}} - \frac{q_{H_4}+(q_{H_2}+q_{H_4})q_D}{2^{k_0}} - \frac{q_{RE}+2q_D}{q}\Big)$$

□

Detecting Hidden Encrypted Volumes

Christopher Hargreaves and Howard Chivers

Centre for Forensic Computing, Cranfield Univeristy, Shrivenham, SN6 8LA, UK
{c.j.hargreaves,h.chivers}@cranfield.ac.uk

Abstract. Hidden encrypted volumes can cause problems in digital investigations since they provide criminal suspects with a range of opportunities for deceptive anti-forensics and a countermeasure to legislation written to force suspects to reveal decryption keys. This paper describes how hidden encrypted volumes can be detected, and their size estimated. The paper shows how multiple copies of an encrypted container can be obtained from a single disk image of Windows Vista and Windows 7 systems using the Volume Shadow Copy feature, and how the changes between shadow copies can be visualised to detect hidden volumes. The visualisation assists in the presentation of this information to a court, and exposes patterns of change which allows the size and file system of the hidden volume to be determined.

Keywords: Forensic Computing, Encryption, Hidden Volumes, RIPA, TrueCrypt.

1 Introduction

This paper examines the problem of hidden encrypted volumes during digital forensic investigations. A hidden encrypted volume is a feature of certain encryption systems that allows two keys to be created for a volume; one key decrypts the true contents (hidden volume) and the other key (the 'duress key') decrypts some pre-arranged innocent content (cover volume). This could pose a challenge in digital investigations since one of the approaches to gaining access to encrypted evidence is the use of legislation to force a suspect to provide decryption keys. If the suspect provides the 'duress key' and the data is decrypted, since it is not possible to tell if there is any additional hidden content, the effectiveness of this legislative approach is reduced. It is also possible that a password is provided more subtly with the intent to deceive the forensic analyst; e.g. the password to the cover volume is written on a post-it-note stuck to the bottom of a keyboard, making the investigator believe that they have access to the all the encrypted data.

This paper provides a practical solution to the problem of identifying the existence of hidden encrypted volumes and furthermore places it in a forensic computing context, which includes the difficulty of demonstrating the existence of such a hidden volume to a court. In addition the paper also shows how information about the hidden volume (such as its size) can be inferred through an examination of the changes in the free space of the cover volume.

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 233–244, 2010.

The paper is organised as follows: the remainder of this section details the problem of encrypted evidence and possible means of gaining access. It discusses the legislative approach in the UK, i.e. making the failure to provide decryption keys on request an offense (Part 3 of the UK Regulation of Investigatory Powers Act). Limitations of this legislation are also discussed in terms of technical measures, such as hidden volumes, that can be used to frustrate this approach. A particular implementation of hidden volumes is discussed (TrueCrypt), which is used throughout the later examples. Section 2 provides a summary of related work, and Section 3 describes the methodology for the research, including how multiple copies of an encrypted container can be obtained from a single disk image using the Volume Shadow Copy feature of Windows Vista and Windows 7. Section 3 also shows how the changes that occur between multiple versions of the container can be visualised and how information about the hidden volume can be extracted. Section 4 evaluates the approach and Sections 5 and 6 provide a discussion of future work and the conclusions.

1.1 Encrypted Digital Evidence

In a digital investigation there are a number of approaches that can be used to attempt to gain access to encrypted evidence. These are discussed in [1], [2], [3] and [4], and can be summarised as:

- Persuading/forcing suspect to provide the decryption key.
- Locating copies of unencrypted data.
- Locating keys or passphrases.
- Intelligent password attacks.
- Exhaustive key search.
- Exploiting implementation vulnerabilities.
- Hardware or software surveillance.

While any of these approaches can be used, legislation has been passed in the UK to make it an offence for a suspect to fail to provide means to access the encrypted information. This makes the option of forcing the suspect to provide decryption keys more viable. This legislation is contained in Part 3 of the Regulation of Investigatory Powers Act 2000 (RIPA) [5] and the requirements of responding to a RIPA notice are explained in detail in [6]. On receiving a RIPA Part 3 notice, the person concerned is required to either provide the electronic information in intelligible form or to disclose the key to enable the data to be put into intelligible form. While this legislative approach can be used to prosecute those who do not provide decryption keys, there are technical solutions that can be used as a countermeasure to this approach. One such countermeasure is the use of hidden volumes.

Hidden volumes employ the principles of steganography, meaning that the existence of data is hidden in addition to the content. Steganography implementations often involve hiding some secret data within a ‘cover file’, for example hiding text in redundant elements of a jpeg. In the case of hidden encrypted volumes the steganography is implemented as part of the encryption system, where one password decrypts the real content, and a second password decrypts some prearranged innocent content. This is done in such a way that it is not possible to tell if there is also

additional hidden content. An example implementation is detailed in the following section and discusses the popular open source product TrueCrypt.

1.2 TrueCrypt and Hidden Encrypted Volumes

TrueCrypt is a "software system for establishing and maintaining an on-the-fly-encrypted volume" meaning that "data are automatically encrypted or decrypted right before they are loaded or saved, without any user intervention" [7]. TrueCrypt has become a popular tool for encrypting data with over 12 million downloads as of December 2009. TrueCrypt can create encrypted containers and can encrypt full volumes or disks. TrueCrypt also offers hidden volume functionality so that one password decrypts the true content and another password (a 'duress' password) decrypts some prearranged innocent content. The structures of a standard TrueCrypt volume and a 'cover volume' containing a hidden volume are shown in Figure 1.

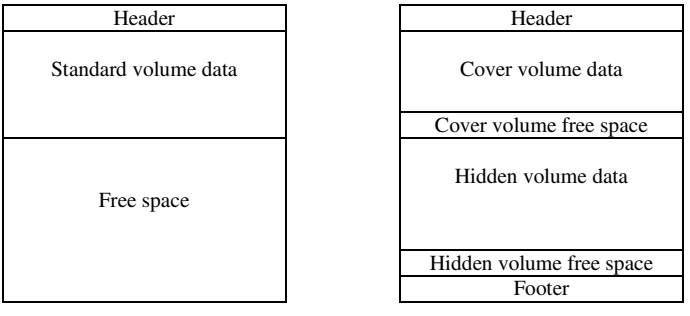


Fig. 1. The structure of a standard TrueCrypt volume (left) and one containing a hidden volume (right)

While the structures of the volumes are shown above, to an analyst encountering an encrypted volume, all that is visible is random data since the header, footer, volume data and volume free space are all encrypted. Furthermore, in a standard TrueCrypt volume the free space is filled with random data [8]. The process by which a TrueCrypt volume that contains a hidden volume is decrypted is as follows:

- The user enters a password and a key is derived from it
- That key used to attempt to decrypt the footer
- If the footer decryption is successful then the volume key is obtained from the footer and used to decrypt the hidden volume
- If the footer is not successfully decrypted then the key is used to attempt to decrypt the header
- If the header decryption is successful then the volume key is obtained from the header and used to decrypt the cover volume

The suspect can therefore use one of two passwords to access the volume. If the 'true' password is provided then it successfully decrypts the footer and access is provided to the hidden volume; if the 'duress' password is provided then it fails to decrypt the footer but decrypts the header and accesses the cover volume data.

In the second case (using the 'duress' key), the process cannot be distinguished from decrypting a standard volume that does not contain a hidden volume, since in both cases the footer will fail to decrypt and the process moves on to attempt decryption of the header. In addition, since free space on a volume is filled with random data, free space on a standard volume is indistinguishable from a hidden container embedded in the free space of a cover volume. Therefore if a suspect is forced to provide a key, if the provided key decrypts a standard volume it is not possible to determine if there is a secondary, hidden volume that would be accessible if a second password was provided.

1.3 Summary

While there are a number of options for gaining access to encrypted data, the simplest is often to ask for the password [10]. While the suspect may choose not to cooperate, in the UK there is now legislation in place to encourage the production of decryption keys (RIPA Part 3). However, technical measures are available to counter this and can take the form of systems that use multiple keys -- one to decrypt the true content and another to decrypt some pre-arranged innocent content.

2 Related Work

Despite the careful design of TrueCrypt, in practice it may still be possible to infer the presence of a hidden volume. This is discussed in both [9] and in the TrueCrypt documentation [8].

Three threat models are described in [9] to which a hidden volume system could be subjected. These are: *one time access*, where the attacker has a single snapshot of the volume; *intermittent access*, where several versions of the volume are available over time; and *regular access*, where many versions of the volume are available taken at short intervals.

Several opportunities to infer the presence of a hidden volume are presented in [9], assuming the most restrictive model (one time access). These include data leakage through the operating system, e.g. shortcut files that are created automatically and point to data on the hidden volume; data leakage through 'primary applications', e.g. Microsoft Word auto-saving copies of a file from a hidden volume in an unencrypted area of the disk; and data leakage through 'non-primary applications' e.g. Google Desktop indexing data on hidden volumes.

In addition to these opportunities, [9] also highlights that if disk snapshots are available at close enough intervals then it is possible to detect the existence of hidden data since "seemingly random bytes on the hard drive will change". However, no practical demonstration of this is provided and it is implied that intermittent or regular access is required to the disk.

The work described here shows how hidden volumes can be detected given only a single copy of a suspect's disk, how information about the size and file system of the hidden volume can be deduced, and demonstrates a visualisation of the results in the context of forensic computing, to allow such inferences to be explained to a court.

3 Methodology

3.1 General Methodology

Previous research has suggested that if multiple copies of an encrypted volume are available then detection of hidden volumes is possible [9], but at least intermittent access is needed to the disk. This paper provides a practical demonstration of how this is also possible from a single disk image by exploiting the Volume Shadow Copy functionality of Windows Vista and Windows 7. This feature is used to obtain multiple copies of encrypted containers produced using TrueCrypt version 6.3a. The term 'encrypted container' is used when referring to the use of Volume Shadow Copy since the technique cannot be used to obtain multiple versions of full volume or full disk encrypted data since Shadow Copies are created for files only. However, all other aspects of this research apply equally to full volume or disk encryption as long as intermittent access is available (this includes obtaining back-up copies), and in these aspects of the research the term 'volume' is used.

Once multiple copies of a volume are obtained, the paper shows how the changes can be visualised, making the presence of a hidden volume apparent in a form that is useful not only to an investigator but also to any non-expert decision makers to which the evidence needs to be presented.

Furthermore, the visualisation of changes that occur in the free space of a decrypted cover volume reveals patterns in the hidden volume which when combined with an understanding of file systems, can be used to infer information about the hidden volume. This is demonstrated by estimating the type and size of the hidden volume assuming a FAT based file system.

3.2 Obtaining Multiple Copies of an Encrypted Container

This section discusses how multiple versions of an encrypted container can be obtained by exploiting the Volume Shadow Copy feature of Windows Vista and Windows 7. This functionality extends the Restore Point feature of Windows XP so that backups are now created not just of important system files but also of user created files [11]. Shadow copies are not created every time a file is changed but when a Restore Point is created. In Windows Vista "restore points are created automatically every day, and just before significant system events such as the installation of a program or device driver" [12]. This means that previous versions of users' files may be available in addition to the current instance.

Forensic acquisition of data from Windows Vista Restore Points is discussed in detail in [13], but the process can be summarised as:

- Booting the suspect system as a clone or virtual machine
- Listing available Restore Points using the command line
- Mounting the restore points using symbolic links at the command line
- Copying out the mounted restore points to blank media

This creates copies of all files from a particular Restore Point. However, once a clone or virtualised version of the suspect system is booted, it is also possible to use the system's user interface to access previous versions of particular files of interest.

Experiments have shown that TrueCrypt encrypted containers are included in these automatic backups, but are not available through the GUI in the usual manner (shown in Figure 2). However, previous versions are available by examining previous versions of the folder in which the containers are stored (also shown in Figure 2). These folders can be restored and the multiple versions of the encrypted containers extracted.

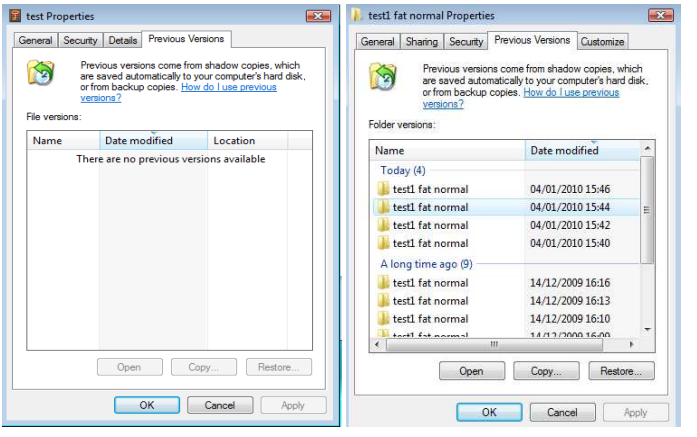


Fig. 2. While there are no Previous Versions visible for a TrueCrypt container (left), they can be accessed by examining previous versions of the containing folder (right)

Therefore, using this technique it may be possible to recover older copies of an encrypted container from a single disk image through the Previous Version functionality of Windows Vista and Windows 7. These multiple versions can then be examined further.

3.3 Visualising Changes

The previous section showed how multiple copies of an encrypted volume can be recovered from a single disk image due to the Volume Shadow Copy feature of Windows Vista and Windows 7. This section describes the further examination of these extracted encrypted volumes.

Multiple extracted encrypted volumes can be viewed using hex editors such as WinHex which offer synchronise and compare functionality. This allows multiple volumes to be compared and the differences highlighted. However, even if differences between two encrypted volumes are seen towards the end of the volumes, from this alone it is not possible to determine if these changes are due to a hidden volume or simply data being written near the end of a standard encrypted volume. However, assuming that some keys are available – either discovered during an investigation or provided as a result of an order placed under disclosure legislation - then decrypted versions of these volumes will be available, albeit they may not be the 'true' contents of the volume. If the volumes are decrypted and then examined in WinHex, since the file system can be interpreted it can be determined if random data is changing in the

free space of the volume (Figure 3). As discussed in [9], if multiple versions of a volume can be examined then existence of the hidden volume is undeniable since "seemingly random bytes in the hard drive will change".

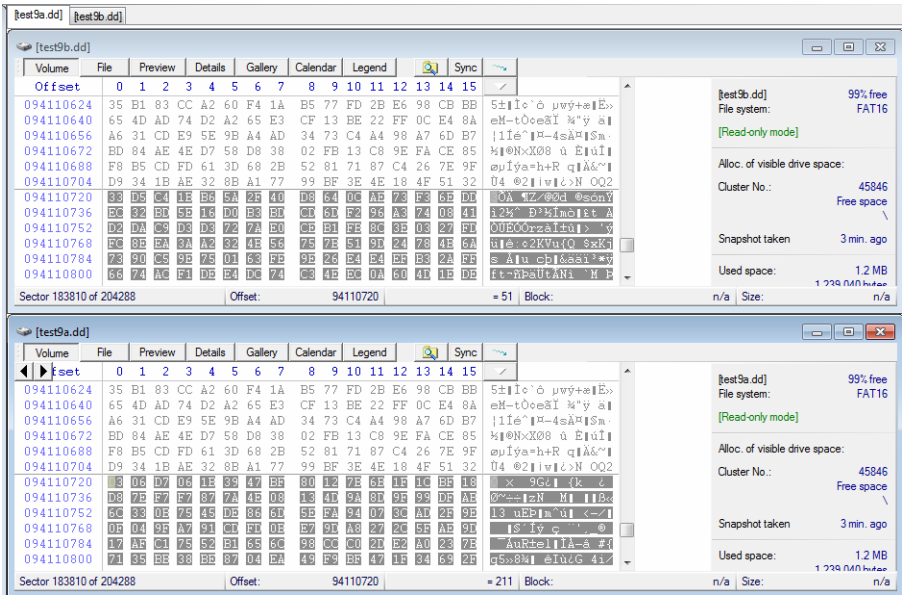


Fig. 3. WinHex highlighting the differences in the free space of two decrypted volumes

While WinHex reports that the changes are occurring in free space of the volume, the nature of the changes is clearer if the volume is examined from a broader perspective; therefore the entire volume is visualised within a fixed space (a 100 x 100 grid). Blocks that are different between versions are highlighted. This is implemented in Python using the standard TkInter graphics library. Free space on the volume is identified using a simple entropy based test for randomness and is differently coloured; also, changes between the versions of the volume, and any changes since the first volume are highlighted. These are shown in Figure 5 and it can be clearly seen that changes are occurring in the free space of the volume.

The visualisation allows the changes to be clearly identified as being in the free space of the volume which indicates the presence of a hidden volume. This visualisation is useful not only for the analyst but also if this information needs to be presented to a court. Visualising changes between different versions of decrypted cover volumes in this way also reveals patterns in the changes to the hidden volume, from which additional information can be inferred.

3.4 Identifying the Size of the Hidden Container

Through visualisation of the decrypted volumes it is possible to infer additional information about the hidden volume. However, it is first necessary to discuss file system structures.

A FAT based file system stores data in clusters, which are the smallest unit of space on a volume which can be allocated to store data, typically 4096 bytes on a modern system. There are a number of variations of the FAT file system including FAT16 and FAT32 [14][15]. FAT file systems store data in a hierarchical structure consisting of directories that contain files and other directories, as are commonly viewed through operating systems. However, the directory entry contains only information about the file; the file content is saved elsewhere on the volume; although the directory entry does contain a pointer to the first cluster in which the file is stored.

Since a saved file may be larger than the size of a cluster, a file may occupy multiple clusters, which may not be contiguous. The file system therefore maintains a record of the 'cluster chains', i.e. given a start cluster it is possible to look up its entry and determine which cluster (if any) should be read next. These records of cluster chains are stored in a File Allocation Table (FAT), which contains an entry for every cluster on the volume. Since the FAT is essential to reading data from the volume, two copies of it are stored, both near the beginning of the volume, one after each other. The size of each entry in the FAT depends on the file system in use; in FAT16 each entry is 16 bits (2 bytes) and in FAT32 each entry is 32 bits (4 bytes).

Therefore, when a file is written to a volume, the FATs are updated to indicate that the cluster(s) in which the file is stored are no longer free. It is this property that is used to extract additional information from the hidden volume.

Since the FATs contain an entry for every cluster in the volume, the size of a FAT is proportional to the size of the volume. Therefore, if the size of the FAT can be determined, so can the size of the volume. Since the FATs are located near the beginning of a volume and follow each other, and since data written to a volume causes both FATs to be updated, it is possible to visually identify candidates for the two FATs; the two FATs are separated by a consistent value and change when data is written to the disk. This is can be seen in Figure 5.

Since the two FATs follow each other, the size of the FAT can be estimated from the difference between the two FATs. A simple example is shown in Figure 4.

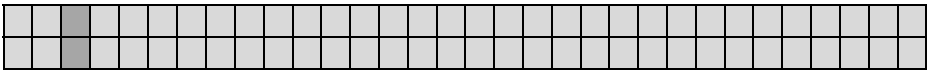


Fig. 4. A simple example highlighting the differences between two volumes

In this example each block represents two bytes. If we assume that the two identified changes are in FAT 1 and FAT 2 then the size of the FAT is the difference between them (32 blocks = 64 bytes). Since the FAT represents all the clusters on the volume, from the size of the FAT it is possible to estimate the size of the volume. It is possible only to estimate the size of the FAT since the end of the first FAT may not align with a cluster boundary and is therefore padded. Also, since encryption is performed in sector sized blocks, changing a single bit in a sector results in the entire block changing and it is therefore not possible to determine the exact position of data changed within a block.

This idea is demonstrated by analysing a hidden volume within a cover volume, both created using TrueCrypt. In the following example a 42 Megabyte (43909120

bytes) hidden volume is created within a 100 Megabyte cover volume. Data are written to the hidden volume and a copy of the volume is made with each change. The visualisation of the changes is shown in Figure 5.

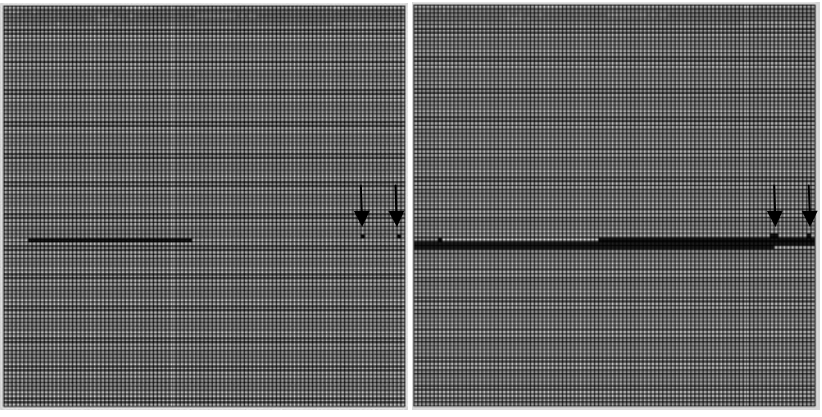


Fig. 5. A visualisation of the changes made to the test volume with the two FATs highlighted. The later contiguous blocks of changing data are the actual file contents being written to the volume.

The visualisation software allows the hypothesised positions of the two FATs to be marked and the size of the hidden volume to be estimated, as described above. However, in order to calculate the size of the hidden volume additional information is necessary.

Remembering that the FAT is an index of all the clusters on the disk, by matching up changes in the FAT with changes they represent in the data area of the hidden volume, it is possible to calculate the ratio between the size of the FAT entries and the size of the data that they index. For example in Figure 6, F_1 and F_2 represent points in FAT1 and using the visualisation tool it is possible to see the corresponding changes in the data area, marked D1 and D2. The visualisation tool can be used to identify the offsets of these:

$F_1 = 60556267$	$F_2 = 60557047$	$F_{diff} = 780$
$D_1 = 60743662$	$D_2 = 61147117$	$D_{diff} = 403455$

Therefore, 780 bytes in the FAT represents 403455 bytes of data, meaning 1 byte in the FAT represents approximately 517.25 bytes of data. Since each entry in the FAT represents one cluster and the default cluster sizes are powers of 2 (e.g. 512, 1024, 2048... bytes) [16], the actual ratio of FAT bytes to data bytes is 512.

The size of the FAT can be estimated using the visualisation; in this case the difference between the entries in the two FATs is approximately 85kB (see Figure 7, below). The size of the volume can be calculated by multiplying the size of the FAT by the ratio of FAT bytes to data bytes (85514 * 512 = 43783168 bytes).

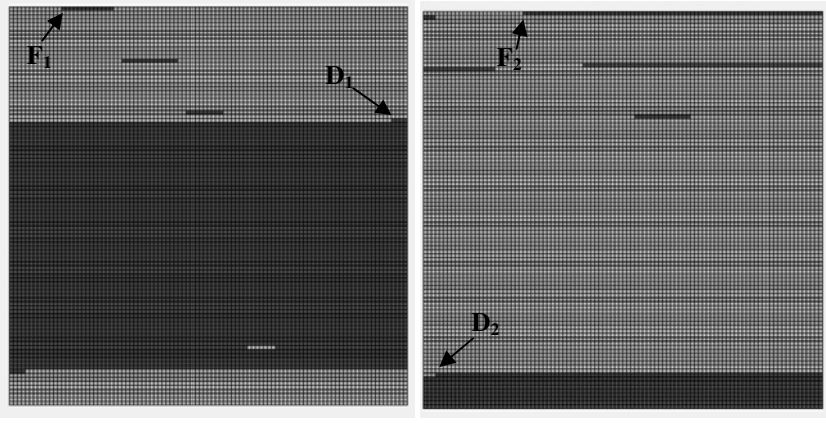


Fig. 6. A visualisation of the changes made to the test volume with points in the hidden volume's FAT and data area marked

In addition to the size of the volume it is also possible to determine the FAT version (FAT16 or FAT32) and the cluster size in use. Since FAT16 and FAT32 have 2 and 4 bytes per FAT entry respectively, there are therefore either 42757 (FAT size/2) or 21378.5 (FAT size/4) FAT entries.

Since there are the same number of FAT entries as there are clusters, the possible cluster sizes can be calculated as volume size divided by the possible number of clusters, giving either 1024 (43783168/42757) or 2048 (43783168/21378.5). From [16] it is possible to determine which of these two possibilities are valid for the volume size identified; in this case it is FAT16 with a cluster size of 1024 bytes.

```
FAT1 offset: 60556276
FAT2 offset: 60641790
bytes difference: 85514
total no of clusters:   42757
estimated volume size: 43783168 bytes
```

Fig. 7. Successful calculation estimating the size of the hidden volume embedded in a decrypted cover volume (actual size is 43909120 bytes)

4 Evaluation

While the previous section has shown that the existence and the size of the hidden volume can be determined there are limitations to this approach.

In this paper, the multiple copies of volumes have been obtained using the Volume Shadow Copy feature of Windows Vista and Windows 7. Firstly this is only an option for encrypted containers and cannot be used for encrypted volumes or disks. Also, if the System Restore feature is turned off then these will not be available; although this obviously removes all the benefits that System Restore provides. However, the developed visualisation technique does not necessarily require Shadow Copies of encrypted volumes and multiple versions of the volumes from any source can be used e.g. from

external backups. Therefore, if external backups are available then the visualisation technique could also be used with full volume or full disk encryption.

By visualising the changes that occur in the free space of a decrypted cover volume and using knowledge of file system structures is has been possible to identify the FATs, estimate the size of the hidden volume and derive the FAT version in use and the cluster size. In future it should also be possible to identify hidden volumes formatted using NTFS although the patterns of changes are different due to the use of a Master File Table rather than FATs.

It should be emphasised that this technique is dependent on identifying patterns in the underlying file system. While this is possible for the current version of TrueCrypt there are other steganography encryption systems where this would not be possible, e.g. Rubberhose [17] which does not store the file system in a linear manner.

The value of this work to a forensic investigation is that this technique allows investigators who believe they have recovered encryption keys to assess the likelihood of deliberate deception. In the UK there is legislation in place to encourage suspects to provide decryption keys; therefore demonstrating the existence of a hidden container is sufficient to allow non-technical measures to be used to further an investigation. Also, the estimation of the size of a volume can be used to give an idea of how much data could be hidden; while this not essential for a RIPA prosecution, it may add weight to a case against a defendant for refusing to supply decryption keys.

5 Future Work

In addition to estimating the size of the volume, since the Shadow Copies have associated dates and times, it may be possible to determine the amount of data written to a hidden volume between two dates. This may then be correlated with other sources of digital evidence, for example records from Internet Service Providers.

This visualisation technique can be extended to examine other file systems including NTFS, EXT3 and HFS+ and it may be possible to identify other information about the hidden volume in addition to determining the size of the hidden volume.

It is also desirable to determine additional information on FAT file systems, for example the position of the boot sector. This is particularly useful since it is used in known-plaintext-based decryption key recovery approaches such as [18].

6 Conclusions

There are several approaches to addressing the problem of encrypted evidence, and in the UK, legislation has been passed to encourage decryption keys to be provided by the suspect. However, there are technical measures that can be employed to counter this legislative approach, including the use of hidden volumes. This paper has demonstrated how these technical measures can be overcome by acquiring multiple copies of an encrypted container from a single disk image using the Volume Shadow Copy feature of Windows Vista and Windows 7. It has also shown how these multiple copies can be used to detect the presence of hidden volumes within a standard encrypted volume. A visualisation of the volume and the changes to that volume can be used to

infer additional information about the hidden volume, for example to estimate its size. Demonstrating the existence of hidden volumes is useful in an investigation since it allows investigators who believe they have recovered encryption keys to assess the likelihood of deliberate deception, potentially motivating further non-technical investigative measures.

References

1. Casey, E.: Practical Approaches to Recovering Encrypted Digital Evidence. International Journal for Digital Evidence 1 (2002)
2. Wolfe, H.B.: Encountering Encrypted Evidence (Potential). In: Proceedings of the 4th Conference on Information Technology Curriculum (2002)
3. Wolfe, H.: Encountering Encryption. Computers and Security 22, 388–391 (2003)
4. Wolfe, H.: Penetrating Encrypted Evidence. Digital Investigation 1, 102–105 (2004)
5. United Kingdom: Regulation of Investigatory Powers Act 2000. HMSO (2000)
6. Home Office: Investigation of Protected Electronic Information: Code of Practice (2007)
7. TrueCrypt: TrueCrypt Documentation (2009), http://www.truecrypt.org/docs/
8. TrueCrypt: TrueCrypt Documentation: Hidden Volume (2009)
9. Czeskis, A., Hilaire, D.J.S., Koscher, K., Gribble, S.D., Kohno, T., Schneier, B.: Defeating encrypted and deniable file systems: TrueCrypt v5. 1a and the case of the tattling OS and applications (2008)
10. Craiger, J.P., Pollitt, M., Swauger, J.: Law Enforcement and Digital Evidence (2005), http://ncfs.org/craiger.delf.revision.pdf
11. Microsoft: Learn about the features: Shadow Copy (2007), http://www.microsoft.com/windows/products/windowsvista/features/details/shadowcopy.mspx
12. Microsoft: System Restore: frequently asked questions (2008)
13. Titheridge, D.: Microsoft Windows Vista Registry. MSc. Cranfield University (2009)
14. Carrier, B.: File System Forensic Analysis. Addison-Wesley, Reading (2005)
15. Sammes, T., Jenkinson, B.: Forensic Computing: A Practitioners Guide, 2nd edn. Springer, Heidelberg (2007)
16. Microsoft: Default cluster size for NTFS, FAT, and exFAT (2009), http://support.microsoft.com/kb/140365
17. Assange, J., Weinmann, R.P., Dreyfus, S.: Rubberhose, http://iq.org/~proff/marutukku.org/ (Undated)
18. Hargreaves, C., Chivers, H.: Recovery of Encryption Keys from Memory Using a Linear Scan. In: The International Workshop on Digital Forensics, Barcelona, Spain (2008)

Tor HTTP Usage and Information Leakage

Markus Huber[1], Martin Mulazzani[2], and Edgar Weippl[1]

[1] Vienna University of Technology, Austria
mhuber@sba-research.org,
weippl@ifs.tuwien.ac.at
[2] Secure Business Austria, Austria
mmulazzani@sba-research.org

Abstract. This paper analyzes the web browsing behaviour of Tor users. By collecting HTTP requests we show which websites are of interest to Tor users and we determined an upper bound on how vulnerable Tor users are to sophisticated de-anonymization attacks: up to *78 %* of the Tor users *do not* use Tor as suggested by the Tor community, namely to browse the web with TorButton. They could thus fall victim to de-anonymization attacks by merely browsing the web. Around 1% of the requests could be used by an adversary for exploit piggybacking on vulnerable file formats. Another 7 % of all requests were generated by social networking sites which leak plenty of sensitive and identifying information. Due to the design of HTTP and Tor, we argue that HTTPS is currently the only effective countermeasure against de-anonymization and information leakage for HTTP over Tor.

Keywords: Tor, information leakage, privacy.

1 Introduction

The Tor network [1] is a widely deployed anonymization network which hides the user's IP address on the Internet. It is expected to be used by hundreds of thousands of users every day and is the most heavily used open anonymization network today [2]. The main contribution of this paper is an in-depth analysis on how the Tor network is used to browse the web.

At the time of writing little is known about the traffic that leaves the Tor anonymization network. McCoy et al. [3] published a first investigation into how Tor is being used by its end-users. They found that the majority of connections leaving the Tor network are caused by the hypertext transfer protocol (HTTP). According to statistics from the Tor project [4], their anonymization network played a crucial role in the aftermaths of the latest Iranian elections whereas HTTP-based services such as Facebook had been blocked on a national-level. The ability of Tor to bypass Internet-censorship techniques got recently even the attention of mainstream media (e.g. Forbes [5]). A deeper analysis of HTTP traffic that is tunnelled trough the Tor network is however lacking. Hence we aim to provide a first investigation into how Tor is used to access the world wide web.

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 245–255, 2010.

By running a Tor exit server and logging the HTTP requests, we collected in total 9 x 10^6 HTTP requests over a period of several weeks. The captured HTTP requests form the basis of our investigation into the web browsing behaviour of Tor users. The main contributions of this paper are:

- An in-depth analysis of HTTP traffic tunnelled through the Tor network (Section 4).
- HTTP-based attack scenarios and mitigation strategies (Section 5).
- Potential risks for complete HTTP user de-anonymization, which can not be prevented by Tor (Section 5.1).

We additionally show dangers to which users are exposed by using Tor incorrectly and which information users leak by browsing the web with Tor.

The rest of the paper is organized as follows: Section 2 discusses Tor and previous work about Tor usage analysis. Section 3 shows how we collected our data while we interpret the results in Section 4. Section 5 describes new attack scenarios based on HTTP as well as countermeasures. The last section is dedicated to the conclusion.

2 Tor Security and Threat Model

The Tor network hides the user's IP address by sending its packets through a number of Tor servers. Tor itself does not hide nor encrypt communication content leaving the Tor network: the user has to take care that it is used correctly. Sensitive information should only be sent over an encrypted protocol such as HTTP secure (HTTPS). A passive adversary running an exit server would need to break the end-to-end encrypted transmission in order to capture sensitive information. We will show later to what extent sensitive information is transmitted unencrypted over HTTP.

The basic infrastructure of Tor is run by volunteers, and anyone can set up a relay at relatively low cost. It provides reliable, bi-directional communication that can be used for low latency communication such as interactive conversations, shell access or web browsing. Tor can be used by any program that is able to use a SOCKS proxy and is freely available for almost any operating system as well as in the form of prepared live CDs and virtual machines. Tor uses three servers per path by default to hide its users real IP address, all servers chosen at the client-side. The user's communication content is protected from eavesdropping within the Tor network by using multiple layers of encryption. Before forwarding a message, every Tor server removes his layer of encryption.

At the time of writing, the Tor network consists of approximately 1600 running servers, distributed all over the world. The first server in a path is chosen from an ordered set of so called "entry nodes"; these servers are able to see the users real IP address. Without the ordered set, every entry node would eventually see every user's real IP address. The last Tor server in the path is the so called "exit server" and is chosen based on the communication target's port number and self-proclaimed available bandwidth. The so called "exit policy" at every

server specifies which port numbers are allowed for communication and whether the server is allowing connections only within Tor or leaving Tor to the regular Internet. The exit policy is defined by the server operator. Finally, a small set of trusted "directory servers" collect information about the current state of the network and vote on the current set of reachable servers. This information is publicly available to all clients. The security of the Tor network relies on the fact that instead of a single point or entity a user has to trust (for example by using an open proxy server or a dubious VPN service); the trust is distributed among the three Tor relays in a path.

Previous research about the usage of Tor has been conducted in the beginning of 2008 [3]: by running a high bandwidth Tor relay and inspecting the communication content it was found that the majority of connections leaving Tor was created by HTTP traffic, in total more than 90 %. However, a disproportional part of the transferred data was caused by BitTorrent, a common file sharing protocol. Yet a detailed analysis of the HTTP usage has not been conducted. Another analysis has been conducted by Dan Egerstad in 2007 [6] who published a list of 100 sensitive email accounts including passwords from embassies that apparently used Tor incorrectly. Other published attacks on Tor aimed at decreasing or defeating the users anonymity by means of traffic analysis [7,8,9,10] as well as attacks on unique aspects such as path selection [11] or the "hidden services" of the Tor network [12,13].

3 Tor HTTP Sniffing - Ethics and Methodology

In our experiment we ran a Tor exit node and collected all HTTP requests by running *urlsnarf* from the *dsniff* toolkit [14]. Urlsnarf sniffs HTTP traffic and is able to format it in CLF (Common Log Format), which is a format commonly used by webservers. Compared to other experiments [3] our server was advertising less bandwidth to represent an average node and not to bias the client's Tor path selection algorithm towards our node; only HTTP traffic was allowed in our exit policy. The collection period was from December 2009 till January 2010 with 9 x 10^6 HTTP requests in total resulteding in a logfile of 2.5 gigabytes. We took special care that the users identities were not endangered or revealed during the collection process: we did not store any packet dumps, user credentials, authentication cookies or any other possibly compromising data except the HTTP request. We did not try to become a "guard server" which clients use as their entry to the Tor network and which are able to see the users real IP address, and we did not monitor incoming connections. The data was stored in an encrypted filecontainer and moved to another computer after the collection process to protect against data disclosure in case of a server compromise.

A Python script using the "apachelog" libary [15] deconstructed the requests into three parts, according to our evaluation criteria:

– *Target domain*: the domain name of the request, without any deeper analysis of the specific requested file or session context.

- *User agent*: the string that the browser sends to identify itself. This gives a hint if the users are using TorButton, the recommended software by the Tor developers to prevent user identification by an adversary.
- *File type*: the extension of the requested file. Executables pose a direct danger to the user as an adversary might replace or modify them to defeat anonymity or even worse. Indirect danger comes from file formats were there exist known vulnerabilities in the corresponding software.

Subsequently the requests were sanitized and normalized for evaluation:

- File formats: various file extensions are treated the same on the client side, notably image file formats such as `jpg` and `jpeg` or websites within the browser, such as `html`, `htm`, `php` or `cgi`.
- Domain affiliation: some websites use different domains for content distribution, such as for example `fbcdn.net` and `facebook.com` belong to the same website.

4 Results and Data Interpretation

Our goal was to analyze what Tor is used for, which domains are most popular among Tor users and to discover potential threats to users.

4.1 Domains

In a first analysis based on the collected HTTP traffic we looked into depth into the different websites that were visited through our Tor exit server.

Table 1a shows the top 10 visited domains, based on the percentage a certain domain accounts to the total requests. Facebook.com and google.com were amongst the most visited sites. In fact, the majority of the requests belonged to one of the following website categories:

- *Social networking* sites (e.g. facebook.com, blackplanet.com)
- *Search engines* (e.g. google.com, yellowpages.ca)
- *File sharing* (e.g. btmon.com, torrentbox.com)

Social networking sites (SNSs), which today account to the most popular web sites, account in total to 7.33 per cent of all analysed HTTP traffic. These webservices are interesting for two reasons: SNSs leak plenty of personal information and secondly these services do not support HTTPS at the time of writing, except for user authentication. Table 1b shows the Top SNSs as well as how much of the SNSs traffic accounts to which service.

4.2 Fileformats

Among all the collected HTTP GET requests, `.html` was predominant with almost 54 % of all the requests. Around 32 % were caused by image formats, followed by JavaScript with 4.25 % of all GET requests. The details of the top 10 requested file extensions are shown in table 2.

Table 1. Analysis of most accessed domains

(a) Overall services

Domain (total %)	
URL	Per cent (%)
'facebook.com'	4.33
'www.google.com'	2.79
'blackplanet.com'	1.61
'yandex.ru'	1.57
'btmon.com'	1.47
'photobucket.com'	0.98
'craigslist.org'	0.90
'torrentbox.com'	0.88
'ameba.jp'	0.87
'maps.google.com'	0.70

(b) Social Networking Sites

Social Networking Site (relative %)	
Name	Per cent (%)
'facebook.com'	59.06
'blackplanet.com'	21.94
'vkontakte.ru'	5.61
'tagged.com'	4.95
'orkut.com'	3.13
'myspace.com'	2.36
'mixi.jp'	1.54
'hi5.com'	0.48
'adultfriendfinder.com'	0.47
'badoo.com'	0.46

Table 2. Top 10 file formats

Fileformat		
Extension	Description	Per cent (%)
'html'	HyperText Markup Language	53.83
'jpg'	JPEG image	18.15
'gif'	Graphics Interchange Format (GIF)	11.43
'js'	JavaScript	4.25
'css'	Cascading Style Sheets (CSS)	3.03
'png'	Portable Network Graphics	2.81
'xml'	Extensible Markup Language	2.62
'ico'	ICO image	0.77
'swf'	Shockwave Flash file	0.48
'rar'	RAR archive file format	0.20

4.3 Web Browser Types

Web browsers submit a text string known as "user agent string" to servers with details on the client version and the operating system they are running on. When looking at the browser user agent string, various browser and operating systems combinations were seen. The browser used to access websites through the Tor network plays a crucial role for the anonymity of the end-user. TorButton [16] is a plugin for the Mozilla Firefox browser developed by Mike Perry and Scott Squires, which makes it possible for the user to switch between using the Tor network and browsing the web directly. Even more important, it disables many types of active content which could be used to bypass Tor and thus defeat Tor's anonymity protecting methods. To begin with, we inspected which were the ten

Table 3. Top10 Browser (Raw user agent string)

Browser	
Full User Agent String	Per cent (%)
'Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.0.7) Gecko/2009021910 Firefox/3.0.7'	18.86
'-'	4.48
'Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9) Gecko/2008052906 Firefox/3.0'	2.71
'Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.16) Gecko/20080702 Firefox/2.0.0.16'	1.81
'Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)'	1.66
'Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)'	1.64
'Mozilla/5.0 (Windows; U; Windows NT 6.0; en-US; rv:1.9) Gecko/2008052906 Firefox/3.0'	1.59
'Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)'	1.50
'Mozilla/5.0'	1.34
'Opera/9.63 (Windows NT 5.1; U; en) Presto/2.1.1'	1.31

most common user agent strings. Table 3 shows the top 10 browser user agent strings within our experimental data.

TorButton uses a more constant user agent string in contrast to Firefox since the Firefox user agent string changes with every update of Firefox. This increases the anonymity set, as long as the used string is plausible and the used version is still in widespread use. By an analysis of the TorButton source code we identified nine distinct user agent strings that have been used by the different versions of TorButton. We found that solely 22 % of all the user agent strings matched one of the nine user agent strings used by the TorButton Firefox extension. Hence we argue that at least 78 % per cent of all traffic originated from a browser without the TorButton extension. It remains unclear if the users take additional preventive measure when using other browsers, however it seems unlikely that the majority of the non-TorButton users enforced all required browserside anonymity protection measures. Table 4 shows the Top 10 user agent strings, whether the user agent strings were used by TorButton and the revision of the TorButton source code. Furthermore the table shows the user agent strings and operating systems versions in a more readable format.

Table 4. Top 10 Browser (interpretation)

Browser		
Version	OS	Per cent (%)
TorButton > svn:r18954 (Firefox/3.0.7)	-	18.86
-	-	4.48
Firefox/3.0 (Gecko/2008052906), en-US	Windows XP	2.71
TorButton > svn:r16066 (Firefox/2.0.0.16)	-	1.81
Internet Explorer 6.0 SV 1.0	Windows XP	1.66
Internet Explorer 6.0	Windows XP	1.64
Firefox/3.0 (Gecko/2008052906)	Windows Vista	1.59
Internet Explorer 6.0	Windows 2000	1.50
Mozilla/5.0	-	1.34
Opera 9.63 (Presto/2.1.1), en	Windows XP	1.31

5 Further Tor Exit Server Attack Scenarios

A malicious exit server has many different options to gain knowledge of the Tor users, active as well as passive. Many HTTP requests leak information of various kinds. Additional information can be gained by active content insertions and modifications.

5.1 Information Leakage of Sensitive Information

Many HTTP requests leak information, sometimes even sensitive information. Even if the malicious exit server is unable to identify the originator of the request, it is able to gain knowledge only by watching the requests passively.

- *Search query strings:* Search queries are often submitted to a search engine via a HTTP GET request. If a Tor user searches information about e.g. a special disease, location information about a hotel, how to get to a certain place, or a recent political event, this gives hints about the identity, location or nationality of the user. Additional information can be deduced by language, browser and operating system to further reduce the anonymity set. This theoretical risk has also become evident in practice by the incident with AOL's release of search queries [17].
- *Social networks:* As described above social networking sites accounted for more than 7 per cent of the all the HTTP traffic captured by our Tor exit server. In the case of Facebook as well as other popular SNSs, HTTP requests include the user-id in plaintext. Because users often represent their realworld persona, SNSs users can easily be identified. The social graph of a user could furthermore reveal the identity of a SNS user. Thus an analysis of a user's id and corresponding social graph could completely deanonymize the Tor user. This is especially dangerous as many Tor users apparently use social networks.
- *Localization information:* Sent by a misconfigured browser reduces the anonymity set considerably. TorButton normalizes all user to use "en-us" within the browser user agent. This increases the size of the anonymity set for users from the US, however decreases the size for the rest of the world. Other localization information can be retrieved from toplevel domains as we suspect that many users e.g. browse to their localized version of the Google search engine (google.ca, google.dk, ...) instead of the normalized google.com.
- *Other:* Sensitive and possibly incriminating content will be transmitted without proper usage of Tor, e.g. authentication cookies. However, as these are not transmitted in the HTTP GET request, we did not include them in our analysis. A motivated attacker surely will harvest those information as well, thereby possibly increasing the chance of defeating anonymity of a certain Tor users.

As an example we will use a search query on google maps, asking for driving instructions from Times Square to Central Park in New York. As it can be seen, plenty of parameters are part of the request, and of course the exact address coordinates as well:

```
http://maps.google.com/maps?f=d&source=s_d&saddr=times+square ->
  &daddr=central+park&hl=en&geocode=&mra=ls ->
  &sll=40.771133,-73.974187&sspn=0.053106,0.111494 ->
  &g=central+park&ie=UTF8&t=h&z=16
```

5.2 Script Injection

Dynamic website content such as e.g. AJAX or Flash is hard to filter: either all dynamic content is blocked which results in poor usability, or dynamic content is allowed which opens the door for exit servers injecting malicious scripts. It has already been shown that the insertion of invisible iframes and Flash can defeat anonymity [18]. By injecting JavaScript within an invisible iframe it was possible to further reduce the requirements on the client to become vulnerable to this attack [19]. The authors even showed that their attack is feasibly by modifying the HTML meta refresh tag of a website, so without JavaScript.

By manipulating HTTP communication content, phishing attacks become possible if no communication encryption or verification is used. Instead of sending the data to the destination, a malicious exit server might save the data and present an error page.

5.3 File Replacement

There exist many file formats commonly used on the Internet which could be used by an adversary to execute malicious code as the corresponding software handling the files has well known weaknesses. Among all the HTTP requests we have seen, up to 97993 requests, or 1% of the total requests were for files with known vulnerabilities and publicly available exploits. As we did not inspect the communication content and the transferred files itself, it remains unclear if the files could have been used for infecting the client or not. Instead, it can be seen as an upper bound for a new possible and yet not documented infection vector.

- executable (5590 requests): `.exe` files could get replaced or malicious code could be appended to the executable. This could result in remote code execution on the client side and could among other things be used to reveal the users identity.
- PDF (1619 requests): vulnerabilities in Adobe Reader and Acrobat as well as alternative PDF viewers could be used to execute malicious code in `.pdf` files.
- Office (400 requests): Microsoft Office has many documented vulnerabilites. The requests we monitored were for `.doc`, `.xls` and `.ppt` files.
- Mediafiles (23821 requests): Several Mediaplayer like e.g. Apples Quicktime, the Windows Media Player or VLC have documented flaws that are exploitable by manipulated media files. We encountered various requests that could be used for exploit piggybacking: `.avi` (10105 requests), `.wmv` (6616 requests), `.mp3` (4939 requests) or `.mpg` (1138 requests).

– other file formats (66563 requests): compressed file archives like `.zip` (9937 requests) and `.rar` (16512 requests) might get used by an attacker as they can be used to exploit vulnerabilities in the software itself; but also to add, manipulate or replace files within those archives. Shockwave flash files `.swf` (40114 requests) account for a major proportion of vulnerable file formats and could be used for client infection as well.

These vulnerabilities could be used to massively compromise anonymity on a large scale by exploiting vulnerable software, using Tor for cheap "man in the middle" attacks and even creating a botnet of Tor clients. However, it has to be noted that this is not a flaw of Tor but of rather an inappropriate use of it. Clients should verify the integrity of their files when using unencrypted communication.

5.4 Countermeasures and Discussion

It is hard if not even impossible to prevent information leakage in HTTP requests, as the requests are often transmitting information of a certain user's context. The collection and aggregation of webservice specific information is non-trivial, but we showed that a great amount of information can be gathered by a passive adversary. In the following we briefly outline three methods that can help to mitigate security and privacy risks caused by a malicious Tor exit server:

Detection of malicious exit servers. The most straightforward solution would be to detect bad exit servers and ban them from the Tor exit server list. McCoy et al. [3] proposed to use reverse DNS lookups in order to detect exit servers that run packet analyzer software with a host resolution feature. A complete passively adversary is almost undetectable. TorFlow is an application developed by Mike Perry [20] which supports exit server integrity checks. Hereby the basic principle is that: a reference file is used as a sample and downloaded through different Tor exit servers, cryptographic checksums are used afterwards to check if a manipulation occurred. The basic idea works fine on files like binaries or static websites, dynamic content however is much harder to verify. For this reason, dynamic content is blocked by TorButton instead of analyzed for maliciousness.

User education. The Tor network offers reliable anonymity in case if properly used. Awareness-campaigns as well as end-user education can help to ensure that people use Tor always in combination with TorButton as well as take special care of which services they use through Tor. The incident with the embassy mail accounts [6] has shown what might happen if Tor is used incorrectly, even seriously harming privacy instead of preventing de-anonymization and obfuscating user activity. Active content (e.g. Macromedia Flash) of all kind should be avoided if possible, and using trusted exit nodes could further reduce the risk of data leakage.

HTTPS. The only solid protection of user data would be the use of strong encryption such as secure HTTP (HTTPS). The usage of the HTTPS protocol is unfortunately not always possible as many website operators do not support it, e.g. https://www.google.com redirects the user to http://www.google.com. At the time of writing the great majority of social networking providers fail to support HTTPS.

6 Summary and Conclusion

By collecting 9 x 10^6 HTTP requests we observed that up to 78 % of the Tor users browse the Internet without TorButton, the recommended software by the Tor community. The majority of users is therefore possibly vulnerable to sophisticated deanonymization attacks by an exit server, e.g. by injecting invisible iframes or scripts. 1 % of the requests were for vulnerable file formats, which could be used for exploit piggybacking. Even if the adversary running an exit server is completely passive, without modifying or redirecting communication content, an uncountable amount of sensitive information like search queries or authentication cookies are leaked. Furthermore, 7 % of all analysed HTTP connections were created by social networking services which leak plenty of personal information. To protect the Tor users, various tools like TorButton or TorFlow have been proposed and implemented. However, the only effective countermeasure at the time of writing is the use of end-to-end cryptography, namely HTTPS.

Future research in this area can be done to quantify the amount of sensitive information observable by an passive adversary. It has to take into account the different requirements for anonymization and/or censorship circumvention by the users. Additional research is needed on protection measures.

References

1. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: 13th USENIX Security Symposium, San Diego, CA, USA (August 2004)
2. Tor: anonymity, https://www.torproject.org/
3. McCoy, D., Bauer, K., Grunwald, D., Kohno, T., Sicker, D.: Shining Light in Dark Places: Understanding the Tor Network. In: Borisov, N., Goldberg, I. (eds.) PETS 2008. LNCS, vol. 5134, pp. 63–76. Springer, Heidelberg (2008)
4. Tor Project. Measuring Tor and Iran (2009), https://blog.torproject.org/blog/measuring-tor-and-iran
5. PluggedIn, F.: Web tools help protect human rights activists (2009), http://www.forbes.com/feeds/afx/2009/08/19/afx6794971.html
6. Egerstad, D.: DEranged gives you 100 passwords to Governments & Embassies (2007) (retrieved 2009-12-20)
7. Raymond, J.F.: Traffic analysis: Protocols, attacks, design issues, and open problems. LNCS, pp. 10–29. Springer, Heidelberg (2001)
8. Mathewson, N., Dingledine, R.: Practical traffic analysis: Extending and resisting statistical disclosure. In: Martin, D., Serjantov, A. (eds.) PET 2004. LNCS, vol. 3424, pp. 17–34. Springer, Heidelberg (2005)

9. Hopper, N., Vasserman, E.Y., Chan-Tin, E.: How much anonymity does net work latency leak? In: Proceedings of the 14th ACM conference on Computer and communications security, p. 91. ACM, New York (2007)
10. Murdoch, S.J., Danezis, G.: Low-Cost Traffic Analysis of Tor. In: SP '05: Proceedings of the 2005 IEEE Symposium on Security and Privacy, Washington, DC, USA, pp. 183–195. IEEE Computer Society, Los Alamitos (2005)
11. Bauer, K., McCoy, D., Grunwald, D., Kohno, T., Sicker, D.: Low-resource routing attacks against Tor. In: Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2007), Washington, DC, USA (October 2007)
12. Murdoch, S.J.: Hot or not: Revealing hidden services by their clock skew. In: Proceedings of the 13th ACM conference on Computer and communications security, p. 36. ACM, New York (2006)
13. Øverlier, L., Syverson, P.: Locating hidden servers. In: Proceedings of the 2006 IEEE symposium on Security and Privacy. IEEE CS, Los Alamitos (May 2006)
14. dsniff, http://www.monkey.org/~dugsong/dsniff/
15. apachelog, http://code.google.com/p/apachelog/
16. Perry, M., Squires, S.: Torbutton, https://www.torproject.org/torbutton/
17. Barbaro, M., Zeller, T.: A face is exposed for AOL searcher no. 4417749. New York Times, 9:2008 (2006)
18. Christensen, A.: Practical Onion Hacking: finding the real address of Tor clients (2006), http://www.fortconsult.net/images/pdf/Practical_Onion_Hacking.pdf
19. Abbott, T.G., Lai, K.J., Lieberman, M.R., Price, E.C.: Browser-Based Attacks on Tor. In: Borisov, N., Golle, P. (eds.) PET 2007. LNCS, vol. 4776, pp. 184–199. Springer, Heidelberg (2007)
20. Perry, M.: Torow: Tor network analysis. In: HotPETs 2009: 9th Privacy Enhancing Technologies Symposium, p. 14 (August 2009)

Secure Communication Using Identity Based Encryption

Sebastian Roschke[1], Luan Ibraimi[2], Feng Cheng[1], and Christoph Meinel[1]

[1] Hasso Plattner Institute (HPI), University of Potsdam,
P.O. Box 900460, 14440, Potsdam, Germany
{sebastian.roschke,feng.cheng,christoph.meinel}@hpi.uni-potsdam.de
[2] University of Twente,
P.O. Box 217, 7500 AE Enschede, The Netherlands
ibraimi@utwente.nl

Abstract. Secured communication has been widely deployed to guarantee confidentiality and integrity of connections over untrusted networks, e.g., the Internet. Although secure connections are designed to prevent attacks on the connection, they hide attacks inside the channel from being analyzed by Intrusion Detection Systems (IDS). Furthermore, secure connections require a certain key exchange at the initialization phase, which is prone to Man-In-The-Middle (MITM) attacks. In this paper, we present a new method to secure connection which enables Intrusion Detection and overcomes the problem of MITM attacks. We propose to apply Identity Based Encryption (IBE) to secure a communication channel. The key escrow property of IBE is used to recover the decryption key, decrypt network traffic on the fly, and scan for malicious content. As the public key can be generated based on the identity of the connected server and its exchange is not necessary, MITM attacks are not easy to be carried out any more. A prototype of a modified TLS scheme is implemented and proved with a simple client-server application. Based on this prototype, a new IDS sensor is developed to be capable of identifying IBE encrypted secure traffic on the fly. A deployment architecture of the IBE sensor in a company network is proposed. Finally, we show the applicability by a practical experiment and some preliminary performance measurements.

1 Introduction

Secured communication has been widely deployed to guarantee confidentiality and integrity of connections. To achieve this, cryptographic techniques are usually applied to secure the connections, e.g., hash algorithms to secure integrity, asymmetric encryption to secure key exchange, and symmetric encryption to secure confidentiality. Various implementations have emerged and are widely used in practice, such as SSL/TLS [1] and IPSec [2] which provide encryption at the transport layer and the Internet layer. Additionally, a connection can also be secured on the application layer, e.g., simple E-Mail encryption using PGP [3] or XML Encryption. Although secure connections are designed to prevent attacks

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 256–267, 2010.

on the connection, they hide attacks inside the channel from being analyzed by Intrusion Detection Systems (IDS). Furthermore, secure connections require a certain key exchange to work properly. For instance, TLS requires the exchange of a public key between a client and a server, which is prone to Man-In-The-Middle (MITM) attacks [25,26].

Effective prevention of attacks on the communication, e.g., IDS and anti-virus, needs to check the channel, either network traffic or the application layer data on the wire, which leads to several problems. The first problem is performance, as both a network-based IDS sensor and anti-virus software have to analyze large amounts of data in a short time. The second problem is detection in the encrypted traffic. Securing connections by encryption (e.g. SSL/TLS, IPSec, PGP) render the network based IDS and anti-virus software useless, as it is not able to decrypt and to recognize malicious data inside encrypted communication. On the other hand, most of existing cryptography based secure communication require insecure key exchange at the initialization phase. It is not difficult for attackers to exploit this phase, get the keys and carry out further steps of concrete MITM attacks.

In an Identity-Based Encryption[12] (IBE) scheme, the public key of the user is derived from its unique identity, e.g., email address or IP address. The Trusted Authority (TA) or the Key Generation Center (KGC) generates the corresponding private keys. Thus, IBE does not require a digital certificate to certify the public key. Key escrow is an inherent property in IBE systems, i.e., the TA can generate each users' private key, because the TA owns the master key **MK** used to generate users' secret keys. This property can be useful for IDS to decrypt network traffic on the fly and perform content scanning on encrypted traffic.

In this paper, we present a new method to secure connections which provide capabilities for Intrusion Detection and overcome the problem of MITM attacks. We propose to apply IBE to secure communication channels. By integrating the TA in the IDS sensor, it is possible to use the key escrow property for decrypting network traffic on the fly and scan for malicious content. In this way, we can detect malicious content in encrypted network stream without knowing the original private key of the receiver, which can be created on runtime using the master key. The decryption and detection are performed using an IBE sensor. As the public key can be generated based on the identity of the connected server and its exchange is not necessary, the method is not prone to MITM attacks any more. To verify our concept, we implement a prototype of a modified TLS scheme with a simple client-server application. Based on this prototype, we develop a Snort *Preprocessor* to recognize and decrypt IBE secured traffic on the fly. A deployment architecture of the IBE sensor in a company network is proposed. Finally, we show the applicability by a practical experiment and preliminary performance measurements. The contributions of this paper can be summarized as follows:

1. A new mechanism to secure connections, which enables detection of malicious traffic and prevents MITM attacks, is proposed.
2. A modified TLS scheme is implemented.

3. A Snort preprocessor is realized to support detection in IBE secured connections.
4. Practical experiments and preliminary measurement results are presented.

The rest of the paper is organized as follows. Section 2 covers the related work, including an overview on secured connections and the IBE encryption scheme. In Section 3, we describe a modified TLS scheme based on IBE. In Section 4, we propose an important application of the IBE scheme for IDS. Section 5 covers the implementation details of the modified TLS as well as the IBE sensor. In Section 6, we shortly describe the experiments based on our prototype implementations and the results of the performance measurements. After some suggestions for future work in Section 7, the work is concluded in Section 8.

2 Related Work

2.1 Secure Network Connections

As the successor of Secure Sockets Layer (SSL), Transport Layer Security (TLS) is a cryptographic protocol that provides integrity, confidentiality, and authenticity for network communications on the transport layer of the TCP/IP protocol stack [1]. It consists of the TLS Record Protocol and the TLS Handshake Protocol. The TLS Record Protocol uses symmetric encryption and a negotiated secret key to secure the connection, e.g., RC4, Triple DES, AES, IDEA, DES, or Camellia. Furthermore, it uses hash functions to guarantee the integrity of a message, e.g., HMAC-MD5 or HMAC-SHA. The TLS Handshake Protocol allows server and client to authenticate each other and to negotiate protocol parameters, such as an encryption algorithm and the cryptographic keys. The identity can be authenticated using asymmetric crypto algorithms, e.g., RSA, DSA, or ECDSA. For key exchange, different algorithms can be used, such as RSA, Diffie-Hellman, ECDH, SRP, or PSK [5].

The TLS Handshake Protocol provides the negotiation of the parameters necessary to establish the secure connection. The used crypto protocols being used as well as the related keys are communicated. During the handshake, at least the server is being authenticated (client authentication is also possible). The TLS handshake starts when a client requests a secure connection from a TLS enabled server. As next step, the client presents a list of supported ciphers and hash functions. The server selects the strongest cipher and hash function available by matching own supported functions and finally negotiates the parameters. Furthermore, the server sends its identification in the form of a digital certificate, e.g., an X.509 certificate, which contains the server name, the trusted certificate authority (CA), and the server's public key K_spub. The client and server exchange the random numbers R_c and R_s to resist replay attacks. After verification of the certificate, the client encrypts a premaster secret PMS with the server's public key K_spub: $E(PMS, K_spub)$ and sends it to the server, which decrypts it with its private key K_spriv: $E(PMS, K_spriv)$. Based on PMS, R_c, and R_s, the server and the client generate the master secret MS, which is used

for encryption of the connection $MS = genkey(R_c, R_s, PMS)$. This procedure can be illustrated as follows:

1. Negotiation of encryption and hashing algorithms for client and server and random numbers R_c and R_s
2. Transmission and verification of certificate including K_spub
3. Exchange of R_c and R_s
4. Encryption and transmission of PMS: $E(PMS, K_spub)$
5. Generation of MS based on R_c, R_s, and PMS:$MS = genkey(R_c, R_s, PMS)$

Even if the handshake is sniffed, the confidentiality of the messages is preserved, as the master key MS is impossible to be obtained by the attacker. Thus, attacks inside a secured channel can not be realized. However, the SSL/TLS protocol is prone to MITM attacks [25,26]. The attacker intercepts packets between the client and the server to establish two encrypted connections: 1) a connection between the client and the attacker and 2) a connection between the attacker and the server. The attacker generates its own public key and modifies the original certificate to trick the user/client, which is even possible without a capability for the user to notice the attack [25]. By doing this, the attacker can read the traffic of the secured connection between the client and the server. To get position of a MITM, the attacker can use well known techniques, such as ARP Spoofing, DHCP Fake, DNS Spoofing, or ICMP Redirects [27].

To increase the system performance, the SSL scheme can be supported by SSL acceleration [17]. SSL acceleration is implemented by a hardware crypto device which processes the SSL handshake and in some cases the symmetric decryption. Such devices are optimized for cryptographic operations and offer a very high performance. The crypto device can be installed on the secured server to decrease excessive load on the server. There are also dedicated SSL acceleration appliances which can be placed in the network to perform SSL decryption and encryption. In this case, the SSL accelerator works as a proxy and provides encrypted SSL traffic to the external network, and decrypted traffic to the internal network. There are different accelerator devices available for deployment in a company's network, e.g., Array SPX1800 [18], AppXcel [19], and IBM Crypto Cards [20].

2.2 ID-Based Encryption Scheme

The idea of generating users' public key from the identity was proposed by Shamir [11], and the first practical scheme was given by Boneh and Fraklin [12].The Boneh-Franklin scheme consists of four algorithms:

1. $\mathsf{Setup}(k)$: Run by the TA, given a security parameter k, the algorithm generates two cyclic groups $\mathbb{G}$ and $\mathbb{G}_1$ of prime order p, a generator g of $\mathbb{G}$, a bi-linear map $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_1$, a master secret key $\mathbf{MK} = \alpha \in \mathbb{Z}_p^*$, and the hash functions $\mathsf{H}_1 : \{0,1\}^* \rightarrow \mathbb{G}$ and $\mathsf{H}_2 : \mathbb{G}_1 \rightarrow \{0,1\}^n$. The publicly available system parameters are $params = (\mathbb{G}, \mathbb{G}_1, p, g, \mathsf{H}_1, \hat{e}, \mathbf{PK}_{TA})$, where $\mathbf{PK}_{TA} = g^\alpha$ is the public key of the TA.

2. $\mathsf{Extract}(ID)$: Run by the TA, the algorithm takes as input an identifier ID, and outputs the private key $\mathbf{SK}_{ID} = \mathbf{PK}_{ID}^{\alpha}$, where $\mathbf{PK}_{ID} = \mathsf{H}_1(ID)$.
3. $\mathsf{Encrypt}(m, ID)$: Run by the message sender, the algorithm takes as input the message m and an identifier $ID \in \{0,1\}^*$, and outputs the ciphertext $c = (c_1, c_2)$ where $c_1 = g^r$, $c_2 = m \oplus \mathsf{H}_2(\hat{e}(\mathbf{PK}_{ID}, \mathbf{PK}_{TA})^r)$, where $r \in \mathbb{Z}_p^*$.
4. $\mathsf{Decrypt}(c, sk_{ID})$: Run by the message receiver, the algorithm takes a input the ciphertext $c = (c_1, c_2)$, and the user's secret key $\mathbf{SK}_{ID}$, and outputs the message $m = c_2 \oplus \mathsf{H}_2(\hat{e}(\mathbf{SK}_{ID}, c_1))$.

3 A Modified TLS Protocol for IBE Secured Communication

In IBE scheme, if the TA intercepts the communication between Alice and Bob, the TA can decrypt the encrypted data sent from Alice to Bob, and vice versa. For example, when Alice sends an encrypted email to Bob, Alice, firstly, generates Bob's public key: $\mathbf{PK}_{Bob} = \mathsf{H}_1(bob@mail.com)$, and then sends the encrypted data: $c_1 = g^r$, $c_2 = m \oplus \mathsf{H}_2(\hat{e}(\mathbf{PK}_{Bob}, \mathbf{PK}_{TA})^r)$ to Bob. A Trusted Authority(TA) intercepting the communication between Alice and Bob, can compute $c_2 \oplus \mathsf{H}_2(\hat{e}(\mathbf{PK}_{Bob}^{\alpha}, c_1))$ to reveal the message m. In practice, the message m can be a virus, a trojan horse, a worm, etc, which can not be detected in encrypted form. It is expected to use IBE for securing the email communication so that the TA can be used to decrypt encrypted ciphertext and make it possible to scan for malicious code.

By providing a modified TLS scheme, it would be possible to also use IBE to secure connections on the network. Modification of the TLS specifications and implementation is necessary, as both do not support the IBE scheme. A possible modification can be done by introducing IBE as supported encryption method and modifying the TLS handshake to use IBE during parameter negotiation. Introducing IBE as encryption method is needed as there are no easy ways to deploy encryption methods at runtime or add encryption methods unknown to TLS. Modifying the TLS handshake in case of IBE as encryption method is reasonable as a certificate providing the public key is not needed any more. The public key could be derived from the identity of the server, which needs to be provided. The identity of the server could be the URL or the IP of the system. So the client could directly encrypt the PMS messages with the identity derived key of the server. The scheme looks as follows:

1. Negotiation of algorithms to use and exchange R_c and R_s
2. Generation of IBE public key K_spub based on the identity of the server ID_s: $K_spub = genkey(ID_s)$
3. Encryption of the pre-master-secret PMS: $E(PMS, K_spub)$ with the public key K_spub and transmission to server
4. Generation of MS based on R_c, R_s, and PMS:$MS = genkey(R_c, R_s, PMS)$

The key escrow property is used to achieve IDS capabilities, i.e., decrypting network traffic on the fly and scan for malicious content. In this way, we can detect

malicious content in encrypted network stream without knowing the original private key of the receiver, which can be created on runtime using the master key. The decryption and detection are performed using an IBE sensor. As the exchange of the certificate is not needed, the attacker has no chance to forge the certificate and the corresponding public key as a MITM. The client will generate the public from the identity of the server and will directly encrypt the session key with this public key. The attacker can intercept this message, but he is not capable of reading the encrypted session key. Thus, a MITM attack to establish two secured channels that appear as one to client and another one to the server can not be performed. He still can disturb the connection, but the confidentiality and integrity of the communication are secured. The only requirement is that the parameters of the cryptosystem are known before. This can be achieved by a similar method as the distribution of trusted certificates for TLS.

4 Detecting Intrusions in the Modified TLS Communication

The modified TLS protocol makes it possible to detect intrusions in encrypted communication channels. To realize this, the IBE supported IDS sensor (which is called IBE Sensor) is required. We apply the proposed modified TLS in some normal services deployed in a typical enterprise network with a demilitarized zone (DMZ) and the internal subnet. In the internal network, there are several connected clients and the PKI infrastructure for IBE. Within the DMZ there are several servers running, providing services for internal and external clients. To support IBE, there are private keys SKP_i deployed on all servers and CKP_i deployed on clients in the network. These private keys are derived from the master key **MK**. The deployed IBE sensor should be able to read all incoming and outgoing network traffic in the DMZ as well as traffic in the internal network.

Using the master key **MK** it would be possible to decrypt the traffic which is encrypted based on the IBE scheme. If the modified TLS scheme is used, the runtime decryption works by extracting the master secret from the TLS handshake while sniffing all network traffic. This is achieved using the following steps:

1. Recognize negotiation of algorithms and exchange of R_c and R_s (store R_c and R_s)
2. Recognize encrypted pre-master secret $E(PMS, K_spub)$ on the network and decrypt it with master key **MK**: $D(E(PMS, K_spub), \mathbf{MK}) = PMS$ (store PMS)
3. Generate MS based on R_c, R_s, and PMS:$MS = genkey(R_c, R_s, PMS)$
4. Use MS to decrypt the encrypted traffic

As this method is passive, no server or client would be disturbed during its communication. The IBE sensor does not need to intercept and forward traffic, but it needs to store information about the TLS session to assign network traffic to a specific connection/TLS session and a related master secret MS.

The distribution of the master key **MK** yields additional risks for the network. As the master key **MK** is able to decrypt any traffic within the network, it might be a main target for attackers. While the risk for a PKI is known and confined by security mechanisms, the possible risks for an IBE sensor are higher as it is exposed to any traffic on the network. Furthermore, the IBE sensor relies on traffic parsers which might be vulnerable to different kinds of attacks, e.g., Buffer Overflows or Format String Exploits [6]. To secure the IBE sensor and the master key **MK**, additional security measures need to be realized. By successfully attacking the IBE sensor, the attacker will be able to passively sniff all encrypted traffic on the network. With knowledge on the identities, the attacker will be able to generate the private keys and to send falsified messages with a spoofed identity.

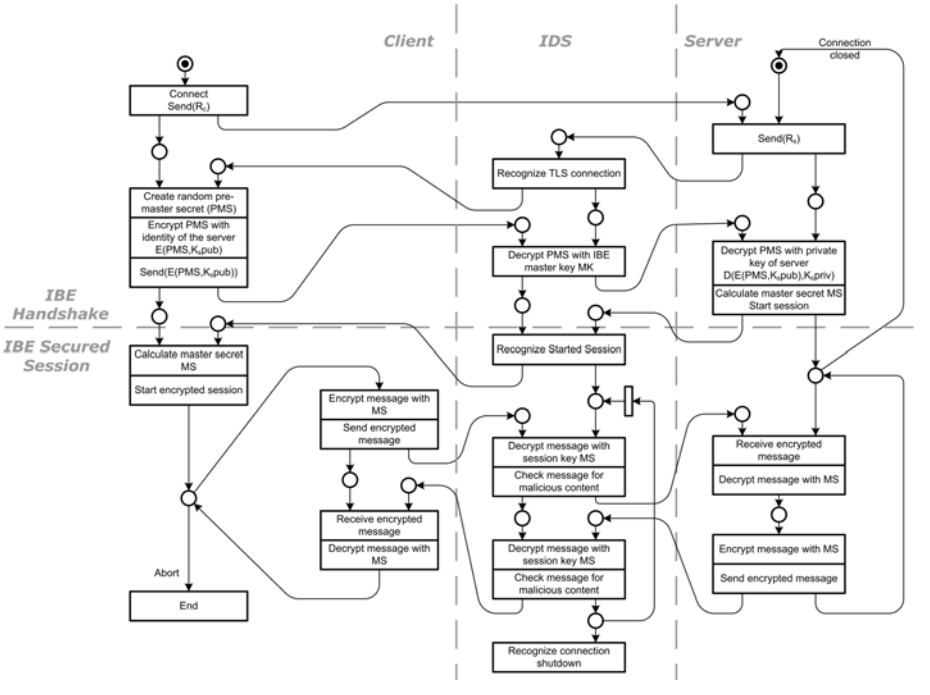


Fig. 1. IBE Sensor Secured Client-Server Communication

5 Implementation

The proof-of-concept implementation consists of two parts: the implementation of a modified TLS specification and the implementation of an IBE-supported Snort IDS sensor (IBE sensor). The modified TLS is implemented between a client and a server. The communication between the client and the server is encrypted based on a session key, which is securely exchanged during the session

negotiation by using the IBE encryption scheme. The Snort IDS sensor is modified to recognize this session negotiation and restore the session key based on the key escrow property.

Figure 1 shows the communication between the client and the server secured by the modified TLS scheme. The scheme consists of two separated phases: the session negotiation and the encrypted session. The initial state is a server listening for connections. After the client connects to the server, it receives the banner and creates a random session key *key_iv*. Then it encrypts the *key_iv* with the identity of the server based on IBE and sends it to the server. The server can decrypt the session key with its private key based on IBE and acknowledge that the session is established. In the following, the client sends multiple encrypted messages to the server. The session is encrypted based on the symmetric Blowfish cipher [23].

An IDS sensor consists of a detection engine, a reporting engine, and a comprehensive rule set for detection. The Snort IDS sensor provides several extension possibilities realized by so called *Dynamic Modules* [4]. *Dynamic modules* can be additional detection rules for the existing detection engine, a new detection engine, or so called *Preprocessors*. A *Preprocessor* is a configurable software component in the Snort architecture, which is called before the detection engine. After the packet has been processed by all preprocessors, it is given to the detection engine to be scanned in detail according to the configured rule sets. To decrypt the TCP stream on the fly before detection, we implemented a *Preprocessor*, which is capable to recognize the modified TLS scheme in a TCP stream.

As shown in Figure 1, the modified Snort can be used to detect attacks in a TCP stream which is encrypted by the modified TLS scheme. To identify a new session, the IDS recognizes the session negotiation, which is sent by the server. The next information sent by the client is the encrypted pre-master secret *PMS*, which can be decrypted by IDS using the master key. By using the extracted *PMS*, the IDS can calculate the session key as well as decrypt and analyze all the network traffic in this session. For this purpose, the *Preprocessor* gets all packets in this session, decrypts it, and modifies the packet on runtime to make it being analyzed by the detection engine. The implementation is done in plain C with support of the OpenSSL library for symmetric encryption and decryption. The session negotiation is implemented on the client and server based on the Stanford IBE implementation [24].

6 Experiment Results

In the experiment scenario, there are several clients and an attacker host connected to the external network, e.g., the Internet. The internal network is attached to the external network using a firewall. The IBE sensor is configured to monitor the internal network and to hold the master key, which is used to generate the server keys SKP_1 to SKP_m. In the internal network there is a server to hold SKP_m and has the identity “server@hpi“. The clients can connect to this

server using the modified TLS scheme, which secures the communication session based on IBE and Blowfish. The sensor is running with an *Intel(R) Core(TM)2 Duo* at $1.4GHz$, i.e., it includes two dedicated cores with $3.072KB$ of cache each. The system has $2GB$ of *RAM*.

During the experiment, we have several clients sending encrypted messages to the server based on the modified TLS scheme. Since he/she can pretend to be a normal user, the attacker is also using the modified TLS scheme to establish a session with the server. Additionally, the attacker tries to send malicious content (e.g., x86 shellcode) to the server within this encrypted session. As the traffic is encrypted based on a session key, the unmodified Snort sensor can not read it and therefore can not detect the shellcode. By using the IBE sensor, which recognizes the session negotiation, we can detect the shellcode within the encrypted session.

The attacker sends shellcode to the server within the encrypted session. This shellcode could be used to exploit the server without being detected by an IDS sensor. The IBE sensor decrypts the session and finds the malicious shellcode in the network traffic. The output is the Snort alert file, which contains the detected attacks on the wire. We also tested other content-based detection rules with the prototype, such as the *Backdoor Rules* used to detect backdoor communication on the network or the *Attack Response Rules* used to detect special responses given by successful attacks [4]. All the tested rule sets worked fine for our prototype.

6.1 Performance Analysis

To test the performance of our implementation, we measured the processing time in the Snort implementation while sending 10 messages with different size from the client to the server. To compare the results, we distinguished three different scenarios. The scenarios are chosen to give a first impression on the performance of the approach. The message sizes in the real world depend on the actual application scenarios:

1. Small size message content - 10 messages, each with 20 Bytes
2. Medium size message content - 10 messages, each with 80 Bytes
3. Large size message content - 10 messages, each with 480 Bytes

We analyzed the results for the key negotiation and the encrypted session separately, to avoid confusion. Finally, Table 1 shows the average values of the processing time comparing the different scenarios, the key negotiation and encrypted session, as well as the processing of Snort with and without runtime decryption. The average of packets during the key negotiation is similar across different scenarios. While the processing time of the packets without runtime decryption is around 35 micro second, the processing time with runtime decryption is around 3600 micro seconds. This is reasonable as the IBE decryption process during the key negotiation needs a lot of additional effort. Furthermore, the effort is the same for the three different scenarios, as the messages which are exchanged during the key negotiation are basically the same. Comparing

the processing times of the packets in an encrypted session, one can realize that the average without runtime decryption is around 28 micro seconds, while the processing time with runtime decryption depends on the message size. The experiment shows the significantly higher processing time of the scenario with large size message content, i.e., 480 Bytes per message. Each encrypted packet needs between 110 and 1250 micro seconds to be processed. The processing time of the scenarios with small size and medium size messages is similar, i.e., between 180 and 210 micro seconds. This can be explained by the functionality of the Blowfish cipher, which is a block cipher. A clear text message is padded to a specific block size (or a multiple of it) to be encrypted with this cipher. The default block size of the Blowfish cipher is 64 bits while our encryption uses 128 bits cleartext blocks to be encrypted. The message size of the two scenarios is padded to the same block size, which needs the same amount of processing time. Therefore, the processing time for both scenarios appears to be the same.

Table 1. Performance Table: Average Snort processing time in micro seconds

	Scenario	Snort Process() without Decryption	Snort Process() with Decryption
Key Negotiation	small	36,5	3820,2
	medium	34,7	3351,75
	large	35,6	3616,88
Encrypted Session	small	30,65	76,29
	medium	29,4	75,73
	large	25,52	369,65

7 Future Work

As the concept shows promising results, the integration of IBE into the real TLS standard should be considered in the future. Besides the evaluation of the applicability, it includes the modification of supported and well-known implementations, such as OpenSSL. In addition to our preliminary measurement results, the performance needs to be analyzed in detail and the scheme can be optimized accordingly to ensure scalability. A possible approach to improve performance can be the utilization of hardware devices to support cryptographic operations for the IDS, e.g., the expensive pairing operations and the symmetric decryption. This could improve the performance and make our approach applicable for huge network environments. The described attack countermeasures does not work with Snort in *In-line* mode [4], a specific operation mode where Snort is not separately deployed in the network. In this way, the sensor can drop malicious packets directly while the sensor itself is exposed to the network and to potential attackers. In this case, further security mechanisms are necessary to secure the master key as well as the sensor itself from being compromised.

Moreover, it could be interesting to analyze the utilization of Attribute Based Encryption (ABE)[13] for secured connections. ABE is a generalization of IBE, and differs from IBE such that the user in ABE get a secret key associated with a set of attributes, while in IBE the user gets a secret key associated with an

identity. Same as IBE, the ABE key escrow is an inherent property since the master key is used to generate each user secret key. A suitable ABE scheme for being applied in IDS is Key-Policy Attribute-Based Encryption (KP-ABE)[15], since it is related to the concept of searching on encrypted data, in which a server on behalf of the user can make a query on the encrypted data without knowing the query and without knowing the plaintext.

8 Conclusion

In this paper, we proposed a new mechanism to create secured connection based on IBE. By using the key escrow property, the secured connection can be recovered and the network traffic is possible to be analyzed by IDS. In this way, we can detect malicious content in encrypted network streams without knowing the original private key of the receiver. Additionally, the secure connections based on IBE have no need to exchange public keys in certificates, which prevents MITM attacks on the secured channel. We provide a deployment architecture of the IBE sensor in a company network. Furthermore, we implemented a prototype of a modified TLS scheme with a simple client-server application to support IBE. Based on this prototype, we developed an IBE sensor (i.e., Snort *Preprocessor*) to recognize and decrypt IBE secured traffic on the fly. Finally, we showed the applicability by a practical experiment and analyzed the performance of our implementation.

References

1. The TLS Protocol, http://www.ietf.org/rfc/rfc2246.txt (accessed January 2010)
2. Security Architecture for the Internet Protocol, http://www.rfc-editor.org/rfc/rfc4301.txt (accessed January 2010)
3. OpenPGP Message Format, http://tools.ietf.org/html/rfc4880 (accessed January 2010)
4. Snort IDS, http://www.snort.org/ (accessed January 2010)
5. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography, 1st edn. CRC Press, Boca Raton (1996)
6. Koziol, J., Litchfield, D., Aitel, D., Anley, C., Eren, S., Mehta, N., Hassell, R.: Shellcoders Handbook. Wiley Publishing, Inc., Chichester (2004)
7. Hallaraker, O., Vigna, G.: Detecting malicious javascript code in mozilla. In: Proceedings of International Conference on Engineering of Complex Computer Systems (ICECCS'05), Shanghai, China, pp. 85–94. IEEE Computer Society, Los Alamitos (2005)
8. Mahoney, M.V., Chan, P.K.: An analysis of the 1999 darpa/lincoln laboratory evaluation data for network anomaly detection. In: Vigna, G., Krügel, C., Jonsson, E. (eds.) RAID 2003. LNCS, vol. 2820, pp. 220–237. Springer, Heidelberg (2003)
9. Ramadas, M., Ostermann, S., Tjaden, B.C.: Detecting anomalous network traffic with self-organizing maps. In: Vigna, G., Krügel, C., Jonsson, E. (eds.) RAID 2003. LNCS, vol. 2820, pp. 36–54. Springer, Heidelberg (2003)

10. Northcutt, S.: Network Intrusion Detection - An Analyst's Handbook. New Riders (1999)
11. Shamir, A.: Identity-based cryptography and signature schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
12. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
13. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
14. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: Proceedings of the 2007 IEEE Symposium on Security and Privacy (S&P'07), Washington, DC, USA, pp. 321–334. IEEE Press, Los Alamitos (2007)
15. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS'06), pp. 89–98. ACM Press, New York (2006)
16. Voltage security, http://www.voltage.com/ (accessed January 2010)
17. SSL Acceleration, http://sslacceleration.info/ (accessed January 2010)
18. Array Networks: Universal Access Controllers, http://www.arraynetworks.net/entry.asp?PageID=110 (accessed September 2009)
19. Radware: AppXcel, http://www.radware.com/ (accessed January 2010)
20. IBM: Crypto Card, http://www-03.ibm.com/security/cryptocards/ (accessed January 2010)
21. Irwin, B.: Unlocking the armour: enabling intrusion detection and analysis of encrypted traffic streams. In: Proceedings of New Knowledge Today Conference (ISSA KTC'05), Sandton, South Africa, pp. 1–10. ISSA Press (2005)
22. Yamada, A., Miyake, Y., Takemori, K., Studer, A., Perrig, A.: Intrusion detection for encrypted web accesses. In: Workshop Proceedings of Advanced Information Networking and Applications (AINA'07), Niagara Falls, Ontario, Canada, pp. 569–576. IEEE Press, Los Alamitos (2007)
23. Schneier, B.: Description of a new variable-length key, 64-bit block cipher (blowfish). In: Anderson, R. (ed.) FSE 1993. LNCS, vol. 809, pp. 191–204. Springer, Heidelberg (1994)
24. Lynn, B.: Stanford IBE Library v0.7.2, http://crypto.stanford.edu/ibe/ (accessed January 2010)
25. Stevens, M., Sotirov, A., Appelbaum, J., et al.: Short Chosen-Prefix Collisions for MD5 and the Creation of a Rogue CA Certificate. In: Halevi, S. (ed.) Advances in Cryptology - CRYPTO 2009. LNCS, vol. 5677, pp. 55–69. Springer, Heidelberg (2009)
26. Boneh, D., Inguva, S., Baker, I.: SSL Man in the Middle Proxy, http://crypto.stanford.edu/ssl-mitm/ (accessed January 2010)
27. Ettercap, http://ettercap.sf.net/ (accessed January 2010)

Anonymous Client Authentication for Transport Layer Security

Kurt Dietrich

Institute for Applied Information Processing and Communications
University of Technology Graz, Inffeldgasse 16a, 8010 Graz, Austria
{Kurt.Dietrich}@iaik.tugraz.at

Abstract. Nowadays, anonymity and privacy protecting mechanisms are becoming more and more important. The anonymity of platforms and the privacy of users operating in the Internet are major concerns of current research activities. Although different techniques for protecting anonymity exist, standard protocols like Transport Layer Security are still missing adequate support for these technologies. In this paper, we analyze how Trusted Computing technologies and anonymous credential systems can be used in order to allow clients to establish anonymous authentication over secure channels. Furthermore, we analyze how these technologies can be integrated into common security frameworks like the Java Cryptography Architecture. We discuss the performance that can be achieved with this approach and analyse which performance can be expected from currently available Trusted Platform Modules.

Keywords: Trusted Computing, Transport Layer Security, TLS, Direct Anonymous Attestation, Java, DAA.

1 Introduction

Anonymity has been a major topic in Trusted Computing since its beginnings. In order to achieve anonymity protection for trusted platforms, two different concepts have been introduced by the Trusted Computing Group (TCG): the *PrivacyCA* (PCA) scheme and the Direct Anonymous Attestation (DAA) scheme, both allowing trusted platforms to hide their identity when operating over the Internet. Both schemes address the problem that arises when performing public key operations. When doing such operations, a platform can always be tracked and identified by public keys and certificates that are associated with it. Both schemes address this problem, however, with different methods.

The first scheme is based on remote certification of public keys. The platform creates a temporary key-pair for each new transaction. Prior to the transaction, the public part of the key-pair has to be sent to the PCA for certification. A verifier who receives information that was signed with such a temporary key is able to verify the signature and the authenticity of the key, but he only sees the certificate from the PCA and cannot link the signature to the originating platform. However, this approach has some severe drawbacks. Every newly created key pair

B. De Decker and I. Schaumüller-Bichl (Eds.): CMS 2010, LNCS 6109, pp. 268–280, 2010.

has to be sent to the PCA, thereby requiring the PCA to be permanently available. Moreover, the PCA could record and store all certification requests from the single platforms and the PCA would be able to link the issued certificates to the requesting platform. This fact opens a big security leak in case the PCA gets compromised. An adversary could use this information to link transactions and signatures to single platforms. Furthermore, it is not specified how often such a certified key may be used. If the certified key is re-used for different operations, an adversary that is able to track the operations that are performed with this key could link the different signatures to the originating platform. In addition, the revocation of such credentials is still an unsolved problem. There are no mechanisms specified for revoking the certificates and if the PCA decides not to store the certification information, there is no way of revealing the real identity of the corresponding platform in case of fraud. In order to overcome these problems, the TCG introduced another scheme. The Direct Anonymous Attestation scheme relies on local certification, thereby omitting the need for a remote party to certify the keys. It is based on a group signature scheme and Zero-Knowledge proofs, allowing each platform to create signatures on behalf of a group which can be verified by a single group-public key. The advantage of this scheme over the previously discussed one is that no on-line connection to a third party is required. Moreover, different signatures created by the same platform cannot be linked to this certain platform - not even by the group manager which is responsible for issuing group credentials to each platform of the group. In case the issuer becomes compromised, the adversary only gets knowledge that a certain platform is part of the group managed by the issuer, other information is not stored by the group manager. With only this information, it is not possible to link any signature that has been created before the compromission or any signature that will be created afterwards, to a single platform. Another advantage of this scheme is that it supports the unmasking of a platform's real identity based on certain events and conditions in case of misuse. However, this scheme is based on complex computations, making it hard to use on resource constrained devices. Moreover, the scheme requires a tamper resistant storage device (i.e. the TPM) for protection of the group credentials and DAA keys. These credentials must not be copied or moved to a different platform as they are issued to a specific platform. A more detailed discussion of the DAA scheme and its revocation mechanism is given in Section 2.2. The support for both schemes is an integral part of all TPMs since version 1.2. Furthermore, as TPMs are available on many desktop and notebook platforms, the question arises whether these schemes can be applied for applications and technologies other than trusted computing related ones which also have a high demand for privacy protection. One such technology is the Transport Layer Security protocol (TLS) [1] which is used in many systems to establish secure and authenticated connections.

1.1 Related Work

Several ideas for integrating TPMs or Trusted Computing technology in TLS have been published. Latze et. al. propose to use the TPM for identity distribution, authentication and session key distribution and have defined an Extensible

Authentication Protocol (EAP) extension in order to integrate the Trusted Computing and TPM related information [2]. Although the protocol supports anonymous authentication, it is based on the PCA scheme and not on DAA. Moreover, this document is currently work-in-progress and is tagged to be in an "experimental" status.

The approach from Cesena et. al [3] aims at providing anonymous authentication for trusted platforms and trusted applications in the sense of Trusted Computing. In their work, they define a set of extensions for TLS for transporting DAA related information and propose a design for integrating it into OpenSSL. They use the DAA scheme as defined in the Trusted Software Stack specification [4] which requires them to use a full blown Trusted Software Stack (TSS) [5]. They also provide an ECC based variant of the DAA scheme which is not supported by existing TPMs. Another interesting publication that is not directly related to this work, but may have interesting implications on further applications, is discussed in [6]. Bichsel et. al. propose an implementation of DAA on a JavaCard. Using this approach, it might be possible to use the anonymous authentication mechanism in combination with smart cards instead of TPMs. This approach could increase the flexibility and area of application of the anonymous TLS authentication as the anonymous credentials can be bound to a specific user instead to a specific TPM and platform.

1.2 Our Contributions

In this article, we focus on the DAA scheme for providing anonymity in secure channels. Our approach does not focus on promoting Trusted Computing and its applications, it rather makes use of Trusted Computing technologies (i.e. the TPM and DAA support) which are available on many PC platforms and aims at using these technologies for arbitrary applications. Instead of using a full-blown TSS which is typically used for trusted applications to use the TPM features, we focus on a minimal set of functions that are required for communicating with the TPM and employ its DAA features. This is also true for the changes required in the TLS protocol. This approach allows us to use it on embedded systems and mobile phones which are going to be equipped with TPMs in the near future [7]. Different approaches for employing TPM functionality on mobile and embedded devices is discussed in [8], [9], [10] and [11]. In order to demonstrate our achievements, we developed a library with focus on portability, size efficiency and simple usability. This minimal DAA library is based on the DAA scheme which we will address as BCC05 [12], named after its authors (Brickel, Camenish, Chen) and its year of publication.
In contrast to the DAA scheme defined by the TCG (which we will refer to as BCC04) in the TPM 1.2 specification, we use the BCC05 scheme which is a performance optimized scheme that requires less parameters and less computations than the BCC04 scheme, however, both can use the DAA features from TPMs without any modifications of the TPM. A discussion about the differences between BCC04 and BCC05 and the security of the BCC05 scheme can be found in [12]. Furthermore, we give performance measurement values demonstrating the

performance that can be achieved using this technique in combination with currently available TPMs. Our DAA library is developed in the Java programming language and is designed to fit into the Java Cryptography Architecture (JCA), allowing easy use of the DAA scheme for developers that are not familiar with DAA and Trusted Computing. The rest of this article is organized as follows: we give background information on TLS client authentication and the DAA scheme. We discuss our implementation and its integration in the JCA framework and examine the achieved performance values. In one paragraph, some comments on compatibility of TPMs from different vendors are given. Finally, we conclude the article with a summary and suggestions for future improvements.

2 Background

2.1 TLS Client Authentication

TLS provides a feature that allows a server to request authentication information from clients that want to connect to it. This feature is called *client authentication*. Figure 1 shows the basic flow of a TLS handshake. The messages marked with

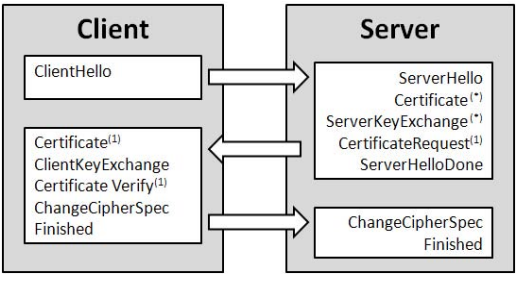


Fig. 1. The TLS Handshake Protocol

(1) are required for client authentication which works as follows: The server sends the client a list with certificate authorities (CAs), which it accepts. The client selects a CA and returns a $Certificate$ message that contains the client's certificate, which then certifies its authentication key. The $CertificateVerify$ message contains a signature on the hash of the handshake messages sent so far. By verifying the signature - the hash can be computed by the server as it knows all messages that have been exchanged with the client - and by verifying the client's certificate plus the corresponding certificate chain, the server can validate the client's authentication information [1].

2.2 Direct Anonymous Attestation

The DAA protocol is basically a group signature scheme based on Zero-Knowledge proofs. Instead of showing a credential to a verifier like in common PKI systems,

the creator of a DAA signature computes a proof that it is in possession of certain group credentials. A detailed discussion is out of scope of this document, hence, we focus on a high-level discussion of the basic protocols *Join* and *Sign*. Before a platform can create anonymous signatures on behalf of a group, it has to *join* the group and obtain credentials from the group manager. Moreover, the TPM contains the secret keys f_0, f_1, ν' (the key f is separated in two parts for performance reasons) that are created during the Join phase. During the join process, the client proves knowledge of f_0, f_1 to the group manager. The group manager computes the credentials (A, e, ν'') where e is a random prime and ν'' a random integer and computes a proof that A was generated correctly. The client verifies the proof on A and verifies that e is a prime in a certain interval.

After successful execution of the Join protocol, the client has obtained the credentials $(A, e, \nu = \nu' + \nu'')$ which represent a signature on the keys f_0, f_1 that are stored in the TPM. Moreover, the TPM has obtained a value ν and the platform has obtained a value ν'' which allows the TPM to create a DAA signature σ together with the platform. Details how this is achieved can be found in Section 4.2.

However, before a client may enter a group, the client and its TPM might have to be authenticated depending on the issuer policy, unless arbitrary clients are allowed to enter the group. Generally, the issuer could have a list of endorsement keys (EK)[1] from the platforms that are allowed to join. The basic approach to identify which TPM the issuer is talking to and whether the TPM is genuine TPM or not, is to encrypt a challenge with the TPM's EK. The TPM then computes the hash of the challenge and a previously committed value. The issuer can now, by verifying the computed hash, determine if the TPM he is communicating with is a registered platform. Moreover, with the EK's certificate, the issuer can determine the vendor of the TPM and whether the TPM is a genuine one or not. This step is executed during the Join phase. The Join protocol is not part of the TLS protocol and has to be executed before a client can use anonymous TLS authentication. The issuer does not have to be the target TLS server, it can be any server, however, the client and the TLS server have to trust the issuer and its public key. When the platform has obtained all required credentials, it is ready to create DAA signatures. A signature is computed by host and TPM, allowing to outsource most of the computations to the host platform and relieving the resource limited TPMs.

The DAA scheme can be used in two different modes of anonymity: *total anonymity* and *pseudonymity*. When using total anonymity, the creator of a DAA signature cannot be identified. This has impact on the revocation check, as malicious platforms and compromised TPMs can also not be detected. When using pseudonymity, a verifier can detect if different signatures stem from the same platform, however, he is not able to identify the platform. How the pseudonymity mode can be used for credential revocation is discussed in the following Paragraph.

[1] The endorsement key pair is a RSA key pair that is unique to every TPM. It is not modifiable and can only be used for decryption operations inside the TPM.

Credential Revocation. Revocation of credentials is an essential feature of common public key infrastructures. Credentials may be revoked for different reasons e.g. loss of the private key, change of the owner's name etc. The DAA scheme also provides a revocation mechanism called *rogue tagging*. If a TPM is compromised and the DAA credentials f_0 and f_1 become public, a verifier can detect the malicious TPM according to its pseudonym. The pseudonym depends on a *basename* that is defined by the verifier. With this *basename*, a signer can compute its pseudonym which can be compared by the verifier with a list of pseudonyms that is computed from the list of compromised DAA keys. However, such lists can get very large as every TPM may generate an arbitrary number of different keys. A more sophisticated revocation mechanism can be establish by involving an *Attribute Revocation Authority* (ARA) [4]. The ARA is a trusted third party that can revoke the anonymity of platforms. The client encrypts its identity with the public key of the ARA. A verifier can then forward this encrypted identity which it received from a signer. By decrypting the identity, the ARA, and only the ARA, can see the signer's identity and reveal it to the verifier.

3 The Authentication Scheme

In contrast to [3], we do not use TLS extensions to transport DAA specific information. In order to keep the implementation for our demonstrator small and simple, we do not implement the extension framework. We, therefore, modify the handshake messages directly as we also want to use the demonstrator on resource constrained mobile devices. For example, the *basename* for rogue tagging that is sent by the verifier is appended to the $ServerHello$ message. Further revisions of the implementation might include support for the TLS extension framework.

As discussed in Section 2, the TLS client, if requested by the server, sends its certificate together with a signature $h_m = H(messages)$ on the handshake messages sent so far. Thus, the server can authenticate the client and the client's key. Typically, the credential is a X.509 certificate signed by a certification authority (CA). With the DAA scheme and a TPM, it is possible to create and certify a temporary key locally on the client by signing it with a DAA signature instead of using a certificate from a CA to authenticate it. In our prototype, we only sign the key and send the key plus the DAA signature to the host instead of a X.509 certificate and the according chain. The certificate has no value in this case as it must not contain any information (e.g. subject or serial number) that might compromise the platform's anonymity. The signature on h_m is done by the temporary key, as discussed in the introduction in Section 1.

Another approach could be to sign h_m directly with a DAA signature, however, the results from Table 2 clearly show that this is not a good approach with common TPMs (except the Intel TPM). The low performance of TPMs prohibits the direct use of the DAA scheme for signing information that is used imminently. A better approach is to create temporary keys and sign and certify them prior to opening a TLS channel. The authentication key can be a public

key of any common signature scheme - it may be a RSA key, an ECC or DH key parameters. For extra security - note that when using the discussed pre-certification, the certified key might be stolen from the platform before it can be used - the temporary authentication key could be created and secured by the TPM. The TPM also provides support for RSA signatures and RSA key-pair generation. These features can be used to create temporary keys and to sign h_m. In this case, the TPM provides additional security as it can create, store and operate with the temporary key in a secure and tamper resistant environment. The speed of this sign operation is between one and two seconds which is sufficient for the signature of the client authentication. At the moment, TPMs only support the RSA signature scheme but this will change in future versions. Current efforts are aiming at equipping TPMs with elliptic curve cryptography. With these new algorithms, TPMs will be able to create ECC signatures, allowing this authentication procedure to support ECC.

4 Implementation Aspects

In order to demonstrate our results, we developed a library that provides the cryptographic operations for the BBC05 scheme and the DAA commands as defined in [13]. In detail, we implemented the following TPM commands: TPM_DAA_Sign, TPM_DAA_Join, TPM_FlushSpecific, TPM_OIAP, TPM_Terminate_Handle and the following structures: TPM_DAA_ISSUER, TPM_NONCE.

Support for the modular arithmetic operations, the RSA-OAEP encryption scheme and the DAA protocol operations is provided by the IAIK-JCE-MicroEdition. Details of our implementation of the cryptographic functions and the BCC05 scheme can be found in [14].

The DAA scheme is basically a signature scheme like RSA or ECDSA. Consequently, it can be integrated into existing security or cryptographic software frameworks like the Java Cryptographic Architecture. The development of a JCA provider for our DAA library allows to abstract the complex API definitions in the TSS [4] specification and makes it accessible to developers that are not familiar with Trusted Computing.

Access to the different TPMs is provided by the Linux kernel module drivers from the specific vendors. The TPM is mapped into the userspace via the `/dev/tpm` device alias. This device can then be accessed by a Java `java.io.RandomAccessFile` object in order to send and receive TPM commands.

4.1 Performance Evaluation

The DAA scheme involves complex mathematical computations, hence, it is of interest which performance can be achieved with currently deployed TPMs. Table 1 shows the performance of the Join protocol on a Intel PC that is equipped with an Infineon TPM 1.2 (rev. 1.2.3.16). The performance values were measured on a HP Compaq dc7900 platform with an Intel Pentium Dual Core CPU E5200 2,5 GHz, a SUN 1.6 Java virtual machine (64 bit) and a Debian Linux operating

Table 1. Performance of the Join Protocol with Intel TPMs

DAA Join	Host	TPM	Host+TPM	Issuer
TPM 1.2$_{INF}$	144,8 ms	56,7 s	56,8 s	712 ms
Emulator	144,8 ms	372,8 ms	517,6 ms	712 ms

system with a 2.6.30-1 Kernel (64 bit). A verification of a DAA signature takes about half the time required for signature creation and does not require a TPM. All results presented in this Section represent the average measurement values of 100 executions of the *Join* and the *Sign* protocol. We have also performed tests

Table 2. Performance of DAA signature creation with Intel TPMs

DAA Sign	Host	TPM	Host + TPM
TPM 1.2$_{INF}$	300 ms	37,7 s	38,0 s
Emulator	300 ms	67 ms	367 ms

with TPMs from ST Micro and Intel. The results are shown in Table 3 which clearly demonstrate the performance advantage of the Intel TPM. This advantage results from the different hardware used to host the TPM functionality. While the ST Micro TPM is basically a smart card controller that is attached to the PC's motherboard via the LPC bus [15], the Intel TPM is located in the Intel motherboard chipset (i.e. the Intel 82801JDO Controller Hub (ICH10DO)) itself [16].

Table 3. Comparison of the DAA Performance of different TPMs

TPM	ST Micro 1.2 (rev 3.11)	Intel 1.2 (rev 5.2)
DAA Join	44.55 s	7.64 s
DAA Sign	33.38 s	4.66 s
Eval. Board	Intel DQ965GF	Intel DQ45CB
Operating System	Ubuntu v2.6.31-19 32 bit	Ubuntu v2.6.31.12-0.1 64 bit

Most of the time is consumed by the modular exponentiations and the paramter handling. As the TPM implementors want to save as much TPM resources as possible, the parameters obtained during the Join protocol are stored - encrypted with the EK - outside the TPM. For example, the DAA keys f_0, f_1, ν_0 and ν_1 have to be loaded into the TPM for each single DAA signature operation which takes about 1.5 seconds for each parameter on the ST Micro TPM. Each modular exponentiation requires about 2 to 8 seconds. The exact sequence of operations can be found in [13].

4.2 Integration into the JCA Architecture

The TCG has specified an API for using the DAA features on trusted platforms. However, this specification is rather complex and is - even for developers that

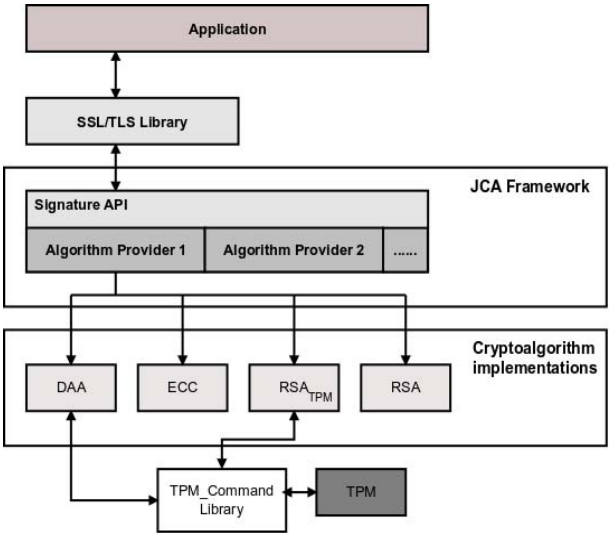


Fig. 2. Integration of the DAA library in the JCA Architecture

are familiar with Trusted Computing - hardly accessible. As mentioned before, the DAA scheme is basically a signature scheme, hence, it is well suited for integration in the Java Cryptography Architecture Framework (JCA) [17]. The advantage of using such a common framework is that for creating signatures, the same API, independent from the signature scheme can be used. Figure 2 shows how our DAA implementation is integrated into the JCA framework. The application uses the iSaSiLk library [18] in order to open a secure channel.

The library uses the JCA framework to request a certain algorithm implementation according to the TLS session parameters that have been negotiated during the TLS handshake. The algorithm implementation can be either a software implementation of the algorithm or an interface to some hardware device. Our DAA signature implementation consists of two parts: the host part and the TPM part. Both parts are abstracted by the signature class API. The host part performs the DAA computations that can be done in software while the TPM part handles the communication with the TPM. For using the DAA sign function of the TPM, the TPM_DAA_Sign [4] command is sent several times in sequence with different parameters to the TPM. From the application's point of view, the algorithm can be initialized via the common Signature API by defining the algorithm and algorithm parameters. The same approach can be applied to use the RSA implementation in the TPM (TPM_{RSA} in Figure 2). Moreover, the JCA framework allows different DAA schemes to be used with the same API. For example, the BBC04 scheme could be added to the framework as an alternative to BCC05. In our implementation, the computation of the DAA signature is done according to the BCC05 scheme algorithm which works as follows:

1. If a *basename* is included in the ServerHello request, the verifier requests rogue tagging and the host computes $\zeta = H(basename)^{(\Gamma-1)/\rho} \bmod \Gamma$ which is sent to the TPM.
2. The host part of the signature class computes: $T = AS_0^{w_0} S_1^{w_1} \bmod n$ with $w_{0,1} \in \{0,1\}^{l_n+l_\phi}$
3. The TPM computes $N_V = \zeta^{f_0+f_1*2^{104}} \bmod \Gamma$. Host and TPM can now compute the "signature of knowledge":
4. The TPM computes: $\tilde{T}_t = R_0^{r_{f_0}} R_1^{r_{f_1}} S_0^{r_{\nu_0}} S_1^{r_{\nu_1}} \bmod n$ with r_{f_0}, r_{f_1} of size $l_f + l_\phi + l_H$ bits and $r_{\nu_{(0,1)}}$ with length $l_n + l_\phi + l_H$. $\tilde{N}_V = \zeta^{\tilde{r}_f} \bmod \Gamma$
5. $\tilde{T}_t$ and $\tilde{N}_V$ are returned to the host which computes: $\tilde{T} = \tilde{T}_t T^{r_e} S_0^{r_{\bar{\nu_0}}} S_1^{r_{\bar{\nu_1}}} \bmod n$ where $\tilde{T}_t$ is the input from the computation process that was performed in the TPM. The random parameter r_e is of size $\{0,1\}^{l_e'+l_\phi+l_H}$ whereas $r_{\bar{\nu}_0}$ and $r_{\bar{\nu}_1}$ are of size $\{0,1\}^{(l_e+l_n+2l_\phi+l_H+1)/2}$ bits.
6. Moreover, the host part computes: $c_h = H((n\|R_0\|R_1\|S_0\|S_1\|Z\|\gamma\|\Gamma\|\rho)\|\varsigma\|T\|N_V\|\tilde{T}\|\tilde{N}_V)\|n_v)$
7. The TPM selects a random $n_t \in \{0,1\}^{l_\phi}$, computes $c = H(H(c_h\|n_t)\|b\|m)$ and $s_{\nu_0} = r_{\nu_0} + c*\nu_0$, $s_{\nu_1} = r_{\nu_1} + c*\nu_1$, $s_{f_0} = r_{f_0} + c*f_0$ and $s_{f_1} = r_{f_1} + c*f_1$ where b is a parameter that defines whether the authenticated data is a key that was loaded into the TPM or some arbitrary data.
8. The host part computes $s_e = r_e + c*(e - 2^{l_e-1})$ and $s_{\bar{\nu_0}} = s_{\nu_0} + r_{\bar{\nu_0}} - cw_0 e$, $s_{\bar{\nu_1}} = s_{\nu_1} + r_{\bar{\nu_1}} - cw_1 e$
9. Finally, the host assembles the signature $\sigma = (\zeta, T, N_v, c, n_t, (s_{\bar{\nu}_0}, s_{\bar{\nu}_1}, s_{f_0}, s_{f_1}, s_{f_e}))$. The signature σ can now be verfied by the public key $PK_I = (n, R_0, R_1, Z, \gamma, \Gamma, \rho)$.

Note that the parameter S is separated into $S_0 = S$ and $S_1 = S^{2^t}$ and $\nu = \nu_0 + \nu_1 * 2^l$ as the crypto co-processor of the TPM cannot compute the modular exponentiations if the exponent is larger than the modulus n.

The computation of the signature is separated between host and TPM, prohibiting that the credentials (A, e, ν'') that are stored on the platform can be copied and used on a different platform. The following listing gives an overview on how a signature object using the DAA algorithm can be initiated and used. Prior to creating a DAA signature, the signature object has to be initialized with several parameters (via the signature.setParams(...) method). Some of the parameters are acquired during the Join phase and are encrypted with the TPM's endorsement key. The parameters include: TPM authentication information, the public part of the EK, issuer parameters, the encrypted values $enc_{EK}(\nu_0)$, $enc_{EK}(\nu_1)$, the encrypted DAA keys $enc_{EK}(daaBlobKey)$ (i.e. group specific credentials that are bound to a specific TPM) and the *basename* and a nonce n_v from the verifier.

The result of the *sign*() method is a DAA signature σ on the message m. The verification process is handled in a similar way by the same class. However, the signature object is initialized in verification mode and the task does not involve a TPM. The resulting implementation consisting of TPM commands, DAA implementation (Join and Sign protocols) and other cryptographic algorithms (RSA and RSA with OAEP, SHA1-HMAC) is about 150 kByte of size. In

addition, the TLS implementation is about 120 kBytes resulting in total of 270 kBytes which allows efficient usage on mobile platforms. Performance results of our implementation on mobile devices can be found in [14].

```
Signature signature = Signature.getInstance("DAA/BCC05",
"Provider");
signature.initSign();
signature.setParams( daaParams );
signature.update( "Test data".getBytes() );
byte[] sigValue = signature.sign();
```

Listing 1. Example Code

4.3 A Note on Specification Compliance

In the TPM specification [13], the commands and structures for using the DAA functions of TPMs are defined. However, during our experiments we realized that the TPM vendors have different interpretations of this specification. We have tested our library with TPMs from Infineon, Atmel, Winbond, Intel, ST Micro and with the TPM emulator. Each of them has some deviations from the original specification, which result in different DAA implementations for TPMs from specific vendors. For example, the TPM emulator stays close to the specification and uses the definition from the specification which says that parameters can be encoded as parameter length plus parameter. The Infineon TPM, however, requires parameter sizes strictly to be 256 bytes long unless otherwise specified. Moreover, the emulator does not correctly check parameter sizes of the commands. Although that does not appear to a big problem, the emulator, respectively its code basis, is used in many implementations such as mobile TPMs or virtual TPMs for XEN [19]. Furthermore, the emulator does not close the join session after it has finished. The corresponding session parameters and reserved resources inside the TPM have to be flushed from the TPM manually by invoking a TPM_FlushSpecific command. According to the specification, the Join session and associated resources must be freed after execution of the command. A special case are TPMs from Atmel and Winbond. During our experiments, we failed to invoke the DAA functions as these TPMs seem to deviate from the standard when it comes to verifying the issuer settings during the Join process. There are many more deviations that have to be taken into account when working with TPMs, especially when using the DAA functionality. Unfortunately, a continuative detailed discussion is out of scope of this document.

5 Conclusion and Future Work

In this paper, we elaborate on different topics: we discuss how a anonymous credential systems based on TPMs can be integerated into the JCA architecture and how they can be used for anonymous TLS client authentication. Moreover,

we show which performance can be achieved with currently available TPMs. Although the performance of TPMs like the ST Micro or the Infineon TPM is rather slow, the fast Intel TPM allows the use of anonymous credentials in an efficient way.

References

1. Dierks, T., Rescorla, E.: R.I: The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard) (August 2008)
2. Latze, C., Ultes-Nitsche, U., Baumgartner, F.: Extensible Authentication Protocol Method for Trusted Computing Groups (TCG) Trusted Platform Modules. Internet-Draft (July 2009)
3. Cesena, E., Ramunno, G., Vernizzi, D.: D03c.3 ssl/tls daa-enhancement specification. Technical report, Politecnico Di Torino (May 2009)
4. Trusted-Computing-Group-TSS-Working-Group: TCG Software Stack (TSS) Specification Version 1.2 Level 1. Part1: Commands and Structures (January 2006), Specification available online at, `https://www.trustedcomputinggroup.org/specs/TSS/TSS_Version_1.2_Level_1_FINAL.pdf`
5. IBM: TrouSerS The opensource TCG Software Stack (November 2, 2007)
6. Bichsel, P., Camenisch, J., Groß, T., Shoup, V.: Anonymous credentials on a standard java card. In: CCS '09: Proceedings of the 16th ACM conference on Computer and communications security, pp. 600–610. ACM, New York (2009)
7. Trusted Computing Group - Mobile Phone Working Group: TCG Mobile Trusted Module Sepecification Version 1 rev. 1.0, Specification available online at, `https://www.trustedcomputinggroup.org/specs/mobilephone/tcg-mobile-trusted-module-1.0.pdf` (June 12, 2007)
8. Ekberg, J.E., Bugiel, S.: Trust in a small package: minimized mrtm software implementation for mobile secure environments. In: STC '09: Proceedings of the 2009 ACM workshop on Scalable trusted computing, pp. 9–18. ACM, New York (2009)
9. Winter, J.: Trusted computing building blocks for embedded linux-based arm trustzone platforms. In: STC '08: Proceedings of the 3rd ACM workshop on Scalable trusted computing, pp. 21–30. ACM, New York (2008)
10. Dietrich, K.: An Integrated Architecture for Trusted Computing for Java Enabled Embedded Devices. In: STC '07: Proceedings of the 2007 ACM workshop on Scalable trusted computing, pp. 2–6. ACM Press, New York (2007)
11. Dietrich, K., Winter, J.: Implementation aspects of mobile and embedded trusted computing. In: Chen, L., Mitchell, C.J., Martin, A. (eds.) Trust 2009. LNCS, vol. 5471, pp. 29–44. Springer, Heidelberg (2009)
12. Mitchell, C.: Direct Anonymous Attestation in Context. In: Trusted Computing (Professional Applications of Computing), Piscataway, NJ, USA, pp. 143–174. IEEE Press, Los Alamitos (2005)
13. Trusted Computing Group - TPM Working Group: TPM Main Part 3 Commands, Specification available online at, `https://www.trustedcomputinggroup.org/specs/TPM/mainP3Commandsrev103.zip` (July 9, 2007) Specification version 1.2 Level 2 Revision 103
14. Dietrich, K.: Anonymous credentials for java enabled platforms. In: Chen, L., Yung, M. (eds.) INTRUST 2009, pp. 101–116 (2009)
15. Intel: Intel Desktop Board DQ965GF Technical Product Specification, Specification available at, `downloadmirror.intel.com/15033/eng/DQ965GF_TechProdSpec.pdf` (September 2006)

16. Intel: Intel Desktop Board DQ45CB Technical Product Specification, Specification available at, http://downloadmirror.intel.com/16958/eng/DQ45CB_TechProdSpec.pdf (September 2008)
17. SUN Microsystems: Java Cryptography Architecture (JCA) Reference Guide for JavaTM Platform Standard Edition 6, Specification available at, http://java.sun.com/javase/6/docs/technotes/guides/security/crypto/CryptoSpec.html
18. Stiftung SIC: The IAIK JCE iSaSiLk v4.4 TLS Library, Specification available at, http://jce.iaik.tugraz.at/index.php/sic/Products/Communication-Messaging-Security/iSaSiLk
19. Williams, D.E., Garcia, J.R.: Virtualization with Xen: including XenEnterprise, XenServer, and XenExpress. Syngress, Burlington, MA (c2007) Includes index

Author Index