

CONFERENCES IN RESEARCH AND PRACTICE IN
INFORMATION TECHNOLOGY

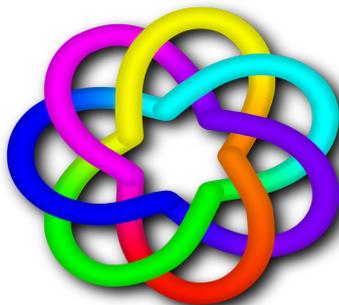
VOLUME 105

INFORMATION SECURITY 2010

AUSTRALIAN COMPUTER SCIENCE COMMUNICATIONS, VOLUME 32, NUMBER 4



AUSTRALIAN
COMPUTER
SOCIETY



 **CORE**
Computing Research & Education

INFORMATION SECURITY 2010

Proceedings of the
Eight Australasian Information Security Conference
(AISC 2010), Brisbane, Australia,
January 2010

Colin Boyd and Willy Susilo, Eds.

Volume 105 in the Conferences in Research and Practice in Information Technology Series.
Published by the Australian Computer Society Inc.



Published in association with the ACM Digital Library.

Information Security 2010. Proceedings of the Eight Australasian Information Security Conference (AISC 2010), Brisbane, Australia, January 2010

Conferences in Research and Practice in Information Technology, Volume 105.

Copyright ©2010, Australian Computer Society. Reproduction for academic, not-for-profit purposes permitted provided the copyright text at the foot of the first page of each paper is included.

Editors:

Colin Boyd

Faculty of Information Technology,
Queensland University of Technology,
Brisbane Q4001
Australia
Email: c.boyd@qut.edu.au

Willy Susilo

School of Computer Science and Software Engineering,
University of Wollongong
Northfields Avenue
Wollongong NSW 2522
Australia
Email: wsusilo@uow.edu.au

Series Editors:

Vladimir Estivill-Castro, Griffith University, Queensland
Simeon J. Simoff, University of Western Sydney, NSW

crpit@scm.uws.edu.au

Publisher: Australian Computer Society Inc.
PO Box Q534, QVB Post Office
Sydney 1230
New South Wales
Australia.

Conferences in Research and Practice in Information Technology, Volume 105.
ISSN 1445-1336.
ISBN 978-1-920682-86-6.

Printed, December 2009 by UWS Press, Locked Bag 1797, South Penrith DC, NSW 1797, Australia
Document engineering by Susan Henley, University of Western Sydney
Cover Design by Matthew Brecknell, Queensland University of Technology
CD Production by FATS Digital, 318 Montague Road, West End QLD 4101, <http://www.fats.com.au/>

The *Conferences in Research and Practice in Information Technology* series aims to disseminate the results of peer-reviewed research in all areas of Information Technology. Further details can be found at <http://crpit.com/>.

Table of Contents

Proceedings of the Eight Australasian Information Security Conference (AISC 2010), Brisbane, Australia, January 2010

Preface	vii
Programme Committee	viii
Organising Committee	ix
Welcome from the Organising Committee	x
CORE - Computing Research & Education	xi
ACSW Conferences and the Australian Computer Science Communications	xii
ACSW and AISC 2010 Sponsors	xiv

Keynote Talk

Information Sharing in the 21st century: Progress and Challenges	3
<i>Ed Dawson, Jason Reid, Farzad Salim, Mark Burdon</i>	

Contributed Papers

Secure Coprocessor-based Private Information Retrieval without Periodical Preprocessing	7
<i>Peishun Wang, Huaxiong Wang and Josef Pieprzyk</i>	
Reconstruction of Falsified Computer Logs for Digital Forensics Investigations	15
<i>Maolin Tang and Colin Fidge</i>	
Advantages und vulnerabilities of pull-based email-delivery	25
<i>Natascha Chrobok, Andrew Trotman and Richard O'Keefe</i>	
An Administrative Model for UCON	35
<i>Farzad Salim, Jason Reid and Ed Dawson</i>	
Impeding CAPTCHA Breakers with Visual Decryption	43
<i>Simon Lang and Neville Williams</i>	
Information security culture: A Behaviour Compliance Conceptual Framework	51
<i>Salahuddin Alfawaz, Karen Nelson and Kavoos Mohannak</i>	
Multi-Factor Password-Authenticated Key Exchange	61
<i>Douglas Stebila, Poornaprajna Udipi and Sheueling Chang</i>	
An Analysis of the RC4 Family of Stream Ciphers against Algebraic Attacks	73
<i>Kenneth Koon-Ho Wong, Gary Carter and Ed Dawson</i>	
Certificateless Key Agreement in the Standard Model	81
<i>Georg Lippold and Juanma González Nieto</i>	
Multicollision attacks on generalized iterated hash functions	93
<i>Kimmo Halunen, Juha Kortelainen and Tuomas Kortelainen</i>	

Author Index 101

Preface

The Australasian Information Security Conference (AISC) 2010 was held on the 19th and 20th January 2010 in Brisbane, Australia, as a part of the Australasian Computer Science Week 2010. AISC grew out of the Australasian Information Security Workshop and officially changed the name to Australasian Information Security Conference in 2008. The main aim of the AISC is to provide a venue for Australasian and other researchers to present their work on all aspects of information security and promote collaboration between academic and industrial researchers working in this area.

This year we received 21 submissions from Australia, Brazil, Finland, Japan, Korea, New Zealand, Singapore and United States. After a thorough refereeing process we accepted 10 papers for presentation at AISC 2010. We extend our thanks to all the AISC 2010 authors for their quality submissions and all the members of the Program Committee and additional referees for their expert reviews.

Following AISC tradition from previous years, we have selected a paper for the Best Student Paper Prize. Papers can be considered for this award only if the major contribution is due to a student author, who must be the first author of the paper. This year the award went to Farzad Salim from the Information Security Institute, Queensland University of Technology for the paper “An Administrative Model for UCON” by Farzad Salim, Jason Reid and Ed Dawson. Our hearty congratulations to Farzad and his co-authors on this fine achievement!

The invited keynote address for AISC 2010 was presented by Ed Dawson. We are very grateful to Ed for delivering the lecture on “Information Sharing in the 21st Century: Progress and Challenges”.

As part of AISC 2010 a workshop on Denial of Service was held during the first afternoon. Our sincere thanks go to George Mohay for organising this workshop and to the invited speakers, Graham Ingram, Craig Lawson, Alan Tickle and Desmond Schmidt. A panel followed the invited talks for which the invited speakers were joined by Bill Caelli for a stimulating discussion.

Special thanks go to Gleb Sechenov for his excellent work on maintaining the AISC 2010 website. We used Easychair software to manage the AISC submissions and reviews. We found this software very helpful and easy to use and we thank the maintainers of the service for this opportunity.

Last but not least we extend our gratitude to the ACSW 2010 chair Wayne Kelly and other members of the organising committee for their hard work and their continuous and invaluable support throughout the preparation of the conference.

Colin Boyd
Queensland University of Technology
Willy Susilo
University of Wollongong
AISC 2010 Programme Chairs
January 2010

Programme Committee

Chairs

Colin Boyd Queensland University of Technology, Australia

Willy Susilo University of Wollongong, Australia

Members

Joonsang Baek, Institute for Infocomm Research, Singapore

Lynn Batten, Deakin University, Australia

Ljiljana Brankovic, University of Newcastle, Australia

Andrew Clark, Queensland University of Technology, Australia

Kathy Horadam, RMIT University, Australia

Chris Mitchell, Royal Holloway, University of London, UK

Paul Montague, DSTO, Australia Yi Mu, University of Wollongong, Australia

C. Pandu Rangan, IIT-Madras, India

Josef Pieprzyk, Macquarie University, Australia

Rei Safavi-Naini, University of Calgary, Canada

Chris Steketee, University of South Australia, Australia

Clark Thomborson, The University of Auckland, New Zealand

Huaxiong Wang, Nanyang Technology University, Singapore)

Duncan S. Wong, City University of Hong Kong, Hong Kong SAR, China

Organising Committee

Co-Chairs

Dr. Wayne Kelly
Prof. Mark Looi

Budget and Facilities

Mr. Malcolm Corney

Catering and Booklet

Dr. Diane Corney

Sponsorship and Web

Dr. Tony Sahama

Senior Advisors

Prof. Colin Fidge
Prof. Kerry Raymond

Finance and Travel

Ms. Therese Currell
Ms. Carol Richter

Registration

Mr. Matt Williams

DVD and Signage

Mr. Matthew Brecknell

Satchels and T-shirts

Ms. Donna Teague

Welcome from the Organising Committee

On behalf of the Australasian Computer Science Week 2010 (ACSW2010) Organising Committee, we welcome you to this year's event hosted by the Queensland University of Technology (QUT). Striving to be a "University for the Real World" our research and teaching has an applied emphasis. QUT is one of the largest producers of IT graduates in Australia with strong linkages with industry. Our courses and research span an extremely wide range of information technology, everything from traditional computer science, software engineering and information systems, to games and interactive entertainment.

We welcome delegates from over 21 countries, including Australia, New Zealand, USA, Finland, Italy, Japan, China, Brazil, Canada, Germany, Pakistan, Sweden, Austria, Bangladesh, Ireland, Norway, South Africa, Taiwan and Thailand. We trust you will enjoy both the experience of the ACSW 2010 event and also get to explore some of our beautiful city of Brisbane. At Brisbane's heart, beautifully restored sandstone buildings provide a delightful backdrop to the city's glass towers. The inner city clusters around the loops of the Brisbane River, connected to leafy, open-skied suburban communities by riverside bikeways. QUT's Garden's Point campus, the venue for ACSW 2010, is on the fringe of the city's botanical gardens and connected by the Goodwill Bridge to the Southbank tourist precinct.

ACSW2009 consists of the following conferences:

- Australasian Computer Science Conference (ACSC) (Chaired by Bernard Mans and Mark Reynolds)
- Australasian Computing Education Conference (ACE) (Chaired by Tony Clear and John Hamer)
- Australasian Database Conference (ADC) (ADC) (Chaired by Heng Tao Shen and Athman Bouguet-taya)
- Australasian Information Security Conference (AISC) (Chaired by Colin Boyd and Willy Susilo)
- Australasian User Interface Conference (AUIC) (Chaired by Christof Lutteroth and Paul Calder)
- Australasian Symposium on Parallel and Distributed Computing (AusPDC) (Chaired by Jinjun Chen and Rajiv Ranjan)
- Australasian Workshop on Health Informatics and Knowledge Management (HIKM) (Chaired by Anthony Maeder and David Hansen)
- Computing: The Australasian Theory Symposium (CATS) (Chaired by Taso Viglas and Alex Potanin)
- Asia-Pacific Conference on Conceptual Modelling (APCCM) (Chaired by Sebastian Link and Aditya Ghose)
- Australasian Computing Doctoral Consortium (ACDC) (Chaired by David Pearce and Rachel Cardell-Oliver).

The nature of ACSW requires the co-operation of numerous people. We would like to thank all those who have worked to ensure the success of ACSW2010 including the Organising Committee, the Conference Chairs and Programme Committees, our sponsors, the keynote speakers and the delegates. Special thanks to Justin Zobel from CORE and Alex Potanin (co-chair of ACSW2009) for his extensive advice and assistance. If ACSW2010 is run even half as well as ACSW2009 in Wellington then we will have done well.

Dr Wayne Kelly and Professor Mark Looi

Queensland University of Technology

ACSW2010 Co-Chairs

January, 2010

CORE - Computing Research & Education

CORE welcomes all delegates to ACSW2010 in Brisbane. CORE, the peak body representing academic computer science in Australia and New Zealand, is responsible for the annual ACSW series of meetings, which are a unique opportunity for our community to network and to discuss research and topics of mutual interest. The original component conferences ACSC, ADC, and CATS, which formed the basis of ACSWin the mid 1990s now share the week with seven other events, which build on the diversity of the Australasian computing community.

In 2010, we have again chosen to feature a small number of plenary speakers from across the discipline: Andy Cockburn, Alon Halevy, and Stephen Kisely. I thank them for their contributions to ACSW2010. I also thank the keynote speakers invited to some of the individual conferences. The efforts of the conference chairs and their program committees have led to strong programs in all the conferences again, thanks. And thanks are particularly due to Wayne Kelly and his colleagues for organising what promises to be a strong event.

In Australia, 2009 saw, for the first time in some years, an increase in the number of students choosing to study IT, and a welcome if small number of new academic appointments. Also welcome is the news that university and research funding is set to rise from 2011-12. However, it continues to be the case that per-place funding for computer science students has fallen relative to that of other physical and mathematical sciences, and, while bodies such as the Australian Council of Deans of ICT seek ways to increase student interest in the area, more is needed to ensure the growth of our discipline.

During 2009, CORE continued to work on journal and conference rankings. A key aim is now to maintain the rankings, which are widely used overseas as well as in Australia. Management of the rankings is a challenging process that needs to balance competing special interests as well as addressing the interests of the community as a whole. ACSW2010 includes a forum on rankings to discuss this process. Also in 2009 CORE proposed a standard for the undergraduate Computer Science curriculum, with the intention that it be used for accreditation of degrees in computer science.

CORE's existence is due to the support of the member departments in Australia and New Zealand, and I thank them for their ongoing contributions, in commitment and in financial support. Finally, I am grateful to all those who gave their time to CORE in 2009; in particular, I thank Gill Dobbie, Jenny Edwards, Alan Fekete, Tom Gedeon, Leon Sterling, and the members of the executive and of the curriculum and ranking committees.

Justin Zobel

President, CORE
January, 2010

ACSW Conferences and the Australian Computer Science Communications

The Australasian Computer Science Week of conferences has been running in some form continuously since 1978. This makes it one of the longest running conferences in computer science. The proceedings of the week have been published as the *Australian Computer Science Communications* since 1979 (with the 1978 proceedings often referred to as *Volume 0*). Thus the sequence number of the Australasian Computer Science Conference is always one greater than the volume of the Communications. Below is a list of the conferences, their locations and hosts.

2011. Volume 33. Host and Venue - Curtin University of Technology, Perth, WA.

2010. Volume 32. Host and Venue - Queensland University of Technology, Brisbane, QLD.

2009. Volume 31. Host and Venue - Victoria University, Wellington, New Zealand.

2008. Volume 30. Host and Venue - University of Wollongong, NSW.

2007. Volume 29. Host and Venue - University of Ballarat, VIC. First running of HDKM.

2006. Volume 28. Host and Venue - University of Tasmania, TAS.

2005. Volume 27. Host - University of Newcastle, NSW. APBC held separately from 2005.

2004. Volume 26. Host and Venue - University of Otago, Dunedin, New Zealand. First running of APCCM.

2003. Volume 25. Hosts - Flinders University, University of Adelaide and University of South Australia. Venue - Adelaide Convention Centre, Adelaide, SA. First running of APBC. Incorporation of ACE. ACSAC held separately from 2003.

2002. Volume 24. Host and Venue - Monash University, Melbourne, VIC.

2001. Volume 23. Hosts - Bond University and Griffith University (Gold Coast). Venue - Gold Coast, QLD.

2000. Volume 22. Hosts - Australian National University and University of Canberra. Venue - ANU, Canberra, ACT. First running of AUC.

1999. Volume 21. Host and Venue - University of Auckland, New Zealand.

1998. Volume 20. Hosts - University of Western Australia, Murdoch University, Edith Cowan University and Curtin University. Venue - Perth, WA.

1997. Volume 19. Hosts - Macquarie University and University of Technology, Sydney. Venue - Sydney, NSW. ADC held with DASFAA (rather than ACSW) in 1997.

1996. Volume 18. Host - University of Melbourne and RMIT University. Venue - Melbourne, Australia. CATS joins ACSW.

1995. Volume 17. Hosts - Flinders University, University of Adelaide and University of South Australia. Venue - Glenelg, SA.

1994. Volume 16. Host and Venue - University of Canterbury, Christchurch, New Zealand. CATS run for the first time separately in Sydney.

1993. Volume 15. Hosts - Griffith University and Queensland University of Technology. Venue - Nathan, QLD.

1992. Volume 14. Host and Venue - University of Tasmania, TAS. (ADC held separately at La Trobe University).

1991. Volume 13. Host and Venue - University of New South Wales, NSW.

1990. Volume 12. Host and Venue - Monash University, Melbourne, VIC. Joined by Database and Information Systems Conference which in 1992 became ADC (which stayed with ACSW) and ACIS (which now operates independently).

1989. Volume 11. Host and Venue - University of Wollongong, NSW.

1988. Volume 10. Host and Venue - University of Queensland, QLD.

1987. Volume 9. Host and Venue - Deakin University, VIC.

1986. Volume 8. Host and Venue - Australian National University, Canberra, ACT.

1985. Volume 7. Hosts - University of Melbourne and Monash University. Venue - Melbourne, VIC.

1984. Volume 6. Host and Venue - University of Adelaide, SA.

1983. Volume 5. Host and Venue - University of Sydney, NSW.

1982. Volume 4. Host and Venue - University of Western Australia, WA.

1981. Volume 3. Host and Venue - University of Queensland, QLD.

1980. Volume 2. Host and Venue - Australian National University, Canberra, ACT.

1979. Volume 1. Host and Venue - University of Tasmania, TAS.

1978. Volume 0. Host and Venue - University of New South Wales, NSW.

Conference Acronyms

ACDC	Australasian Computing Doctoral Consortium
ACE	Australasian Computer Education Conference
ACSC	Australasian Computer Science Conference
ACSW	Australasian Computer Science Week
ADC	Australasian Database Conference
AISC	Australasian Information Security Conference
AUIC	Australasian User Interface Conference
APCCM	Asia-Pacific Conference on Conceptual Modelling
AusPDC	Australasian Symposium on Parallel and Distributed Computing (replaces AusGrid)
CATS	Computing: Australasian Theory Symposium
HIKM	Australasian Workshop on Health Informatics and Knowledge Management

Note that various name changes have occurred, which have been indicated in the Conference Acronyms sections in respective CRPIT volumes.

ACSW and AISC 2010 Sponsors

We wish to thank the following sponsors for their contribution towards this conference.



CORE - Computing Research and Education,
www.core.edu.au



CEED,
www.corptech.com.au



Queensland University of Technology,
www.qut.edu.au



CSIRO ICT Centre,
www.csiro.au/org/ict.html



AUSTRALIAN
COMPUTER
SOCIETY
Australian Computer Society,
www.acs.org.au



SAP Research,
www.sap.com/about/company/research



QUT Information Security Institute,
www.isi.qut.com

KEYNOTE TALK

Information Sharing in the 21st century: Progress and Challenges

Ed Dawson

Jason Reid

Farzad Salim

Mark Burdon

Information Security Institute,
Queensland University of Technology,
GPO Box 2434, Brisbane 4001, Queensland, Australia

Abstract

With the increasing threat of cyber and other attacks on critical infrastructure, federal governments throughout the world have been organizing industry to share information on possible threats. In Australia the Office of the Attorney General has formed Trusted Information Sharing Networks (TISN) for the various critical industries such as banking and electricity. Currently the majority of information for a TISN is shared at physical meetings. To meet cyber threats there are clearly limitations to physical meetings. Many of these limitations can be overcome by the creation of a virtual information sharing network (VISN). However there are many challenges to overcome in the design of a VISN both from a policy and technical viewpoint. We shall discuss some of these challenges in this talk.

CONTRIBUTED PAPERS

Secure Coprocessor-based Private Information Retrieval without Periodical Preprocessing

Peishun Wang¹Huaxiong Wang^{2,3}Josef Pieprzyk³

¹ School of Computer Science and Software Engineering
University of Wollongong, NSW 2522, Australia
Email: peishun@uow.edu.au

² Center for Advanced Computing – Algorithms and Cryptography, Department of Computing
Macquarie University, NSW 2109, Australia
Email: hwang, josef@ics.mq.edu.au

³ Division of Mathematical Sciences, School of Physical and Mathematical Sciences
Nanyang Technological University, Singapore

Abstract

Early works on Private Information Retrieval (PIR) focused on minimizing the necessary communication overhead. They seemed to achieve this goal but at the expense of query response time. To mitigate this weakness, protocols with secure coprocessors were introduced. They achieve optimal communication complexity and better online processing complexity. Unfortunately, all secure coprocessor-based PIR protocols require heavy periodical preprocessing. In this paper, we propose a new protocol, which is free from the periodical preprocessing while offering the optimal communication complexity and almost optimal online processing complexity. The proposed protocol is proven to be secure.

Keywords: Private information retrieval, secure coprocessor.

1 Introduction

A private information retrieval (PIR) protocol allows a user to retrieve a data item of her choice from a database such that the database server does not learn any information on the identity of the item fetched. The problem was formulated by Chor et al. (Chor et al. 1998), and has attracted a considerable amount of attention. The efficiency of PIR protocols is typically measured by their communication complexity and computation overhead necessary to answer a query. Early works on PIR protocols (for both information theoretic and computational models) have been mainly focused on minimizing the communication complexity between the user and the database server. Many proposed PIR protocols (see, for example, (Ambainis 1997, Beimel et al. 2002, Chang 2004, Chor & Gilboa 1997, Gentry & Ramzan 2005, Woodruff & Yekhanin 2005)) succeeded in achieving this goal. However, these protocols have very high computation overheads, requiring the computation on the entire database in order to retrieve a single bit. This results in excessive query response time, and makes them impractical. Sion and Carbunar (Sion & Carbunar 2007) showed that the naive solution (*i.e.*, downloading the whole database)

is more efficient than a carefully designed PIR protocol with sophisticated mathematical computation. To address this problem, several attempts have been made, and one of the most efficient solutions is based on a tamper-proof *secure coprocessor* (SC), which prevents anybody from accessing its memory from outside even if the adversary has direct physical access to the device (Smith et al. 1998). Being more specific, the database server installs a secure coprocessor, which works as a user extension at the server side. If the internal memory space of secure coprocessor was large enough to hold the entire database, the PIR problem would be solved easily. A user simply applies existing network protocols like HTTP and SSL to negotiate a secure session with the coprocessor, and makes a query. Since SC is physically protected, no one including the server should be able to observe what the query is. Unfortunately, its internal memory can only hold a fixed and small number of records at a time. Thus, secure coprocessor-based PIR protocols aim to provide private access to a large database while using only a small amount of coprocessor memory space.

Some secure coprocessor-based PIR protocols were proposed in (Asonov & Freytag 4/2002, 5/2002, Iliev & Smith 2003, 2004, 2005, Smith & Safford 2000, 2001, Wang et al. 2006, Yang et al. 2008). They achieved optimal communication complexity and good online processing complexity. However, all these protocols need a heavy preprocessing that periodically shuffles the database. In this paper, we propose a protocol without periodical preprocessing.

Our Contributions. We propose a new secure coprocessor-based PIR protocol that works in the two stages: offline shuffling and online retrieving. During the offline shuffle, the secure coprocessor double-encrypts all records and permutes the original database. During the online retrieval, the secure coprocessor usually reads two records from the shuffled database and writes two records back. We prove the security of the protocol. Unlike the previously published coprocessor-based PIR protocols, the proposed protocol uses two new ideas, namely, double-encryption and twin-writing, and consequently removes the need for periodical preprocessing. The analysis of efficiency indicates that the performance of the proposed protocol is better than the performance of previous coprocessor-based PIR protocols.

Organization. Section 2 reviews the related work. In Section 3 we provide the model. In Section 4 we construct a new protocol and show its security. In

Copyright ©2010, Australian Computer Society, Inc. This paper appeared at Australasian Information Security Conference (AISC 2010), Brisbane, Australia. Conferences in Research and Practice in Information Technology, Vol. 105. Colin Boyd and Willy Susilo, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

Section 5 we analyze the performance. Finally Section 6 concludes the work.

2 Related Work

Smith and Safford (Smith & Safford 2000, 2001) came up with the idea of using a secure coprocessor (SC) for PIR. In their works, SC preprocesses the database, *i.e.*, it shuffles the records offline before the PIR protocol starts. More precisely, SC reads entire database n times record by record, and each time SC leaves a record in its secure memory and encrypts the record using a secret key known to SC but not to the server, then SC writes the encrypted record in the shuffled database. During the phase of online retrieval, when SC receives a query from a user, it reads through the entire shuffled database, and keeps the desired record in its internal memory. Then SC decrypts the record and sends it via a secure channel to the user. Obviously, the communication complexity is optimal, but the online computation complexity is $O(n)$ and the offline preprocessing complexity is $O(n^2)$.

To minimize the online computation, Asonov and Freytag (Asonov & Freytag 4/2002) modified Smith and Safford's protocol as follows. To answer the first query, SC accesses the desired encrypted record directly instead of reading the entire database. The encrypted record is decrypted inside the SC and sent to the user via a secure channel. To answer the k^{th} ($k \geq 2$) query, SC has to read the $k - 1$ previously accessed records first, then the desired one. In case the k^{th} query requests the same record as one of the $k - 1$ previous queries, SC reads a random (previously unread) record. Evidently, SC has to keep track of the accessed records. The database server can decide at which point $m = \max(k)$ ($1 \leq m \leq n$) to stop and to switch to shuffle the database again. Since m is a constant independent of n , if the maximal allowed query response time is fixed, m can be chosen without considering preprocessing. Thus, the server processes $O(1)$ records online in order to answer each query. Their protocol improves the online computation complexity from $O(n)$ to $O(n^{1/2})$ (even $O(1)$), and retains the optimal communication complexity. But the offline preprocessing complexity is still $O(n^2)$, and the reshuffling takes place when a fixed number of queries have been answered.

The above two protocols would be optimal from the user point of view, but they are still quite expensive for the server. For example, assuming that accessing one database record takes 0.01 second for SC and $n = 10000$, then we need $n^2 * 0.01 \approx 2$ weeks to prepare one shuffled database and an optimal trade-off parameter $m = 141$. This means that we need to reshuffle the database once per 141 retrievals. To reduce the offline preprocessing, Asonov and Freytag (Asonov & Freytag 5/2002) modified their protocol. The server first splits each record in the database into p equal parts. As the result, the database is transformed into p share databases. Then SC shuffles all the share databases based on the same algorithm. The authors showed that for an optimal p , it is possible to reduce the preprocessing complexity to $O(n^{1.5})$. This means that $n^{1.5} * 0.01 \approx 3$ hours are needed to prepare a shuffled database. Iliev and Smith used the Beneš permutation networks as the shuffling algorithm to reduce the complexity of database shuffling. They presented protocols (Iliev & Smith 2003, 2005) with the preprocessing complexity of $O(n \lg n)$. Under the same condition, the preparation of one shuffled database needs $n \lg n * 0.01 \approx 23$ minutes. They modified their protocols and proposed a new variant

(Iliev & Smith 2004), which requires a smaller internal storage space of size $O(\lg n)$ for shuffling. Furthermore, Wang *et al.* (Wang et al. 2006) and Yang *et al.* (Yang et al. 2008) constructed protocols with new shuffling methods and reduced the computational cost for a query. Unfortunately, the problem of periodical switching to a new shuffled database still exists in these protocols.

3 Model

Throughout this paper, we use the following notation.

Let $a \stackrel{R}{\leftarrow} A$ denote choosing an element a uniformly at random from the set A . For an integer n , $[n]$ denotes the set of integers $\{1, 2, \dots, n\}$. We write $x||y$ to denote the concatenation of the bit-strings x, y . By default, $\lg k \stackrel{def}{=} \lceil \log_2 k \rceil$. For a database DB with n records, DB_i ($i \in [n]$) denotes the i^{th} record in DB . As in (Asonov & Freytag 5/2002), we assume that it takes several orders of magnitude longer to operate on data in the external storage than to access main memory. Thus, the preprocessing complexity and online processing complexity of a PIR protocol are measured by the number of records accessed on the external storage, *i.e.*, by the number of I/Os.

3.1 The Model

Let's briefly recall the model of secure coprocessor-based PIR. It consists of a single server – the host H , which is connected with a secure coprocessor SC . A database DB is stored on a suitable high-performance storage medium that is a part of H and is separated from SC . DB has n records. If records are not all of the same length, each record is padded up to the same size (the padding contents are determined by the application). Before online queries start, DB is permuted to a new database D . To retrieve a record DB_i ($i \in [n]$), a user sends SC a query specifying the index i via a secure channel. SC then interacts with H , which accesses D instead of DB , to reply.

The adversary in the model attempts to derive information from the PIR protocol execution. Possible adversaries include outside attackers and the host, where the latter is able to not only observe all I/O operations performed for queries, but also can make queries as a legitimate user. The aim of SC is to retrieve a requested record, while hiding the identity of the record from any adversary. This means that the adversary should not be able to determine, which record in the original database has been or has not been requested by a user.

We follow three terms introduced in (Wang et al. 2006), access pattern, stained query and clean query. An access pattern for a time period is an array of the records of D read and written in the period, where the records are arranged chronologically. A stained query means that the plaintext and the index of the desired record are known to the adversary \mathcal{A} without observing any access pattern. A clean query is the one the adversary \mathcal{A} does not know before observing access patterns.

Let $Pr(Q \simeq DB_j)$ denote the probability of the event that the record DB_j is the one desired by the query Q . Note that, in information theoretic or computational PIR protocols, an adversary does not know anything about the queried record by observing access patterns, no matter whether the query is stained or clean. However, in all existing secure coprocessor-based PIR protocols, for a stained query, an adversary might know that the record read from D is the queried one, but for a clean query, she does not. Intuitively, the queried record is selected in the black box

(i.e., internal memory of SC). Now if every record of the database DB is in the SC 's internal memory with nonzero probability, then any adversary cannot learn which record is the queried one. Based on this observation, we give a definition as follows.

Definition 3.1 A SC -based PIR protocol is secure, if for a clean query Q and any possible access pattern $Access$, the following relation holds

$$0 < Pr(Q \simeq DB_j | Access) \leq 1/M, \quad \forall j \in [n],$$

where M is the maximum number of records kept in the SC 's internal memory.

Notice that the above definition of security is different from the security definition in the information theoretic or computational PIR protocols, which guarantees that $Pr(Q \simeq DB_j | Access) = 1/n$ for any $j \in [n]$. However, in some coprocessor-based PIR protocols, the probability $Pr(Q \simeq DB_j | Access)$ is not uniformly distributed for all $j \in [n]$. For example, in the protocols from (Asonov & Freytag 5/2002, Iliev & Smith 2005, Wang et al. 2006), if the l^{th} query ($1 \leq l \leq M$) is clean, from the view of an adversary, $Pr(Q \simeq DB_j | Access) = \frac{1}{l(n-l+1)}$ for a record $DB_j \notin \{DB_{i_1}, DB_{i_2}, \dots, DB_{i_{l-1}}\}$, where $\{DB_{i_1}, DB_{i_2}, \dots, DB_{i_{l-1}}\}$ are the records read into SC 's internal memory in the previous $l-1$ queries. A protocol satisfying the above definition is called a non-perfect PIR protocol. In the above definition, $Pr(Q \simeq DB_j | Access) = 1/M$ means that DB_j is certainly in SC 's internal memory, and $Pr(Q \simeq DB_j | Access) > 0$ for all $j \in [n]$ implies that an adversary does not know, which record of DB is not in SC 's internal memory.

4 The Proposed Protocol

As the previous works in (Asonov & Freytag 4/2002, 5/2002, Iliev & Smith 2003, 2004, 2005, Wang et al. 2006, Yang et al. 2008), our protocol consists of two phases: offline preprocessing and online retrieval. The database is permuted offline before any query starts. The offline preprocessing (permutation) is executed once only.

During the offline preprocessing phase, the coprocessor SC permutes the database DB into a new database D . Note that z ($1 < z \ll n$) records are kept in SC 's internal memory C , and $n-z$ double-encrypted records are stored in D . SC maintains a table T in its internal memory C . The table T contains n rows and each row T_i ($i \in [n]$) keeps track of the record DB_i . More precisely, each row T_i consists of two fields: a single bit flag T_{i-flag} and an index $T_{i-index}$ that consists of $\lg(n-z)$ bits. There are three possible cases:

1. $T_{i-flag} = 0$ and $T_{i-index} = 0$ – this means that the i^{th} record DB_i is in the internal memory C (It does not exist in the database D);
2. $T_{i-flag} = 0$ and $T_{i-index} = k$ ($k \in [n-z]$) – this means that DB_i is stored in D as the record D_k and this record has never been accessed;
3. $T_{i-flag} = 1$ and $T_{i-index} = k$ ($k \in [n-z]$) – this means that DB_i is stored in D as the record D_k and this record has already been accessed.

During the online retrieval phase, on receiving a query on DB_i from a user, SC locates DB_i by checking T_i . There are three possible cases:

1. If DB_i is in C , then SC reads two records randomly chosen from D into C , where the first read record is accessed, and the second is not.
2. If DB_i is in D and unaccessed, then SC reads one random accessed record and $D_{T_{i-index}}$ into C .
3. If DB_i is in D and accessed, then SC reads $D_{T_{i-index}}$ and one random unaccessed record from D into C .

After sending the desired record DB_i to the user, SC chooses two records at random from C and writes the two records into D at the positions from which the previous two records were taken. Finally, SC changes the values of the corresponding rows in T . For the first query, SC reads/writes one record from/into D , and for any other query, SC reads two records from D , which was called *twin-reading* in (Yang et al. 2008), and writes two records into D , which we call *twin-writing*. After $n-z$ queries, all records are treated as the unaccessed again. Note that there are always z records stored in C after SC finishes an online query.

4.1 Offline Preprocessing

SC applies the approach from (Iliev & Smith 2005) to permute the database DB , but the writing operation (i.e., the record DB_i ($i \in [n]$) of the database DB is permuted to the record D_j ($j \in [n-z]$) of the database D) is modified as follows.

Step 1 SC generates two secret keys, k_0 and k_1 , and creates a table T of $n \times (1 + \lg(n-z))$ bits, which has n rows $T_i = T_{i-flag} || T_{i-index}$ ($i = 1, \dots, n$), where T_{i-flag} is a 1-bit flag and $T_{i-index}$ is $\lg(n-z)$ bits for storing an index. Each row T_i is initialized to the value 0, and the table is stored in C .

Step 2 SC uses k_1 to encrypt every record DB_i , appends k_1 to the encrypted record, and then encrypts it again under k_0 .

Step 3 The encrypted-appended-encrypted (called double-encrypted) version of the record DB_i is written into the j^{th} entry in D as the record D_j ;

Step 4 SC keeps $T_{i-flag} = 0$ and sets $T_{i-index} = j$, and then rewrites T_i in the table T .

Step 5 During the process of permutation, SC selects z records $\{DB_{i_j}\}_{j=1}^z$ ($i_j \in [n]$) uniformly at random and stores them and k_0, k_1 in C .

4.2 Online Retrieval

To present the protocol in a convenient way, let \mathcal{G} denote a secret key generator that takes an old key k_s as input to generate a new secret key k_{s+1} . Now we describe two algorithms: **Reading**(j, k_0) and **Writing**(j, k_0, k_s, DB_r).

Reading(j, k_0) is a deterministic algorithm, which takes as input an index j ($j \in [n-z]$) and the secret key k_0 , and outputs the corresponding record DB_l ($l \in [1, n]$) (i.e., $T_{l-index} = j$). It works according to the following steps.

1. Reads the record D_j from D into C ;
2. Decrypts D_j with k_0 , gets a data and the appended secret key k_s ($s \geq 1$), and then decrypts the data with k_s to get the corresponding record DB_l .

Writing(j, k_0, k_s, DB_r) is a deterministic algorithm, which takes as input an index j ($j \in [n-z]$), two secret keys k_0 and k_s and a record DB_r ($r \in [n]$), and outputs a record D_j . It goes through the following steps.

1. Encrypts DB_r with k_s , appends k_s to the encrypted record, and then encrypts the appended-encrypted record with k_0 ;
2. Writes the double-encrypted version of the record DB_r at the j^{th} entry in D as the record D_j ;

During the phase of online retrieval, on receiving the t^{th} query Q_t on the record DB_i from a user, if $t = x(n-z) + 1$ for some $x \in \{0, 1, 2, \dots\}$, SC carries out the **Algorithm 1** (below); otherwise, executes the **Algorithm 2** (below).

Algorithm 1: Single-Reading & Single-Writing

```

1: check  $T_i$  in  $T$ ;
2: if  $T_i = T_{i-flag} || T_{i-index} = 0$  then
3:   choose  $T_j \xleftarrow{R} T$  such that  $T_j \neq 0$ ;
4:    $l = j$ 
5: else
6:    $l = i$ 
7: endif
8: execute the algorithm Reading( $T_{l-index}, k_0$ ) to
   put a record  $DB_l$  into  $C$ ;
9: send  $DB_l$  to the user as the answer to the query;
10: take a record  $DB_r \xleftarrow{R} C$ ;
11:  $k_{t+1} = \mathcal{G}(k_t)$ ;
12: delete  $k_t$ ;
13: execute Writing( $T_{l-index}, k_0, k_{t+1}, DB_r$ );
14: if  $r \neq l$  then
15:    $T_{r-index} = T_{l-index}$ ;
16:   reset  $T_l = 0$ ;
17: endif
18: set  $T_{r-flag} = 1$ ;
19: set  $T_r$  according to the new values of  $T_{r-flag}$ 
   and  $T_{r-index}$ .
```

Algorithm 2: Twin-Reading & Twin-Writing

```

1: check  $T_i$  in  $T$ ;
2: if  $T_i = T_{i-flag} || T_{i-index} = 0$  then
3:   choose  $T_{j_1}, T_{j_2} \xleftarrow{R} T$  such that  $T_{j_1-flag} = 1$ ,
    $T_{j_2-flag} = 0$  and  $T_{j_2-index} \neq 0$ ;
4: else if  $T_{i-flag} = 0$  then
5:   choose  $T_{j_1} \xleftarrow{R} T$  such that  $T_{j_1-flag} = 1$ ;
6:    $j_2 = i$ ;
7: else
8:   choose  $T_{j_2} \xleftarrow{R} T$  such that  $T_{j_2-flag} = 0$ 
   and  $T_{j_2-index} \neq 0$ ;
9:    $j_1 = i$ ;
10: endif
11: endif
12: execute the algorithm Reading( $T_{j_1-index}, k_0$ ) to
   put a record  $DB_{j_1}$  into  $C$ ;
13: execute the algorithm Reading( $T_{j_2-index}, k_0$ ) to
   put a record  $DB_{j_2}$  into  $C$ ;
14: send  $DB_i$  to the user as the answer to the query;
15: take two records  $DB_r, DB_s \xleftarrow{R} C$ ;
16:  $k_{t+1} = \mathcal{G}(k_t)$ ;
17: delete  $k_t$ ;
18: execute Writing( $T_{j_1-index}, k_0, k_{t+1}, DB_r$ );
19: execute Writing( $T_{j_2-index}, k_0, k_{t+1}, DB_s$ );
20: take  $T_{r-flag} = 1$  and  $T_{r-index} = T_{j_1-index}$ , and
   reset  $T_r$ ;
21: take  $T_{s-flag} = 1$  and  $T_{s-index} = T_{j_2-index}$ , and
   reset  $T_s$ ;
22: if  $r \neq j_1$  then reset  $T_{j_1} = 0$ ;
23: endif
24: if  $s \neq j_2$  then reset  $T_{j_2} = 0$ ;
25: endif
26: if  $t = x(n-z)$  for some integer  $x$  then
   reset  $T_{l-flag} = 0$  for all  $l \in [n]$ ;
27: endif
```

Now we consider the security of proposed protocol.

Theorem 1 *The proposed SC-based PIR protocol is secure according to Definition 3.1.*

Proof:

We use induction on the number N of queries to prove the security. Let $Pr(DB_i \simeq \{D_{j_1}, \dots, D_{j_m}\})$ ($m \in [n-z]$) denote the probability of the event that DB_i is permuted to one of the records $\{D_{j_1}, \dots, D_{j_m}\}$, $Pr(DB_i \simeq C)$ denote the probability of the event that DB_i is in C , and $Access(Q)$ denote the access pattern (including reading and writing patterns) for the query Q . Note that, in the proposed protocol, the maximum number of records in C is $M = z + 2$.

CASE $N = 1$: the 1st online query Q_1 on DB_{i_1} .

$Access(Q_1)$:

1. Reading Pattern: SC reads a record D_{x_1} from D into C .
2. Writing Pattern: SC chooses a random record DB_r from C and executes the algorithm **Writing**(x_1, k_0, k_1, DB_r) to write a record D_{x_1} into D . Note that the currently written record D_{x_1} is different from the just read record D_{x_1} even though they are permuted from an identical record in DB because of using different encryption keys.

Analysis :

1. Consider that the query is clean. D is permuted from DB in an oblivious way used in (Iliev & Smith 2005) during the offline preprocessing. According to the proof in (Iliev & Smith 2005), although the adversary \mathcal{A} knows that D_{x_1} is read into C , she does not know which record in DB is or is not permuted to D_{x_1} . The original records in C are randomly and secretly selected by SC . So, from the view of \mathcal{A} , every record in DB can be located in C with the probability of $\frac{z+1}{n}$. Therefore,

$$Pr(Q_1 \simeq DB_j) = \frac{1}{z+1} \cdot \frac{z+1}{n} = \frac{1}{n}, \quad \forall j \in [n].$$

2. By observing the access pattern $Access(Q_1)$, \mathcal{A} knows the following information about the queried record DB_{i_1} ,

$$Pr(DB_{i_1} \simeq \{D_{x_1}\}) = 1/(z+1) \text{ and } Pr(DB_{i_1} \simeq C) = z/(z+1).$$

However, if the query is clean, the information gives \mathcal{A} nothing to identify which record in DB is or not the queried record DB_{i_1} .

Conclusion : For a clean query Q_1 , we have

$$0 < Pr(Q_1 \simeq DB_j | Access) \leq 1/M, \quad \forall j \in [n],$$

where $Access = \{Access(Q_1)\}$ is the access pattern.

CASE $N = 2$: the 2nd online query Q_2 on DB_{i_2} .

$Access(Q_2)$:

1. Reading Pattern: SC reads the record D_{x_1} and another record D_{x_2} ($x_2 \neq x_1$) from D into C .

2. Writing Pattern: SC chooses two random records $\{DB_r, DB_s\}$ from C and executes **Writing** (x_1, k_0, k_2, DB_r) and **Writing** (x_2, k_0, k_2, DB_s) to write the records $\{D_{x_1}, D_{x_2}\}$ into D .

Analysis :

1. Consider that the query is clean. Whatever the first query is, D_{x_1} is read back into C . That means, DB_{i_1} is certainly in C (i.e. with the probability of 1). In addition, the records in $D \setminus \{D_{x_1}\}$ are permuted in an oblivious way during the offline preprocessing, \mathcal{A} does not know which record in $DB \setminus \{DB_{i_1}\}$ is or is not permuted to D_{x_2} . So, from the view of \mathcal{A} , any other record (except DB_{i_1}) in DB is located in C with the probability of $\frac{z+1}{n-1}$. Therefore,

$$\Pr(Q_2 \simeq DB_{i_1}) = \frac{1}{z+2}, \text{ and}$$

$$\Pr(Q_2 \simeq DB_j) = \frac{z+1}{(z+2)(n-1)}, \quad \forall j \in [n] \setminus \{i_1\}.$$

2. By observing the $Access(Q_1)$ and $Access(Q_2)$, \mathcal{A} knows the following information,

$$\Pr(DB_{i_1} \simeq \{D_{x_1}, D_{x_2}\}) = 2/(z+2) \text{ and}$$

$$\Pr(DB_{i_1} \simeq C) = z/(z+2),$$

$$\Pr(DB_{i_2} \simeq \{D_{x_1}, D_{x_2}\}) = 2/(z+2) \text{ and}$$

$$\Pr(DB_{i_2} \simeq C) = z/(z+2).$$

However, if the query $q = j$ ($j \in \{1, 2\}$) is clean, the information gives \mathcal{A} nothing to identify which record in DB is or not the queried record DB_{i_j} .

Conclusion : For a clean query Q_2 , we have

$$0 < \Pr(Q_2 \simeq DB_j | Access) \leq 1/M, \quad \forall j \in [n],$$

where $Access = \{Access(Q_1), Access(Q_2)\}$.

CASE $N = 3$: the 3rd online query Q_3 on DB_{i_3} .

$Access(Q_3)$:

1. Reading Pattern: SC reads a record $D_{y_3} \stackrel{R}{\leftarrow} \{D_{x_1}, D_{x_2}\}$ and another record D_{x_3} ($x_3 \notin \{x_1, x_2\}$) from D into C .
2. Writing Pattern: SC chooses two random records $\{DB_r, DB_s\}$ from C and executes **Writing** (y_3, k_0, k_3, DB_r) and **Writing** (x_3, k_0, k_3, DB_s) to write the records $\{D_{y_3}, D_{x_3}\}$ into D .

Analysis :

1. Consider that the query is clean.
 - (a) The record $D_{y_3} \in \{D_{x_1}, D_{x_2}\}$ is read back into C , this means, DB_{i_1} is in C with the probability of $\frac{z+1}{z+2}$, and so DB_{i_2} is.
 - (b) The records in $D \setminus \{D_{x_1}, D_{x_2}\}$ are permuted in an oblivious way during offline preprocessing, so \mathcal{A} does not know which record in $DB \setminus \{DB_{i_1}, DB_{i_2}\}$ is or is not permuted to D_{x_3} . Hence, from the view of \mathcal{A} , every record in $DB \setminus \{DB_{i_1}, DB_{i_2}\}$ is in C with the probability of $\frac{z+1}{n-2}$.

Therefore,

$$\Pr(Q_3 \simeq DB_j) = \frac{z+1}{(z+2)^2}, \quad \forall j \in \{i_1, i_2\}, \text{ and}$$

$$\Pr(Q_3 \simeq DB_j) = \frac{z+1}{(z+2)(n-2)}, \quad \forall j \in [n] \setminus \{i_1, i_2\}.$$

2. By observing all access patterns $Access(Q_1)$, $Access(Q_2)$ and $Access(Q_3)$, \mathcal{A} knows the following information,

$$\Pr(DB_{i_3} \simeq \{D_{y_3}, D_{x_3}\}) = 2/(z+2) \text{ and}$$

$$\Pr(DB_{i_3} \simeq C) = z/(z+2).$$

For a previous query $q = j$ ($j \in \{1, 2\}$), \mathcal{A} knows

$$\Pr(DB_{i_j} \simeq \{D_{y_3}, D_{x_3}\}) = \frac{2(z+1)}{(z+2)^2}, \text{ and}$$

$$\Pr(DB_{i_j} \simeq C) = \frac{z(z+1)}{(z+2)^2}.$$

However, if the query $q = j$ ($j \in \{1, 2, 3\}$) is clean, the information gives \mathcal{A} nothing to identify which one in DB is or not the queried record DB_{i_j} .

Conclusion : For a clean query Q_3 , we have

$$0 < \Pr(Q_3 \simeq DB_j | Access) \leq 1/M, \quad \forall j \in [n],$$

where $Access = \{Access(Q_1), Access(Q_2), Access(Q_3)\}$.

Induction Step: Suppose that the same conclusion holds for **CASE** $N = t$ ($2 < t$). This is, for the t^{th} query Q_t on DB_{i_t} , we have

Result 1 : For the current query Q_t , \mathcal{A} knows the following information,

$$\Pr(DB_{i_t} \simeq \{D_{y_t}, D_{x_t}\}) = 2/(z+2) \text{ and}$$

$$\Pr(DB_{i_t} \simeq C) = z/(z+2),$$

where $y_t \in \{x_1, \dots, x_{t-1}\}$.

For a previous query Q_j on DB_{i_j} ($j \in [t-1]$), \mathcal{A} knows

$$\Pr(DB_{i_j} \simeq \{D_{y_t}, D_{x_t}\}) > \frac{2z^{t-j}}{(z+2)^{t-j+1}} \text{ and}$$

$$\Pr(DB_{i_j} \simeq C) > \left(\frac{z}{z+2}\right)^{t-j+1}.$$

Result 2 : If the query Q_t is clean, it holds that

$$0 < \Pr(Q_t \simeq DB_j | Access) \leq 1/M, \quad \forall j \in [n],$$

where $Access = \{Access(Q_1), Access(Q_2), \dots, Access(Q_t)\}$.

Now, we proceed to prove that for $N = t+1$, i.e., the $(t+1)^{\text{th}}$ online retrieval, if the query is clean, by observing all access patterns, \mathcal{A} cannot determine which one in the original database DB is or not the queried record.

CASE $N = t+1$: the $(t+1)^{\text{th}}$ online query Q_{t+1} on $DB_{i_{t+1}}$.

$Access(Q_{t+1})$:

1. Reading Pattern: SC reads a record $D_{y_{t+1}} \stackrel{R}{\leftarrow} \{D_{x_1}, \dots, D_{x_t}\}$ and another record $D_{x_{t+1}}$ ($x_{t+1} \notin \{x_1, \dots, x_t\}$) from D into C .
2. Writing Pattern: SC chooses two random records $\{DB_r, DB_s\}$ from C and executes **Writing** $(y_{t+1}, k_0, k_{t+1}, DB_r)$ and **Writing** $(x_{t+1}, k_0, k_{t+1}, DB_s)$ to write the records $\{D_{y_{t+1}}, D_{x_{t+1}}\}$ into D .

Analysis :

1. Consider that the query is clean.

Case A : $D_{y_{t+1}}$ is read from $\{D_{x_1}, \dots, D_{x_{t-1}}\} \setminus \{D_{y_t}, D_{x_t}\}$ into C .

- (a) Let's treat the dataset $D \setminus \{D_{x_t}\}$ as D , then, this case is the same as the case of $N = t$. According to the **Result 2**, we have

$$0 < Pr(Q_{t+1} \simeq DB_j | Access) \leq 1/M,$$

$\forall j \in [n] \setminus \{i_t\}$,
where $Access = \{Access(Q_1),$
 $Access(Q_2), \dots, Access(Q_t)\}$.

- (b) According to the **Result 1**, DB_{i_t} is located in C with the probability of $z/(z+2)$, so we have

$$Pr(Q_{t+1} \simeq DB_{i_t}) = \frac{z}{(z+2)^2}.$$

Case B : $D_{y_{t+1}}$ is read from $\{D_{y_t}, D_{x_t}\}$ into C .

- (a) The records in $D \setminus \{D_{x_1}, \dots, D_{x_t}\}$ are permuted in an oblivious way during offline preprocessing, so \mathcal{A} does not know which record in $DB \setminus \{DB_{i_1}, \dots, DB_{i_t}\}$ is or is not permuted to $D_{x_{t+1}}$. Hence, from the view of \mathcal{A} , every record in $DB \setminus \{DB_{i_1}, \dots, DB_{i_t}\}$ is in C with the probability of $\frac{z+1}{n-(t \bmod n)}$.

- (b) The record $D_{y_{t+1}}$ is read from $\{D_{y_t}, D_{x_t}\}$ into C , according to the **Result 1**, we know that, DB_{i_t} is in C with the probability of $\frac{z+1}{z+2}$, and the record DB_{i_j} ($j = 1, \dots, t-1$), which is queried by the previous query Q_j , is in C with the probability of at least $\frac{z^{t-j}(1+z)}{(z+2)^{t-j+1}}$.

Therefore,

$$\begin{aligned} Pr(Q_{t+1} \simeq DB_{i_t}) &= \frac{z+1}{(z+2)^2}, \\ Pr(Q_{t+1} \simeq DB_{i_j}) &\geq \frac{z^{t-j}(1+z)}{(z+2)^{t-j+2}} \text{ and} \\ Pr(Q_{t+1} \simeq DB_{i_j}) &\leq \frac{1}{z+2}, \forall j \in [t-1], \\ Pr(Q_{t+1} \simeq DB_j) &= \frac{z+1}{(z+2)(n-(t \bmod n))}, \\ &\forall j \in [n] \setminus \{i_1, \dots, i_t\}. \end{aligned}$$

2. By observing all access patterns $Access(Q_1), \dots, Access(Q_{t+1})$, \mathcal{A} knows the probabilities of the events that the current and previous queried records are in different locations, which, however, give \mathcal{A} nothing to identify which one in DB is or not the queried record $DB_{i_{t+1}}$.

Conclusion : For a clean query Q_{t+1} , we have

$$0 < Pr(Q_{t+1} \simeq DB_j | Access) \leq 1/M, \quad \forall j \in [n],$$

where $Access = \{Access(Q_1), Access(Q_2), \dots, Access(Q_{t+1})\}$.

The proof is concluded.

5 Performance

Let YDDB, WDDB, IS, AF and SS denote the schemes of Yang *et al* (Yang *et al.* 2008), Wang *et al* (Wang *et al.* 2006), Iliev & Smith (Iliev & Smith 2003, 2004, 2005), Asonov & Freytag (Asonov & Freytag 4/2002, 5/2002) and Smith & Safford (Smith & Safford 2000, 2001), respectively. Now we give a comparison of our protocol against these secure coprocessor-based PIR protocols as follows.

Table 1. Comparison of Performance

Scheme	OCC	AOPC	OPPC	OSIM
Ours	yes	4	$O(n \lg n)$	$4R+n(1+\lg n)$
YDDB	yes	$O(n/M)$	$O(n \lg n)$	$\sqrt{n}R+2P$
WDDB	yes	$O(n/M)$	$O(n \lg n)$	$\sqrt{2n}R+1P$
IS	yes	$O(n \lg n/M)$	$O(n \lg n)$	$\sqrt{2n}R+1P$
AF	yes	$O(n/M)$	$O(n^{1.5})$	$\sqrt{2n}R+1P$
SS	yes	$O(n)$	$O(n^2)$	$1R+1P$

- * OCC: optimal communication complexity
- * AOPC: average online processing complexity for a query
- * OPPC: offline preprocessing complexity
- * OSIM: optimal size of SC 's internal memory for minimizing query response time
- * n : the database size
- * M : the maximum of records kept in SC 's internal memory, $1 < M \ll n$
- * R : a record
- * P : a permutation function

According to the above table, except the parameter OSIM, the performance of our protocol is better than the performance of other existing coprocessor-based PIR protocols. Note that, in our protocol before DB replies to a query, just two records are read. Although online processing complexity is four I/O operations, but for a single query, the user observes a delay equivalent to two I/O operations. Only in the environment of burst queries, the user observes a slightly longer delay.

Now consider the parameter OSIM. It is easy to construct a bijective permutation from a range to another range, so the permutation functions in the schemes of Wang *et al* (Wang *et al.* 2006), Iliev & Smith (Iliev & Smith 2003, 2004, 2005), Asonov & Freytag (Asonov & Freytag 4/2002, 5/2002) and Smith & Safford (Smith & Safford 2000, 2001) need a small storage space. However, one of two permutations in Yang *et al*'s scheme (Yang *et al.* 2008) is a bijective function from a set of random numbers to another set of random numbers. It is hard to construct and needs a big storage space. In practice, it is usually replaced with a matching table. So, Yang *et al*'s scheme is less efficient than ours in the parameter OSIM. Comparing with Wang *et al*'s scheme (Wang *et al.* 2006), if $R \geq \frac{n(1+\lg n)-1P}{\sqrt{2n-4}}$, then $(\sqrt{2n}R+1P) \geq (4R+n(1+\lg n))$, this means, when the size of a record in the database DB is over $\frac{n(1+\lg n)-1P}{\sqrt{2n-4}}$ bits, our scheme is more efficient than Wang *et al*'s scheme considering the parameter OSIM. For example, assume that a database has one million records, *i.e.*, $n = 10^6$, and a permutation function has a size of 10^6 bits, then, when the record size is over 14175 bits, the internal storage capacity required in our scheme is smaller than that in Wang *et al*'s scheme.

To the best of our knowledge, except the inefficient schemes of Smith & Safford (Smith & Safford 2000, 2001), our protocol is the first one that does not need to reshuffle the database. Moreover, other SC-based PIR protocols have to make preprocessing periodically. Usually one cannot afford implementing inefficient PIR protocols, so something less secure but practical should be used. Comparing to the PIR protocols with uniform distribution, the proposed scheme is more efficient from implementation point of view.

6 Conclusions and Open Problem

In this paper we proposed a new secure coprocessor-based PIR protocol without reshuffling the database and showed its security. The protocol holds the optimal communication complexity, and its online processing complexity is almost optimal.

The *SC*-based PIR protocols assume that the *SC*'s secure memory can store a small number of data records. This assumption does not hold for multimedia databases, where records can be very long and storing even a single (and very long) record in *SC* can be a problem. Thus, task of designing a PIR protocol for databases with very long records is an interesting open problem.

Acknowledgements

Peishun Wang was supported by the RAACE scholarship, Macquarie University. Huaxiong Wang is supported in part by the Australian Research Council under ARC Discovery Project DP0665035 and the Singapore Ministry of Education under Research Grant T206B2204. Josef Pieprzyk was supported by the ARC grant DP0987734.

References

- Ambainis, A. (1997), Upper bound on the communication complexity of private information retrieval, *in* 'Proc. of the 24th ICALP'.
- Asonov, D., Freytag, J.-C. (4/2002), Almost optimal private information retrieval, *in* 'Proceedings of 2nd Workshop on Privacy Enhancing Technologies (PET2002)', San Francisco, USA.
- Asonov, D., Freytag, J.-C. (5/2002), Private information retrieval, optimal for users and secure coprocessors, Technical Report HUB-IB-159, Humboldt University Berlin.
- Beimel, A., Ishai, Y., Kushilevitz, E., Rayomnd, J.-F. (2002), Breaking the $O(n^{1/(2k-1)})$ barrier for information-theoretic private information retrieval, *in* 'Proc. of the 43st IEEE Sym. on Found. of Comp. Sci.'
- Chang, Y. (2004), Single Database Private Information Retrieval with Logarithmic Communication. *in* 'The 9th Australasian Conference on Information Security and Privacy (ACISP 2004)', LNCS, 3108, pp. 50-61, Springer-Verlag, Sydney, Australia.
- Chor, B., Kushilevitz, E., Goldreich, O., Sudan, M. (1998), Private information retrieval, *in* 'Journal of the ACM', 45. Earlier version in FOCS 95.
- Chor, B., Gilboa, N. (1997), Computationally private information retrieval, *In* 'Proceedings of 29th STOC'.
- Gentry, C., Ramzan, Z. (2005), Single-Database Private Information Retrieval with Constant Communication Rate, L. Caires et al. (Eds.): *in* 'ICALP 2005', LNCS 3580, pp.803-815.
- Iliev, A., Smith, S. (2003), Privacy-enhanced credential services, *in* 'PKI Research Workshop', volume 2, Gaithersburg, MD. NIST.
- Iliev, A., Smith, S. (2004), Private Information Storage with Logarithmic-space Secure Hardware, *in* 'Information Security Management, Education, and Privacy'. Kluwer, pp.201-216.
- Iliev, A., Smith, S. (2005), Protecting Client Privacy with Trusted Computing at the Server, *in* 'IEEE Security & privacy'.
- Sion, R., Carbunar, B. (2007), On the computational practicality of private information retrieval, *in* 'NDSS'.
- Smith, S. W., Palmer, E. R., Weingart, S. H. (1998), Using a high-performance, programmable secure coprocessor, *in* 'Proceedings of the 2nd International Conference on Financial Cryptography'.
- Smith, S. W., Safford, D. (2000), Practical private information retrieval with secure coprocessors. Technical report, IBM Research Division, T.J. Watson Research Center, 2000.
- Smith, S. W., Safford, D. (2001), Practical server privacy with secure coprocessors, *in* 'IBM Systems Journal', 40(3).
- Wang, S., Ding, X., Deng, R., and Bao, F. (2006), Private information retrieval using trusted hardware, *in* 'ESORICS', LNCS 4189, pp. 49-64.
- Woodruff, D., Yekhanin, S. A. (2005), Geometric Approach to Information-theoretic Private Information Retrieval, *in* 'CCC', pp.275-284.
- Yang, Y. Ding, X. Deng, R., Bao, F. (2008), An Efficient PIR Construction Using Trusted Hardware, *in* 'ISC', LNCS 5222, pp. 64-79.

Reconstruction of Falsified Computer Logs for Digital Forensics Investigations

Maolin Tang

Colin Fidge

Faculty of Science and Technology
Queensland University of Technology
Brisbane, Australia
Email: {m.tang, c.fidge}@qut.edu.au

Abstract

Digital forensics investigations aim to find evidence that helps confirm or disprove a hypothesis about an alleged computer-based crime. However, the ease with which computer-literate criminals can falsify computer event logs makes the prosecutor's job highly challenging. Given a log which is suspected to have been falsified or tampered with, a prosecutor is obliged to provide a convincing explanation for how the log may have been created. Here we focus on showing how a suspect computer event log can be transformed into a hypothesised actual sequence of events, consistent with independent, trusted sources of event orderings. We present two algorithms which allow the effort involved in falsifying logs to be quantified, as a function of the number of 'moves' required to transform the suspect log into the hypothesised one, thus allowing a prosecutor to assess the likelihood of a particular falsification scenario. The first algorithm always produces an optimal solution but, for reasons of efficiency, is suitable for short event logs only. To deal with the massive amount of data typically found in computer event logs, we also present a second heuristic algorithm which is considerably more efficient but may not always generate an optimal outcome.

Keywords: Digital forensics, computer logs, event correlation algorithms.

1 Introduction

Digital forensics involves investigations into suspected crimes or misbehaviors that are manifested in computer-based evidence (Richard III & Roussev 2006). Part of this process is showing that sequences of events hypothesised by a forensic investigator or legal prosecutor are consistent with the available digital evidence (Mohay 2005). If the integrity of the evidence, typically computer-generated logs, is in doubt, then the assumed actual sequence of events needs to be reconstructed, and its relationship to the digital artifacts needs to be explained in terms of actions that could reasonably have been performed by the defendant.

We wish to thank the anonymous reviewers for their many helpful comments. This work was supported in part by the Defence Signals Directorate and the Australian Research Council via ARC-Linkage Projects grant LP0776344.

Copyright ©2010, Australian Computer Society, Inc. This paper appeared at the Australasian Information Security Conference (AISC 2010), Brisbane, Australia. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 105, Colin Boyd and Willy Susilo, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

Digital forensics has three major phases: *acquisition*, *analysis* and *presentation* (Carrier 2002). In the acquisition phase the state of a digital system is preserved so that it can be analysed later. Tools used in the acquisition phase are programs that can copy data and software from a suspect storage device to a trusted one. In the analysis phase, the acquired data is examined in order to find pieces of evidence to support or contradict a given hypothesis intended to explain how the evidence was created. In the presentation phase the conclusions from the analysis phase are presented in a legal setting in a way comprehensible to non-experts. Here we are concerned with the analysis phase, especially the process of explaining how deliberately falsified computer log evidence may have been created.

Analyses of computer logs usually rely on timestamps to determine the order in which events occurred. However, the clocks used to generate timestamps may be inaccurate when compared to 'absolute' time (Schatz et al. 2006). Even worse, computer-literate criminals may attempt to cover their tracks by adjusting the clocks on their computers to create misleading event logs (Willassen 2008a).

To overcome this, we need to know whether or not an acquired event log may have been falsified and, if so, how this could have been done. To some extent this problem can be approached by comparing the suspect event log with independently-generated, trusted event logs. The difficulty, however, is that event logs produced by different sources, e.g., computer operating systems, door swipecard readers, network firewalls, network routers, Internet Service Providers, web servers, etc, contain records of different kinds of events. Determining the relationship between these events, and even finding a common representation for them, can be highly challenging (Chen et al. 2003).

Previous work has focussed on determining which trusted, causally-related event orderings are consistent with the timestamped evidence, according to a particular falsification hypothesis (Willassen 2008b). However, the problem of reconstructing hypothesised event sequences from the available digital evidence has received relatively little attention.

In this paper we consider the problem of determining how a suspect event log may have been created, in the context of a sequence of events hypothesised from independent, trusted sources of information. This outcome allows a prosecutor to assess the likelihood of different assumed actions by the defendant. A proposed scenario which involves relatively little effort on the part of a suspected criminal is more likely than one which requires a large number of actions in order to produce the presumably-falsified computer log.

We present two algorithms for quantifying the number of steps required to turn a prosecutor's hypothesised sequence of actual events into the sequence

of events found in a seized computer's log. The first algorithm always produces an optimal result, in the sense that it completes the transformation in the fewest steps. However, because computer forensics typically involves analysing massive amounts of data (Mohay 2005), we also present a heuristic algorithm which is considerably more efficient but may not yield an optimal answer. Nevertheless, for the purposes of arguing a legal case, a sub-optimal solution will often suffice.

2 Previous and Related Work

Much work has already been done on the problem of how to analyse and 'correlate' events found in different data logs, but in general this work differs from ours because it does not consider the possibility of reordering the events in a suspect log to match a particular hypothesis.

Broadly speaking, the legal importance of computer logs is illustrated by the emergence of standards for their maintenance. For instance, Kent & Souppaya (2006) present general guidelines for responsible management of computer security logs, including storage, data formats, data integrity, data confidentiality, etc.

Numerous tools have been proposed to help forensic investigators process the large amounts of data produced by computer-based systems. Case et al. (2008) note that existing forensics tools provide isolated data mining functions but leave the job of interpreting and correlating the data to the human investigator. They present a prototype environment for integrating and correlating timestamped events obtained from several different data logs into a single sequence. To do this they rely on the timestamps in login files and network traces to reconstruct the actions performed by a user during a particular session. They do not, however, consider the possibility that the ordering implied by the timestamps is not the actual one.

Similarly, Best et al. (2004) devised a tool for analysis of operating system logs to help security auditors mine the generated data. The tool searches for attempts to circumvent security mechanisms and changes in users' behaviour, as compared to previously-accumulated user profiles. More recently, Raghavan et al. (2009) proposed a software architecture for integrating forensic data from multiple sources that allows an investigator to explore theories about past behaviours, but again it does not consider the possibility of event reordering.

Since forensic evidence often relates to causal relationships between events, much previous work has focussed on detecting causal relationships across different logs. For instance, in the context of distributed systems, Gazagnaire & Hérouët (2007) considered the problem of knowing whether or not two events in different logs are causally related. Since logged data is typically incomplete with respect to event causality, they developed a theory for composition of partially-ordered event sets that do not include all causal relationships between events. Their 'event correlation' theory allows missing causal relationships to be reconstructed. However, they assumed that the causal relationships present in the given logs are correct, unlike our situation.

In a related vein, Schatz et al. (2004) focussed on the semantic content of event logs, so that domain-specific inferences can be drawn about possible hypotheses using data other than that which is obviously security-related. They used pattern-based reasoning rules to correlate logged events so that causal relationships between events can be identified. They

subsequently extended this work so that the reasoning rules could infer information from multiple heterogeneous domains, e.g., firewall logs and swipecard access logs (Schatz et al. 2005). The motivation for this work is similar to our own, but again they assume that the ordering of events in the individual logs being correlated is accurate, and make no allowance for deliberate falsification of event orderings.

In practice, the causal ordering of logged events is assumed to be determined by their associated timestamps, so many studies have focussed on the accuracy of timestamping mechanisms. For instance, Boyd & Forster (2004) noted that analysis of timestamped events can be complex due to different date-time data formats and time zones. They cite a criminal case in which the defence asserted that police had falsified evidence by adding files to the defendant's computer after it had been seized. Ultimately it transpired that this was not the case—the defence's expert witness had, in fact, misinterpreted the ordering of events by failing to include a necessary timezone-related offset to the timestamps.

Gladyshev & Patel (2005) presented a formalisation of the problem of placing bounds on the range of times within which a non-timestamped event could have occurred, by 'sandwiching' it between two causally-related timestamped events.

Willassen (2008b) treated the problem of inaccurate timestamps as the need to test a 'clock hypothesis' against the timestamped evidence. The aim of this work was to determine which causal action sequences are possible, given the hypothesised behaviour of the clock used to generate the timestamps. This work was then extended so that it could be used to detect 'antedating', i.e., deliberate falsification of timestamps, by detecting that the timestamps on events are inconsistent with (known) necessary causal orderings (Willassen 2008a). This work is highly relevant to our own, but its aim was to detect mismatches between actual and timestamped event sequences, whereas we are concerned with how to transform one to the other.

Similarly, Schatz et al. (2006) considered the problem of correlating timestamped events when the clocks used to generate the timestamps differ from absolute time due to inadvertent clock skew or drift, or deliberate clock tampering. They present the results of empirical experiments for measuring the offset of the timestamping clock from a trusted reference clock in order to determine the likely offset in the timestamps. Again, however, the way in which actual and timestamped event orderings can be linked was not the primary focus of their work.

Finally, to a certain extent, the algorithms we develop below are related to the classic 'marriage problem', in which the most efficient pairing of elements from two disjoint sets must be found. This well-known combinatorial challenge is usually solved via a backtracking algorithm (Berman & Paul 2005). It differs from our problem, however, because there is no concept of an overall ordering in the two sets of elements.

3 Problem Statement

Consider the problem faced by a prosecutor intent on showing that acquired evidence is consistent with a particular hypothesis concerning criminal behaviour, even in the face of deliberate falsification of the evidence. In the case of digital data this usually means attempting to reconcile a hypothesis formed from trusted information, such as the logs generated by an Internet Service Provider, with non-trustworthy information, such as the file and event timestamps on

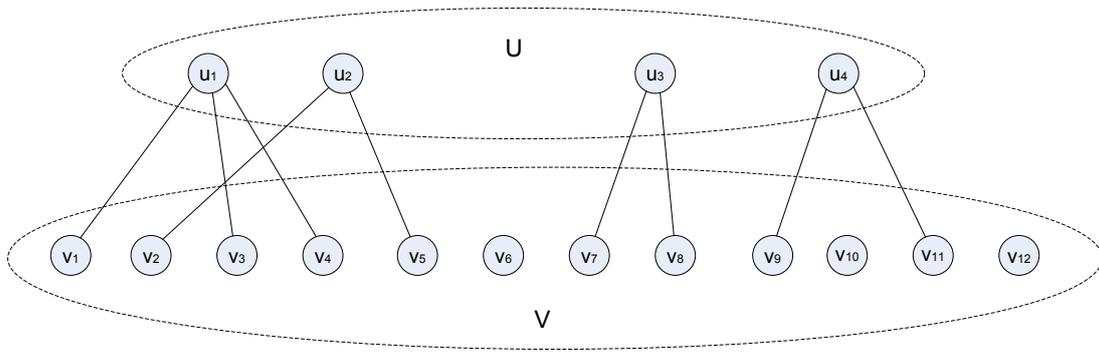


Figure 1: Example computer log and ISP log correlation model

the personal computer owned by the accused (Schatz et al. 2006).

For example, the correlation between the events in a computer log and the events in a corresponding ISP log can be modelled as a bigraph $G = (U \cup V, E)$, where V is the *computer event set*, containing all the events in the potentially-compromised computer's logs, and U is the *ISP event set*, containing all relevant events in the trustworthy ISP's event logs. An edge (u, v) is in E if event $v \in V$ is correlated to event $u \in U$, or if event $v \in U$ is correlated to event $u \in V$.

Each event in the ISP's event log may be related to multiple events in the computer's activity log. For example, an event in the ISP log may be a visit to a particular website from the defendant's personal computer. While visiting the website, however, multiple individual files may have been downloaded to the computer. Thus, in the computer log there will be multiple download events, while the ISP log records only a single website visit.

Figure 1 illustrates such a computer log and ISP log correlation model. In this case there are four events recorded in the ISP's log, u_1, u_2, u_3 and u_4 , but there are twelve events in the personal computer's log, v_1, v_2, \dots, v_{12} . Events v_1, v_3 and v_4 recorded in the computer's log are correlated to event u_1 in the ISP's log; events v_2 and v_5 in the computer's log are related to event u_2 in the ISP's log; events v_7 and v_8 in the computer's log are related to event u_3 in the ISP's log; and events v_9 and v_{11} in the computer's log are related to event u_4 in the ISP's log. Computer log events v_6, v_{10} and v_{12} are generated by local actions that did not involve Internet access, so they have no correspondence to any ISP events.

We assume the events in the ISP's log were all timestamped, in a way accurately reflecting the order in which Internet-related events actually occurred. This ordering can be modeled by a directed graph $G_U = (V_U, E_U)$, where $V_U = U$ is the set of ISP events, and $(u_i, u_j) \in E_U$ if u_i and u_j were two consecutive events in the ISP's log. We call directed graph G_U the *ISP event constraint graph*. For instance, assume the events in the ISP's log in Figure 1 have timestamps corresponding to the sequence u_3, u_2, u_1 and u_4 . Then the ISP event constraint graph is shown in Figure 2.



Figure 2: Example ISP event constraint graph

An event in the ISP's event log may be correlated to multiple events in the corresponding computer event log and there may be reasonableness constraints between the events. For example, a computer file must be created before it can be accessed or modified. Thus, a file creation event must occur before an access or modification event for the same file. Such constraints can be represented by a *computer event constraint graph* $G_V = (V_V, E_V)$, where $V_V = V$ is the computer event set, and $(v_i, v_j) \in E_V$ if event v_i must occur before event v_j . Figure 3 is an instance of the computer event constraint graph for the set of computer events shown in Figure 1.

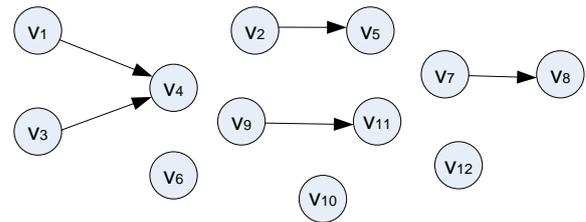


Figure 3: Example computer event constraint graph

All of the information in Figures 1, 2 and 3 can be used by a prosecutor to form a hypothesis about the actual sequence of events that must have occurred in some criminal case. If the log of events recorded on the defendant's computer is inconsistent with these independent sources of information, we are obliged to find an explanation for how the presumably-falsified log could have been created. This is the motivation for our research.

It is assumed that all the events in the computer event log were timestamped. We define a computer event log as a sequence of recorded events ordered by their timestamps. Such a log is considered to be 'falsified' if the timestamped event sequence violates any of the event orderings implied by independent information sources such as those in Figures 1, 2 and 3. For example, the timestamped event sequence $\langle v_1, v_9, v_2, v_4, v_3, v_5, v_6, v_7, v_8, v_{10}, v_{11}, v_{12} \rangle$ is falsified because it violates the constraint that event v_3 must occur before event v_4 in Figure 3, and the requirement that v_5 comes before v_9 as required by Figures 1 and 2.

When someone attempts to disguise their actions by falsifying a computer log, this is often done by resetting the clock on their computer to alter the apparent order in which certain actions were actually performed, and by then setting the clock back again to hide the fact that any such subterfuge has been

attempted (Willassen 2008a). This has the effect of causing a *consecutive sequence* of events to be given misleading timestamps. If the timestamps on events are used to infer the assumed event ordering, the outcome is that the whole sequence of events performed while the computer’s clock was maladjusted will appear in the wrong place in the overall event history. In effect, this mechanism can be used by malefactors to “move” sequences of events in the log.

Given the suspicion that such a deception has taken place, we assume that a forensic investigator (or criminal prosecutor) has a hypothesis in mind about the actual sequence of events, typically informed by independent sources of information such as ISP logs. Our technical goal, therefore, given a purportedly-falsified computer event log, and the forensic investigator’s hypothesised computer event log, is to assess the reasonableness of the forensic investigator’s hypothesis by showing how the hypothesised log can be transformed into the falsified one in the smallest number of moves.

4 Our Approach

This section presents two algorithms for the falsified computer event log reconstruction problem, an optimal algorithm for analysing relatively small data sets, and a heuristic algorithm for large data sets. The latter is necessary in practice because the problem of reconstructing event logs is a combinatorial optimisation problem and the search space of the optimal algorithm has a super-linear order of growth.

4.1 Definitions

Before presenting the algorithms themselves, we introduce some definitions used in the rest of the paper.

A *consecutive event sequence* is a sequence of events in a falsified computer event log that occur consecutively in the corresponding hypothesised computer event log. A *maximal consecutive event sequence* is a consecutive event sequence that is not covered by any other consecutive event sequence. For example, let $S = v_4v_5v_1v_2v_3$ be a falsified computer event log and $S^* = v_1v_2v_3v_4v_5$ be the hypothesised computer event log. Then, $\langle v_1 \rangle$, $\langle v_1v_2 \rangle$ and $\langle v_1v_2v_3 \rangle$ are three consecutive event sequences in S . But only the consecutive sequence $\langle v_1v_2v_3 \rangle$ is a maximal consecutive computer event sequence. There are two maximal consecutive event sequences in S , $\langle v_1v_2v_3 \rangle$ and $\langle v_4v_5 \rangle$.

In order to facilitate the presentation, we assume that the computer events in a hypothesised computer event log are in sequential order according to their subscripts, without loss of generality. Thus, a maximal consecutive computer event sequence in S must be of the form $\langle v_i v_{i+1} \dots v_j \rangle$ and is denoted as v_{i-j} in the rest of this paper, where $i \leq j$.

A falsified computer event log can be represented by a sequence of maximal consecutive event sequences. For example, $S = v_4v_5v_1v_2v_3$ can be represented as $v_{4-5}v_{1-3}$.

4.2 Finding Maximal Consecutive Event Sequences

To start our analysis we first partition the falsified event sequence S into a sequence of maximal consecutive event sequences, so that these entire sequences can be ‘moved’ as a single unit. This is necessary since a single adjustment to a computer’s clock will effectively shift an entire sequence of subsequent events in the log. However, we do not want to consider each of

Algorithm 1 Transform a computer event sequence into a sequence of maximal consecutive event sequences

Require: A computer event sequence $S = v_1v_2 \dots v_n$, and its corresponding hypothesised computer event sequence $S^* = v_1^*v_2^* \dots v_n^*$;

Ensure: The maximal consecutive event sequence representation of S .

$i = 1$;

$seqs = \emptyset$;

while $i \leq n$ **do**

 Find the maximal consecutive event sequence in S starting from the i^{th} computer event using Algorithm 2;

$i = i +$ the length of the maximal consecutive event sequence;

$seqs = seqs +$ the maximal consecutive event sequence;

end while;

return $seqs$.

the events individually, as this would give a misleading impression about the ‘effort’ required to falsify the log.

Algorithm 1 transforms a computer event sequence into a sequence of maximal consecutive event sequences. Firstly, it finds the maximal consecutive computer event sequence in S starting from S ’s first event, then finds the next maximal consecutive computer event sequence starting from the end of the maximal sequence, and so on. The process of finding the next maximal consecutive computer event sequence is repeated until all maximal consecutive computer event sequences are found. The maximal consecutive computer event sequences are concatenated to form a sequence of maximal consecutive computer event sequences.

Algorithm 2 finds a maximal consecutive computer event sequence in S , starting from a particular event v_i , and is used as an auxiliary subroutine by Algorithm 1. It first searches through S^* for the starting event v_i , and then accumulates events as long as those in S and S^* match.

Algorithm 2 Find a maximal consecutive event sequence

Require: A computer event sequence $S = v_1v_2 \dots v_n$, its corresponding hypothesised computer event sequence $S^* = v_1^*v_2^* \dots v_n^*$, and a starting location i ;

Ensure: A maximal consecutive event sequence starting from location i .

$j = 1$;

while $v_i \neq v_j^*$ **do**

$j = j + 1$;

end while;

$start_location = j$;

repeat

$i = i + 1$;

$j = j + 1$;

until $v_i \neq v_j^*$;

$end_location = j - 1$;

return $v_{start_location-end_location}$.

It is straightforward to show that the computational complexity of Algorithm 1 is $O(n \times m)$, where n is the number of computer events in S and m is the number of maximal consecutive computer event sequences in S . Algorithm 2 takes $O(n)$ time to find the location of event v_i in S^* and $O(n)$ time to find the maximal computer event sequence starting from this event, so the computational complexity of Algo-

gorithm 2 is $O(n)$. Assume that S contains m maximal consecutive computer event sequences. Then Algorithm 1 invokes Algorithm 2 m times. Thus, the computational complexity of Algorithm 1 is $O(n \times m)$.

4.3 An Optimal Algorithm for Determining the Moves Required to Falsify a Log

This section presents an algorithm guaranteed to find the optimal solution to the problem of how a computer log could have been falsified, where optimality is defined as transforming the hypothesised log into the presumably-falsified one in the minimal number of moves. Each ‘move’ involves shifting an entire sequence of events and, in practice, is the consequence of resetting the computer’s clock to disguise the true sequence of events.

The algorithm is basically an A^* algorithm (Hart et al. 1968). An A^* algorithm is a best-first graph searching algorithm that finds the shortest path from a given source node s to a goal node t in the graph. It uses a heuristic function, $f(x)$, to determine the order in which to visit nodes in the graph. The heuristic function is a sum of two subfunctions: a distance function, $g(x)$, which is the distance from the source node s to current node x , and an admissible “heuristic estimate” $h(x)$ of the distance from node x to goal node t . Function $h(x)$ must be an admissible heuristic, that is, it must not overestimate the distance from x to t in order to guarantee the admissibility and optimality of the A^* algorithm (Hart et al. 1968).

The evaluation function used by our A^* algorithm is defined by Equation 1 below.

$$f(x) = g(x) + h(x) \quad (1)$$

Here $g(x)$ is the actual distance (the number of moves) from s to x , and $h(x)$ is defined by Equation 2.

$$h(x) = \lceil (m - 1)/3 \rceil \quad (2)$$

Here m stands for the number of maximal consecutive event sequences in x and $\lceil m/3 \rceil$ is the smallest integer greater than or equal to $m/3$. Function $h(x)$ is an admissible heuristic estimate of the distance, i.e., the number of moves, from x to t because the number of moves from x to t is greater than or equal to $\lceil m/3 \rceil - 1$ (see Section 4.4).

Algorithm 3 is our A^* search algorithm and Algorithm 4 is an auxiliary algorithm for backtracking and generating output.

In the A^* algorithm, four data structures are used. The first is *OpenSet*, which is a set that stores computer event logs that are at the frontier of the A^* search. The second is *ClosedSet*, which keeps computer event logs that have been visited. The third is *came_from*[x], which is used to store the parent computer event log sequence of x . For example, if *came_from*[x] = y , then it indicates that y was obtained by moving a maximal computer event sequence in x . The fourth is *moves*[y], which is used to store how y was obtained. Specifically, *moves*[y] contains a maximal computer sequence and the location of the maximal computer sequence in its parent. For example, *moves*[y] = [v_{i-j} , *from*, *to*] indicates that the parent computer event log of y can be built by moving the maximal consecutive event sequence v_{i-j} from location *from* to location *to* in y . Note that locations *from* and *to* are those in the original computer event log, rather than the locations in the maximal consecutive event sequence.

Consider a simple example to illustrate how the A^* search algorithm works. Let the hypothesised event sequence be $S^* = v_1v_2v_3v_4v_5v_6v_7v_8$ and the falsified one be $S = v_3v_4v_5v_7v_8v_6v_1v_2$. First of all,

Algorithm 3 A^* search

Require: A falsified sequence of computer events S , and a hypothesised computer event log S^* ;

Ensure: A sequence of moves for transforming S^* into S .

Transform S into a sequence of maximal consecutive computer event sequences using Algorithm 1; m = the number of maximal consecutive events in S ;

for $i = 1, m$ **do**

came_from[i] = *null*;

moves[i] = *null*;

end for;

ClosedSet = \emptyset ;

OpenSet = $\{S\}$;

$g(S) = 0$;

$h(S) = \lceil (m - 1)/3 \rceil$, where m is the number of maximal consecutive event sequences in S ;

$f(S) = g(S) + h(S)$;

while *OpenSet* $\neq \emptyset$ **do**

x = the element in *OpenSet* that has the least $f(x) = g(x) + h(x)$ and contains the minimal number of maximal consecutive event sequences (if there is more than one element that satisfies the conditions, then the one that was added to *OpenSet* first is selected);

Remove x from *OpenSet*;

Add x to *ClosedSet*;

m = the number of maximal consecutive computer event sequences in x ;

if $m = 1$ **then**

Use Algorithm 4 to backtrack and output the sequence of moves from S^* to S , and then stop.

end if;

for each y that can be obtained by moving a maximal consecutive computer event sequence from location i to location j , where $1 \leq i, j \leq m$ **do**

if $y \notin \textit{ClosedSet}$ **then**

new_g(y) = $g(x) + 1$;

came_from[y] = x ;

moves[y] = [x, i, j];

if $y \notin \textit{OpenSet}$ **then**

Add y to *OpenSet*;

$g(y) = \textit{new_g}(y)$;

$h(y) = \lceil (m - 1)/3 \rceil$, where m is the number of maximal consecutive event sequences in y ;

$f(y) = g(y) + h(y)$;

else

if *new_g*(y) < $g(y)$ **then**

$g(y) = \textit{new_g}(y)$;

end if

end if

end if

end for

end while

Algorithm 4 Backtracking moves

Require: A hypothesis computer event sequence S^* , a falsified computer event sequence S , and values *came_from* and *moves* generated by the search algorithms;

Ensure: A sequence of moves that transform a hypothesised computer event sequence S^* to a falsified computer event sequence S .

$x = S^*$;

$y = \textit{came_from}[x]$;

while $y \neq \textit{null}$ **do**

print *moves*[x];

$x = y$;

$y = \textit{came_from}[x]$;

end while

the A^* search algorithm transforms S into a sequence of maximal consecutive computer event sequences $v_{3-5}v_{7-8}v_{6-6}v_{1-2}$. The number of maximal consecutive computer event sequences is $m = 4$.

After initializing variables $came_from[i]$, $moves[i]$, $ClosedSet$ and $OpenSet$, the A^* search algorithm explores all the neighbours of S . A neighbour of S is defined as a computer event log that can be obtained by moving a maximal consecutive computer event sequence from one location to another in S . After initialisation, $OpenSet = \{S\}$ and $ClosedSet = \emptyset$, and $f(S)$ is calculated.

In the first exploration, the A^* search algorithm selects S to explore as it is the only element in $OpenSet$ and moves it from $OpenSet$ to $ClosedSet$. Then, all neighbours of S are added to $OpenSet$ as none of them is in $OpenSet$ or $ClosedSet$, all the f values of the neighbours are updated, and all the $came_from$ and $moves$ elements are updated as well. The search space after this exploration is shown in Figure 4.

After the first exploration, $ClosedSet$ equals $\{S\}$ and $OpenSet$ is $\{v_{7-8}v_{3-6}v_{1-2}, v_{7-8}v_{6-6}v_{3-5}v_{1-2}, v_{7-8}v_{6-6}v_{1-5}, v_{3-8}v_{1-2}, v_{3-6}v_{1-2}v_{7-8}, v_{6-6}v_{3-5}v_{7-8}v_{1-2}, v_{3-5}v_{7-8}v_{1-2}v_{6-6}, v_{1-5}v_{7-8}v_{6-6}, v_{3-5}v_{1-2}v_{7-8}v_{6-6}\}$.

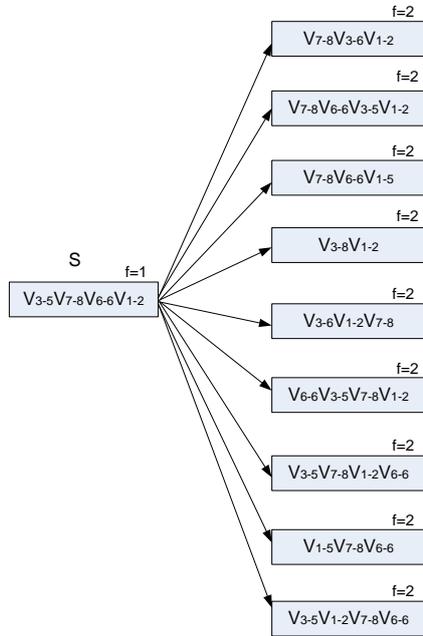


Figure 4: The A^* search space after exploring S

In the second exploration, the A^* search algorithm selects $v_{3-8}v_{1-2}$ to explore as it has the minimal f value and contains a minimal number of maximal consecutive event sequences. The algorithm moves $v_{3-8}v_{1-2}$ from $OpenSet$ to $ClosedSet$. Then, $v_{1-8} = S^*$ is added to $OpenSet$ as v_{1-8} is the only neighbour of $v_{3-8}v_{1-2}$ that is not in $OpenSet$ or $ClosedSet$, the f value of v_{1-8} is calculated, and the $came_from$ and $moves$ elements are updated as well. The search space after this exploration is shown in Figure 5.

After the second exploration, $ClosedSet$ equals $\{S, v_{3-8}v_{1-2}\}$ and $OpenSet$ equals $\{v_{7-8}v_{3-6}v_{1-2}, v_{7-8}v_{6-6}v_{3-5}v_{1-2}, v_{7-8}v_{6-6}v_{1-5}, v_{3-6}v_{1-2}v_{7-8}, v_{6-6}v_{3-5}v_{7-8}v_{1-2}, v_{3-5}v_{7-8}v_{1-2}v_{6-6}, v_{1-5}v_{7-8}v_{6-6}, v_{3-5}v_{1-2}v_{7-8}v_{6-6}, v_{1-8}\}$.

In the third exploration, the A^* search algorithm selects sequence v_{1-8} to visit. Since $m = 1$, the A^* search algorithm has found the goal. After the goal computer event log is found, the backtracking algorithm is used to retrieve the path from S^*

to S . This path is then the optimal sequence of moves, $[v_{3-8}, 3, 1]$ followed by $[v_{7-8}, 5, 4]$. In other words, event sequence $v_1v_2v_3v_4v_5v_6v_7v_8$ can be transformed into event sequence $v_3v_4v_5v_7v_8v_6v_1v_2$ in only two ‘moves’, highlighting the fact that this seemingly complex rearrangement of events does not take a major effort. In a criminal case this observation may be crucial in making the prosecutor’s case convincing.

4.4 A Heuristic Algorithm for Determining the Moves Required to Falsify a Log

The A^* algorithm can always find an optimal solution, i.e., a minimal sequence of moves that transform a hypothesised computer event log into a falsified one. Unfortunately, A^* algorithms are notoriously expensive. Their time complexity ranges from polynomial to exponential, depending on the heuristic function used, and their space complexity is often exponential. Given the large number of events in actual computer logs, the algorithm in Section 4.3 will often prove impractical.

Therefore, this section presents a heuristic algorithm that is more efficient. Although it will always find a solution, it cannot be guaranteed to find the optimal one. Nevertheless, by making use of some contextual information, the heuristic algorithm will usually find a solution involving a small number of moves. Furthermore, in the context of a legal argument about the likelihood of a crime having been committed, finding a mathematically optimal solution may not be necessary.

Like the A^* algorithm, it is assumed that a falsified computer event log was created by repeatedly moving one consecutive computer event sequence at a time from one location to another. In order to minimise the number of moves, we assume that whenever a consecutive computer event sequence is moved, it has to be a maximal consecutive computer event sequence. The algorithm’s input again includes a falsified computer event log S and a hypothesised computer event log S^* . The output of the algorithm is a sequence of moves of maximal consecutive computer event sequences that transforms S^* into S .

The heuristic algorithm treats the log reconstruction problem as one of finding a shortest path in a state space. Each state in the state space represents a computer event sequence that can be transformed from S by moving some maximal consecutive computer event sequences in S in a particular order. Once the path is found, the heuristic algorithm backtracks along the shortest path, which represents the sequence of moves of maximal consecutive computer event sequences, using the same backtracking procedure used by the A^* algorithm.

Algorithm 5 is our heuristic algorithm for the falsified computer event log reconstruction problem. The search process is iterative. In each iteration the algorithm explores all the new computer event sequences that can be obtained by moving one of the maximal consecutive event sequences in S and chooses the new computer event sequence that contains the minimal number of maximal consecutive event sequences. By so doing it minimizes the number of moves from the falsified computer event sequence to the hypothesis computer event sequence.

There are two operations that are used frequently in the heuristic algorithm. One is to update the maximal consecutive event sequence in a temporary falsified computer event sequence S_1 and the other is to update the number of maximal consecutive computer event sequences in S_1 . To do this we merely need to perform two major steps.

Assume that a maximal consecutive computer event sequence is being moved from location i to lo-

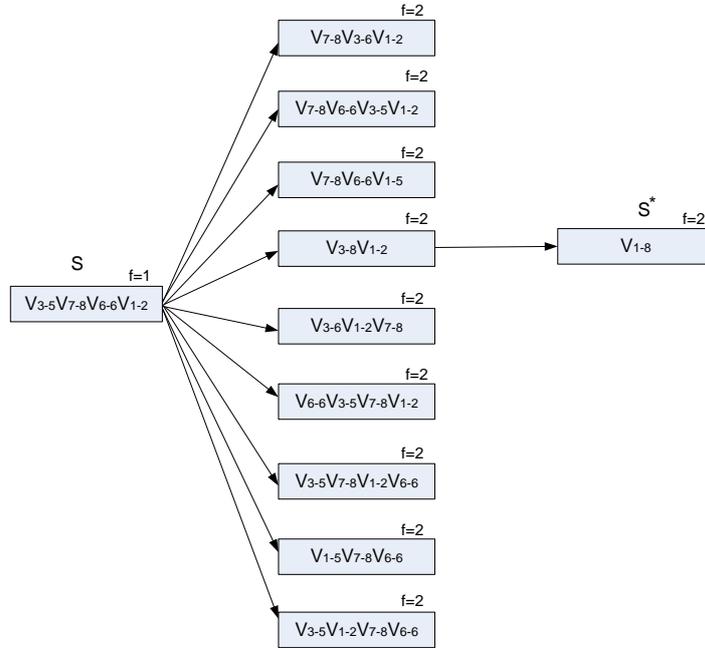


Figure 5: The A^* search space after exploring $v_{3-8}v_{1-2}$

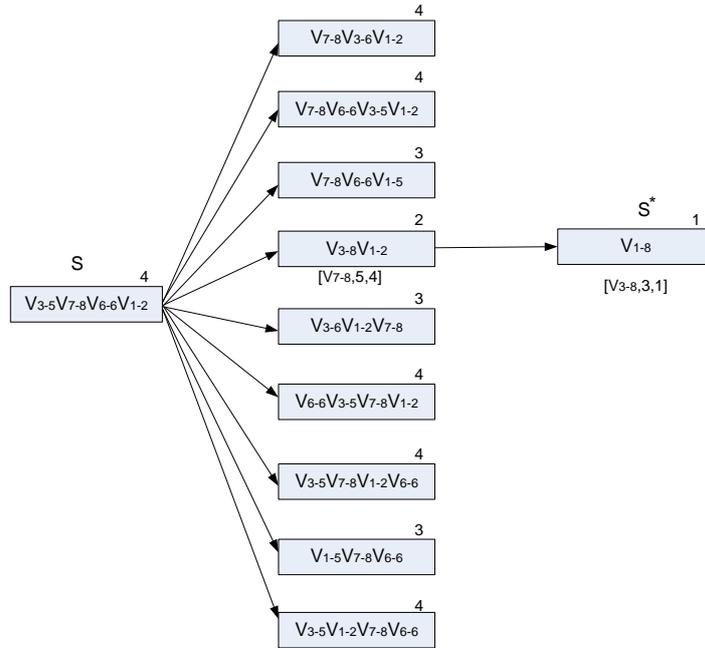


Figure 6: The search space of the heuristic algorithm

cation j in S_1 . We check if the maximal consecutive computer event sequences ending at locations $i - 1$ and $i + 1$ can be merged to form a new maximal consecutive computer event sequence, where $i + 1 \leq m_1$ and m_1 is the number of maximal consecutive computer event sequences in S_1 . This handles the situation where moving a maximal event sequence allows the sequences that surrounded it in its old location to be conjoined to create a new maximal consecutive sequence. If so, then they are merged to form a new maximal sequence, and we decrease the number of maximal consecutive computer event sequences by one.

Similarly, we further check whether or not moving a sequence to a new location allows it to be conjoined

at one or both ends with its new neighbours to create an even longer maximal sequence. In other words, we want to know if the newly-moved sequence can be merged into a single maximal consecutive computer event sequences at both new locations $j - 1$ and $j + 1$, where $j < m_1$ and m_1 is the number of maximal consecutive computer event sequences in S_1 . If so, we merge them and reduce the number of maximal consecutive computer event sequences by two. However, if the moved maximal sequence can be conjoined with its new neighbours at just one end, i.e., at location $j - 1$ or $j + 1$, we perform the merge and reduce the number of maximal consecutive computer event sequences by one.

Figure 7 shows an example of sequence concatena-

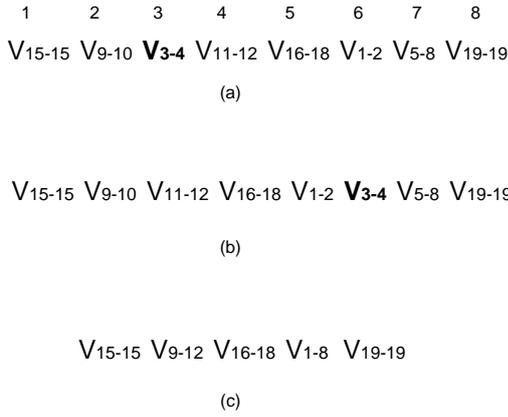


Figure 7: An example of sequence concatenation when moving a sequence

tion. In the example, maximal consecutive sequence v_{3-4} is being moved from location 3 to location 7 in $S_1 = v_{15-15}v_{9-10}v_{3-4}v_{11-12}v_{16-18}v_{1-2}v_{5-8}v_{19-19}$. Figure 7(a) is the state of S_1 before the move; Figure 7(b) shows the intermediate state of S_1 after v_{3-4} has been moved from location 3 to location 7 but before sequence concatenation is performed; Figure 7(c) displays the state of S_1 after sequence concatenation is done. Once sequence v_{3-4} is moved from between them, maximal consecutive sequences v_{9-10} and v_{11-12} become adjacent in S_1 and are merged to form a longer maximal consecutive sequence v_{9-12} . Also when maximal consecutive sequence v_{3-4} is placed between v_{1-2} and v_{5-8} , all three are merged to form a new maximal consecutive sequence v_{1-8} . The total number of maximal consecutive sequences in S_1 is thus reduced by three in this case.

Theorem 1 *Algorithm 5 can always find a feasible solution that moves maximal consecutive event sequences between $\lceil(m-1)/3\rceil$ and $m-1$ times, where m is the number of maximal consecutive event sequences in a falsified computer event sequence.*

Proof. Assume that a maximal consecutive computer event sequence is being moved from location i to location j in S_1 . In the best case, when the maximal consecutive computer event sequences at the original locations $i-1$ and $i+1$ can be merged to form a new maximal consecutive computer event sequence and the maximal consecutive computer event sequence at new location $j-1$, the newly-moved sequence, and the event sequence at new location $j+1$ can all be merged to form a new maximal consecutive computer event sequence, the number of maximal consecutive computer event sequences in S will be reduced by a total of three. Since there are m maximal consecutive event sequences in S initially, it may thus take as few as $\lceil(m-1)/3\rceil$ moves to reduce the number of maximal consecutive event to one.

In the worst case, when the maximal consecutive computer event sequence being moved can be merged only with either the maximal consecutive computer event at location $j-1$ or the one at location $j+1$, the number of maximal consecutive computer event sequences will be reduced by one only. Since there are m maximal consecutive event sequences in S initially, it may thus take at most $m-1$ moves to reduce the number of maximal consecutive sequences to one.

In the following we use a simple example to illustrate how the heuristic algorithm works. The process is shown in Figure 6. In this example, $S^* =$

Algorithm 5 A heuristic algorithm for the falsified computer event log reconstruction problem

Require: A falsified sequence of computer events S , and a hypothesised sequence of computer events S^* ;

Ensure: A sequence of moves that transforms S^* into S .

Transform S into a sequence of maximal consecutive event sequences using Algorithm 1;

m = the number of maximal consecutive events in S ;

for $i = 1, m$ **do**
 $came_from[i] = null$;
 $moves[i] = null$;

end for;

while $m > 1$ **do**

$S_2 = S$;

$m_2 = m$;

for $i = 1, m$ **do**

for $j = 1, m$ **do**

$S_1 = S$;

 Move the i^{th} maximal consecutive event sequence in S_1 from location i to location j ;

 Update the maximal consecutive computer event sequences in S_1 and the number of maximal consecutive event sequences in S_1 ,

m_1 ;

if $m_1 < m_2$ **then**

$S_2 = S_1$;

$m_2 = m_1$;

$from = i$;

$to = j$;

end if

end for

end for

$came_from[S_2] = S$;

$moves[S_2] = [S, from, to]$;

$S = S_2$;

$m = m_2$;

end while;

Backtrack the moves from S^* to S using Algorithm 4.

$v_1v_2v_3v_4v_5v_6v_7v_8$ and $S = v_3v_4v_5v_7v_8v_6v_1v_2$. First of all, the heuristic algorithm transforms S into a sequence of maximal consecutive computer event sequences $v_{3-5}v_{7-8}v_{6-6}v_{1-2}$. The number of maximal consecutive computer event sequences $m = 4$.

The heuristic algorithm explores all the sequences that can be obtained by moving one of the maximal consecutive sequences in S from one location to another, to see which of them produces the minimal number of maximal consecutive computer event sequences. Since $v_{3-8}v_{1-2}$ is the one that has minimal number of maximal consecutive computer event sequences, in this case two, the search moves from $S = v_{3-5}v_{7-8}v_{6-6}v_{1-2}$ to $v_{3-8}v_{1-2}$.

The heuristic algorithm then similarly explores all the state-space neighbours of $v_{3-8}v_{1-2}$. In this case $v_{3-8}v_{1-2}$ has only one neighbour v_{1-8} and the number of maximal consecutive computer event sequences in v_{1-8} is one, so the search process terminates. Then the backtracking algorithm backtracks along the path from S^* to S . As a result, the sequence of moves $[v_{3-8}, 3, 1]$ and $[v_{7-8}, 5, 4]$ is obtained.

In this particular case the heuristic algorithm finds the same two-move solution as the optimal one. Although this may not always be the case in general, the heuristic algorithm will always find a solution, and this will typically be one involving a small number of steps thanks to the process of conjoining maximal sequences into even longer ones whenever possible during an iteration of the algorithm. More importantly,

the heuristic algorithm can be applied to larger data sets than the optimal one, in general.

This is because the A^* algorithm is a global optimal search algorithm. In the best case it may find a globally-optimal solution by exploring only a small part of the potential search space, but in the worst case it will be forced to explore all the alternatives in which an optimal solution may exist. By contrast, the heuristic algorithm is a local search algorithm. It explores only one local optimum area and converges to its local optimum (which may or may not be a global optimum). Therefore, even though it is possible for the A^* algorithm to outperform the heuristic algorithm in particular cases, i.e., when the A^* algorithm is lucky enough to find a globally-optimal solution in a small search area early, and the heuristic algorithm is forced to explore a large locally-optimal search space, this will not be true in general. For large data sets with multiple local optimums, the A^* algorithm's average performance will be significantly worse than that of the heuristic algorithm.

5 Conclusion

Analysing computer forensic evidence is highly challenging. We have presented two algorithms for determining how a falsified computer log can be related to a hypothesised sequence of events, in terms of the effort required to transform one into the other. This gives us a sound basis for arguing about the likelihood of someone deliberately disguising their actions by tampering with computer logs, e.g., by adjusting the clock on their personal computer.

In future work we will perform empirical studies showing how the approach works in practice on actual large-scale computer logs. In addition, we will investigate how to construct the hypothesised computer event log using data from multiple trusted sources. In real life there may be many ways to obtain event logs, from timestamps on computer files, to logs on web servers, to logs of swipe card accesses on doors. A major unresolved challenge is to match an assumed “scenario” with all of these logs simultaneously.

References

- Berman, K. A. & Paul, J. L. (2005), *Algorithms: Sequential, Parallel and Distributed*, Thomson. ISBN 0-534-42057-5.
- Best, P., Mohay, G. & Anderson, A. (2004), ‘Machine-independent audit trail analysis—a decision support tool for continuous audit assurance’, *Intelligent Systems in Accounting, Finance and Management* **12**(2), 85–102.
- Boyd, C. & Forster, P. (2004), ‘Time and date issues in forensic computing: A case study’, *Digital Investigation* **1**, 18–23.
- Carrier, B. (2002), Open source digital forensics tools: The legal argument, Technical report, @stake. www.atstake.com/research/reports/acrobat/atstake_opensource_forensics.pdf.
- Case, A., Cristina, A., Marziale, L., Richard, G. & Roussev, V. (2008), ‘FACE: Automated digital evidence discovery and correlation’, *Digital Investigation* **5**, 65–75.
- Chen, K., Clark, A., De Vel, O. & Mohay, G. (2003), ECF—event correlation for forensics, in ‘Proceedings of the First Australian Computer, Network and Information Forensics Conference, Perth, 25 November’.
- Gazagnaire, T. & Hérouët, L. (2007), Event correlation with boxed pomsets, in ‘Proceedings of the 27th IFIP WG 6.1 international Conference on Formal Techniques For Networked and Distributed Systems (FORTE 2007)’, Vol. 4574 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 160–17.
- Gladyshov, P. & Patel, A. (2005), ‘Formalising event time bounding in digital investigations’, *International Journal of Digital Evidence* **4**(2).
- Hart, P., Nilsson, N. & Raphael, B. (1968), ‘A formal basis for the heuristic determination of minimum cost paths’, *IEEE Transactions on Systems Science and Cybernetics* **4**(2), 100–107.
- Kent, K. & Souppaya, M. (2006), Guide to computer security log management, Technical report, U.S. Department of Commerce, National Institute of Standards and Technology. Special Publication 800-92.
- Mohay, G. (2005), Technical challenges and directions for digital forensics, in ‘Proceedings of the First International Workshop on Systematic Approaches to Digital Forensic Engineering’, IEEE Computer Society Press, pp. 155–161.
- Raghavan, S., Clark, A. & Mohay, G. (2009), FIA: An open forensic integration architecture for composing digital evidence, in ‘Proceedings of the Second International Conference on Forensics in Telecommunications, Information and Multimedia, Adelaide, January 2009’, pp. 83–94.
- Richard III, G. G. & Roussev, V. (2006), ‘Next-generation digital forensics’, *Communications of the ACM* **48**(2), 76–80.
- Schatz, B., Mohay, G. & Clark, A. (2004), Rich event representation for computer forensics, in ‘Proceedings of the Asia Pacific Industrial Engineering and Management Systems Conference (APIEMS 2004), Gold Coast, Queensland, 12–15 December’.
- Schatz, B., Mohay, G. & Clark, A. (2005), ‘Generalising event forensics across multiple domains’, *The Journal of Information Warfare* **4**(1), 69–79.
- Schatz, B., Mohay, G. & Clark, A. (2006), ‘A correlation method for establishing provenance of timestamps in digital evidence’, *Digital Investigation* **3**, 98–107.
- Willassen, S. Y. (2008a), Finding evidence of antedating in digital investigations, in ‘Proceedings of the Third International Conference on Availability, Reliability and Security (ARES 2008), 4–7 March’, pp. 26–32.
- Willassen, S. Y. (2008b), Timestamp evidence correlation by model based clock hypothesis testing, in ‘Proceedings of the First International Conference on Forensic Applications and Techniques in Telecommunications, Information and Multimedia, Adelaide, January 21–23’.

The *SC*-based PIR protocols assume that the *SC*'s secure memory can store a small number of data records. This assumption does not hold for multimedia databases, where records can be very long and storing even a single (and very long) record in *SC* can be a problem. Thus, task of designing a PIR protocol for databases with very long records is an interesting open problem.

Acknowledgements

Peishun Wang was supported by the RAACE scholarship, Macquarie University. Huaxiong Wang is supported in part by the Australian Research Council under ARC Discovery Project DP0665035 and the Singapore Ministry of Education under Research Grant T206B2204. Josef Pieprzyk was supported by the ARC grant DP0987734.

References

- Ambainis, A. (1997), Upper bound on the communication complexity of private information retrieval, *in* 'Proc. of the 24th ICALP'.
- Asonov, D., Freytag, J.-C. (4/2002), Almost optimal private information retrieval, *in* 'Proceedings of 2nd Workshop on Privacy Enhancing Technologies (PET2002)', San Francisco, USA.
- Asonov, D., Freytag, J.-C. (5/2002), Private information retrieval, optimal for users and secure coprocessors, Technical Report HUB-IB-159, Humboldt University Berlin.
- Beimel, A., Ishai, Y., Kushilevitz, E., Raymond, J.-F. (2002), Breaking the $O(n^{1/(2k-1)})$ barrier for information-theoretic private information retrieval, *in* 'Proc. of the 43st IEEE Sym. on Found. of Comp. Sci.'
- Chang, Y. (2004), Single Database Private Information Retrieval with Logarithmic Communication. *in* 'The 9th Australasian Conference on Information Security and Privacy (ACISP 2004)', LNCS, 3108, pp. 50-61, Springer-Verlag, Sydney, Australia.
- Chor, B., Kushilevitz, E., Goldreich, O., Sudan, M. (1998), Private information retrieval, *in* 'Journal of the ACM', 45. Earlier version in FOCS 95.
- Chor, B., Gilboa, N. (1997), Computationally private information retrieval, *In* 'Proceedings of 29th STOC'.
- Gentry, C., Ramzan, Z. (2005), Single-Database Private Information Retrieval with Constant Communication Rate, L. Caires et al. (Eds.): *in* 'ICALP 2005', LNCS 3580, pp.803-815.
- Iliev, A., Smith, S. (2003), Privacy-enhanced credential services, *in* 'PKI Research Workshop', volume 2, Gaithersburg, MD. NIST.
- Iliev, A., Smith, S. (2004), Private Information Storage with Logarithmic-space Secure Hardware, *in* 'Information Security Management, Education, and Privacy'. Kluwer, pp.201-216.
- Iliev, A., Smith, S. (2005), Protecting Client Privacy with Trusted Computing at the Server, *in* 'IEEE Security & privacy'.
- Sion, R., Carbunar, B. (2007), On the computational practicality of private information retrieval, *in* 'NDSS'.
- Smith, S. W., Palmer, E. R., Weingart, S. H. (1998), Using a high-performance, programmable secure coprocessor, *in* 'Proceedings of the 2nd International Conference on Financial Cryptography'.
- Smith, S. W., Safford, D. (2000), Practical private information retrieval with secure coprocessors. Technical report, IBM Research Division, T.J. Watson Research Center, 2000.
- Smith, S. W., Safford, D. (2001), Practical server privacy with secure coprocessors, *in* 'IBM Systems Journal', 40(3).
- Wang, S., Ding, X., Deng, R., and Bao, F. (2006), Private information retrieval using trusted hardware, *in* 'ESORICS', LNCS 4189, pp. 49-64.
- Woodruff, D., Yekhanin, S. A. (2005), Geometric Approach to Information-theoretic Private Information Retrieval, *in* 'CCC', pp.275-284.
- Yang, Y. Ding, X. Deng, R., Bao, F. (2008), An Efficient PIR Construction Using Trusted Hardware, *in* 'ISC', LNCS 5222, pp. 64-79.

Advantages and vulnerabilities of pull-based email-delivery

Natascha Chrobok

Andrew Trotman

Richard O'Keefe

Department of Computer Science
University of Otago,

PO Box 56, Dunedin 9054, New Zealand

Email: nat@cs.otago.ac.nz, andrew@cs.otago.ac.nz, ok@cs.otago.ac.nz

Abstract

Over the last decade spam has become a serious problem to email-users all over the world. Most of the daily email-traffic consists of this unwanted spam. There are various methods that have been proposed to fight spam, from IP-based blocking to filtering incoming email-messages. However it seems that it is impossible to overcome this problem as the number of email-messages that are considered spam is increasing. But maybe these techniques target the problem at the wrong side: it is the email-delivery protocol itself that fosters the existence of spam. What once was created to make internet-mail communication as easy and as reliable as possible became abused by modern day spammers. This paper proposes a different approach: instead of accepting all messages unquestioned it introduces a way to empower the receiver by giving him the control to decide if he wants to receive a message or not. By extending SMTP to pull messages instead of receiving them an attempt to stem the flood of spam is made. The pull-based approach works without involvement of the end-users. However this new system does not come without a price: it opens the possibility of a distributed denial of service (DDOS)-attacks against legitimate mail-transfer agents. This vulnerability and possible ways to overcome it are also discussed in this paper.

Keywords: Spam, SMTP, Pull-based email retrieval, Denial of Service

1 Introduction

Since the early days of the internet, email has been an important part of electronic communication between people. While there are numerous ways to exchange information, email still seems to be one of the most popular ways to communicate. Used in private as well as in business environments, electronic mail became an important part of our daily life. But with all the positive aspects of communication, communication with email has its dark side: spam.

Over the last decade unsolicited bulk email, commonly known as spam, has become an increasing problem to email-users all over the world. While in the beginning it was just annoying to delete all the unwanted email-messages in the inbox, the public perception of spam changed dramatically.

But these unwanted email-messages are not only annoying end-users, they also cost tremendous

amounts each year. Companies and public institutions are spending considerable effort and money to find ways to stem the further spread of spam. The yearly costs of spam are estimated to be as high as US-\$50 billion (Ferris-Research 2005). However it seems that no matter how hard we try the spammers always seem to be one step ahead. There are estimations that approximately 90 percent of all email messages could be considered spam (Symantec 2009). Spam costs businesses a lot of wasted bandwidth that could be used elsewhere. It can be considered wasted because it is used for receiving data that will most probably be deleted after it has been received. Also CPU time on mail-servers has to be devoted to process and filter all the incoming messages. Yet there is no guarantee that all unwanted messages will actually be filtered. Those spam-messages that get through to the end-user still need to be deleted manually which costs precious human time. However there is still the risk that regular email-messages might be mistakenly recognized as spam.

In the past few years a vast number of proposals to prevent spam have been made. However it seems that these countermeasures are not effective as spam is still with us. It is as if one of the blessings of the information-age became the ultimate curse: email-users (both corporative and private) all over the world find themselves in the grasp of spammers.

In this paper a closer look is taken at the various techniques of spam prevention. A definition of spam and its origins is given. It discusses the advantages and disadvantages of SMTP and shows how this protocol can be abused. As a huge amount of email-spam originates from illegitimate sources like the botnets, a suggestion to extend SMTP by adding a pull-based approach to make it more robust against misuse is made, and possible ways to introduce the extension are discussed. As the pull-based approach might be vulnerable to distributed denial of service (DDOS) attacks, this issue is discussed and possible solutions to overcome this vulnerability are given.

1.1 What is spam?

Unsolicited bulk email (UBE), also known as email-spam, comes in the form of email-messages for which the recipient has not granted verifiable permission to be sent and which are sent as part of a larger collection of messages (Spamhaus-Project 2009). However this definition is not precise because there also exists spam related to SMS, IP-telephony, chats, web-forums - there even exists YouTube-spam. What all these versions of spam have in common are the following characteristics:

1. Spam comes in the form of electronic messages, which are
2. sent in bulk and are

Copyright ©2010, Australian Computer Society, Inc. This paper appeared at the Australasian Information Security Conference 2010 (AISC2010), Brisbane, Australia. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 105, Colin Boyd and Willy Susilo, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

3. unsolicited.

This definition is valid for all types of spam. However in this paper spam is restricted to email-spam.

1.1.1 Types of Email-Spam

Email-spam comes in different types. According to the intention of the spammer the bulk-emails can be categorized in different ways. Over a one month period (01.08.2009 - 31.08.2009) the universities spam-filter (PureMessage by Sophos) filtered 1471 spam-messages which were dedicated for one of the author's email-addresses (see Figure 1).

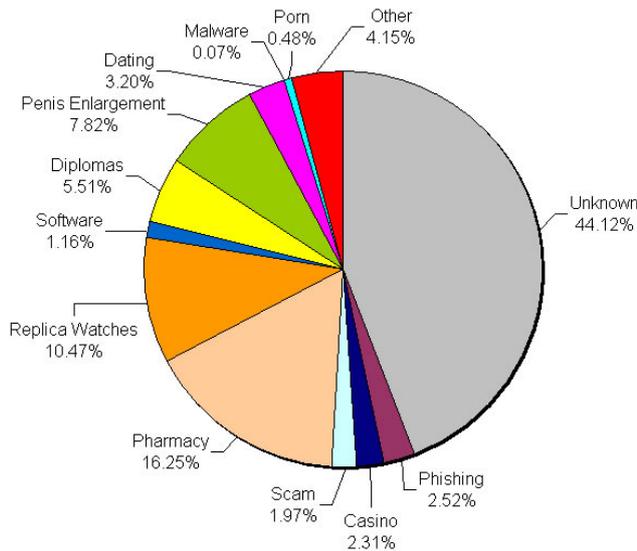


Figure 1: Spam received August 2009

These received spam-messages mostly consisted of advertisements. But there were also fraudulent messages (scam & phishing), messages containing links to adult content as well as a message containing malicious software. Most interesting was the fact that more than 44% of all received spam-messages were not intended for english-speaking recipient and without proper text-encoding (which resulted in unreadable junk-messages).

Based on the received messages, spam can be categorized as

1. advertising
2. fraudulent
3. malware-spam

The best known versions of spam are for pharmacy products or replica watches, university diplomas, online-casinos and offers for the enlargement of certain body parts. But not all advertisements need to be of commercial nature. Sometimes spam is used to propagate political or religious ideas.

The second category is that of fraudulent emails. It consists of spam messages that target naive email-users in the form of scam or phishing-messages. Unlike commercial spam the purpose of this kind of unsolicited email is criminal. Usually the goal is to get money from the recipient - either by persuading them to send money or to reveal their bank or creditcard-information. The most popular scam emails are the so-called 419-scam messages (Levy & Arce 2004) - 419 is the international prefix number of Nigeria where most of these messages originate.

The last category is malware-spam. Its main purpose is to install malicious software on the recipients computer. This malicious software could be used to gather information like email-addresses from the victims PC or it could turn the computer into a so-called zombie-PC - a remotely controlled computer which is part of a global network of hijacked computers, called the botnets. The computers in a botnet are used to send spam, host phishing sites or for cyber-warfare.

1.1.2 Distribution of Spam

The way spam is distributed has changed significantly over the last decade. While before spammers used their own mail-servers to send spam, legislative anti-spam measures such as the can-spam act 2003 and efforts from the anti-spam community have forced them to use different ways to distribute their emails. They reacted by either moving their servers into off-shore countries that have no anti-spam legislation or by abusing badly configured, third-party mail-servers. Recently the distribution of spam via the botnets, vast armies of remotely controlled zombie-PCs has dramatically increased (Herley & Florencio 2008). Estimations are that about 80% of all spam messages sent are originating from the botnets (MessageLabs/Symantec 2009). So it is essential to take the botnet as spam-generator into consideration when searching for ways to reduce the number of unsolicited email-messages.

1.1.3 Why do spammers send Spam?

A lot of spam-messages are filtered or deleted and only a very small percentage of it reaches its recipient. Nevertheless spammers tend to send millions of messages each day. The reason why they still send advertisements is that there are some people who are buying those products. It is believed that for 100 boxes of Viagra sold a spammers margin of profit could be between US-\$ 1,000 and US-\$ 2,000 (Spammer-X 2004). Even if it is very annoying for most of the email-users to have their inboxes flooded with unwanted email-messages, it is this tiny percentage of people who positively respond to spam to keep spammers sending. According to the results of a study carried out by researchers from University of California, Berkeley and UC, San Diego (UCSD) the revenue rate of spammers is very low compared to the number of spam sent (Kanich et al. 2008): by getting control over parts of the Storm-botnet they were able to monitor the distribution of three spam-campaigns over the time period of 26 days. During this period more than 350 million email messages containing pharmacy-related spam were distributed resulting in only 28 sales (roughly 0.00001% of all spam-messages sent). Thus the average daily revenue rate was about US-\$100. However the researchers controlled only a small part of the botnet (about 1.5% of all worker-bots), so they estimate the daily revenue rate for the whole botnet is about US-\$7,000. Considering the large number of spam-messages sent and the small number of sales leads to the conclusion that the only way for spammers to increase their profits is to send more spam.

2 How to abuse SMTP

2.1 Overview

One of the reasons why email became so popular is the way messages are sent via the Simple Mail Transfer Protocol (Klensin 2008). The original standard for message forwarding via SMTP was created in 1982

and after nearly 30 years mail-providers still use this protocol. There have been extensions to the protocol over the years to provide new functionality but SMTP is still essentially the same protocol as it was in the beginning.

Unlike other internet protocols which are pulling information from servers SMTP works the other way round: it pushes messages from the sender to the receiver. While we know exactly which website we want to browse we usually have no idea who wants to send us mail. Therefore SMTP leaves the responsibility that a message reaches its recipient with the sender. In the early days of internet-mail this was acceptable. There was a limited number of users and everybody could be trusted. Most of the users were members of universities or government agencies. It was unthinkable that one of the users of the email-network would abuse the system for his personal gain. So there was no use for more sophisticated security features. It was more important that a message was delivered to its recipient. The original design of the protocol lacked proper security features that would make it more resistant to misuse.

2.2 Simplicity of SMTP

The advantage of SMTP always was its simplicity. However exactly this simplicity eventually led to the problems with spam we have today. Anybody can send a message - whether it is wanted or not. The receiver then has to decide if he wants to read the message or not. This is because there is insufficient information about the content of an email available before the data is sent.

An email consists of two main parts: the envelope and the content. The envelope consists of the email-address of the originator, any number of recipients and optionally additional information for protocol extensions. The content is sent in the SMTP-DATA protocol unit and describes the actual internet message as defined by RFC 2822 (Resnick 2001). It includes the subject, the body of the message, any attachments as well as meta-information (such as sender, recipient(s), sent and received dates, or any other useful data) in the message header.

SMTP uses a small set of commands which makes it easy to implement. It is possible to send an email message without using a mail-client or a mail transfer-agent (MTA). By using telnet it is possible to connect to a mail-exchange server and to send a message by simply typing the correct SMTP-commands in the right order. If the content of the message looks legitimate, the chances are high that it will be delivered (and not filtered out). It is exactly this simplicity that makes SMTP so vulnerable and allows spammers to abuse it to distribute their messages.

2.3 Sender-Information provided by SMTP

Mail transactions in SMTP consist of three steps: the MAIL-command which specifies the sender identification, one or more RCPT-commands providing the receiver information and the DATA-command followed by the actual email-content. The information SMTP provides before the DATA-command is limited. A receiving mail transfer-agent (RMTA) only knows the following information:

- IP-address of the sending mail transfer-agent (SMTA)
- the phrase the SMTA authenticated itself with using the EHLO command (this might be a domain- or computer-name but could also be a random sequence of characters)

- the senders email-address (which does not necessarily have to be correct)
- the email-addresses of the recipient(s)

Of all this information the IP-address is the only reliable information. Of course even an IP-address can be spoofed (Savage et al. 2000), but in this case the senders IP-address is needed for protocol-communication. However all the other information provided by the SMTA can be faked.

According to RFC 5321 the SMTA has to identify itself with the EHLO-command. Usually this identification is a full qualified domain name, so the identification might look like

```
EHLO example.com
```

However it is legitimate to use other address literals if a domain-name is not available for whatever reason. Other legitimate address literals are IPv4-addresses (enclosed by brackets), IPv6-addresses, and other ways of addressing. As there are so many legitimate ways a SMTA might identify itself it is nearly impossible to determine whether or not a given address literal is valid. Most SMTP-servers therefore accept any combination of literals in the EHLO command. So for instance an identification sequence like

```
EHLO spamspamwonderfulspam
```

would be accepted by most mail-servers. Thus the identification provided in the EHLO command is of no value for the RMTA.

The senders email-address is also of no use to determine if the email is originating from a trustful source. Every valid email-address is allowed (and the address does not need to match the one stored as sender-address in the email-message). The purpose of this address is to have a return-address in case a message could not be delivered. The notification that there was a problem can be sent to this address. This cannot be used as information to find out whether the senders address is valid or not.

The next information provided is the email-address of the recipient (or the addresses if the mail is sent to several recipients). It is possible to check if a given receivers email-address exists on the receiving mail-server so it is possible to reject a message if the receiver is not valid. This procedure is usually not used for security purposes as spammers often send emails to randomly generated addresses. It would be very unwise from a IT-security stance to reject invalid recipients as spammers then could find out which email-addresses are valid by using trial-and-error systems.

So the only way to determine if the sender is trustworthy at this point is to use the IP-address. The RMTA could check in a white- or blacklist if this IP-address can be trusted or not. However there is still the chance that the SMTA is working as a relay and thus not the originator of the message. Not even the IP-address therefore could be used as a way to identify if the message is from a trusted source.

Currently the only way to determine if an email-message is spam or not is to receive the whole content of the message. The various parts of the message can then be analyzed by spam filters. Unfortunately this means that the whole SMTP-process has to be performed, i.e. the whole message has to be received. In the case of a spam-message (which would be dropped immediately if recognized as such) this is both a waste of bandwidth (for the delivery-process) as well as storage (on the RMTA until the filtering has been done).

Method	Side	Effect
TCP-Blocking	Sender	Blocking the ports usually used for mail-transmitting
Limitation of Emails	Sender	Limiting the number of emails that can be sent
Micro-payment	Sender	Charging a small fee for each outgoing email
Blacklisting	Receiver	Using a list of IP-addresses which should be blocked
Whitelisting	Receiver	Using a list of IP-addresses that are always accepted
Greylisting	Receiver	Delaying the mail-transfer by rejecting the first connection
Authentication	Sender/Receiver	Incoming connections must be authenticated first before any mail-traffic happens
Challenge/Response	Sender/Receiver	Sender must correctly respond to a challenge sent by the receiver
Filtering	Receiver	Analyzing the messages to determine if they contain spam or not

Table 1: Methods to overcome spam

3 Review of methods to overcome spam

There are several methods to overcome unsolicited emails (Hayati & Potdar 2008). Table 1 shows an overview of the most popular methods. They are used to fight spam at different stages of the email delivery process. There are some measures that can be applied at the senders side, but the majority of the anti-spam methods are on the receiver side. This is mostly because the spammers have more control of the sender side. However this does not mean per-se that spam could not be prevented at the beginning of the delivery process. Such methods can only be applied if the ISP on the sender-side is willing to apply them.

3.1 Sender-side methods

3.1.1 TCP-blocking

One method of spam prevention is TCP-blocking. In this approach the ISP blocks TCP-port 25, which is the one used by the SMTP protocol. This makes it impossible for clients in the ISPs network to send email-messages via this port. So this simple method can prevent infected zombie-PCs from sending spam.

However it makes it impossible for clients in this network to run their own valid mail-servers or to connect to other SMTP-servers (such as freemail-services like gmail). For this reason most ISPs refrain from using this method.

3.1.2 Limitation of emails

Another way to prevent spam on the sender side is by limiting the number of emails a client can send in a certain period of time. So an ISP could impose a limit of 100 outbound email-messages per day which would be more than enough for the majority of its users. If an email-client exceeds this limit, the ISP could either deny sending the message or inform the client that he or she exceeded the limit.

3.1.3 Micropayment

Micropayment is a system in which every time an email-message is sent, the sender is charged a small amount of money, for example 0.0001 Dollars. While this amount is so small that it would be negligible to regular email-users it would be very expensive for spammers sending millions of emails every day. As charges can be reckoned by ISPs this approach is possible, however it does not take into account what happens to unsuspecting users whose PCs have been hijacked.

3.2 Receiver-side methods

3.2.1 Black and Whitelisting

Another approach is IP-based blocking. When a mail-sending host connects to the RMTA the first information the receiver gets from the sender is his IP-address. Spammers might use a number of ways to hide their identity, but they have to give away the IP-address. To make bidirectional IP-based communication such as SMTP possible the receiver needs to know the IP-address at the other end.

This information is used to determine if an SMTP-session with the sender should be accepted or not. There are two ways to use this information: black- and whitelisting. Blacklisting is to determine if the IP-address of the connecting host has sent spam in the past. This is achieved by querying a list of IP-addresses. These lists could be maintained by ISPs or by anti-spam organizations that provide them to third-parties. Some blacklists like the Spamhaus block list are DNS-based. Blacklists contain IP addresses of hosts of known spammers, open relays and proxies. If the IP-address of the connecting host is on this blacklist, the connection is refused and no email-data is received.

Whitelisting is exactly the opposite of blacklisting: a list of trustworthy IP-addresses is used to determine if an incoming request is from a trustworthy source or not. It is not uncommon to use both, black- and whitelists in combination. The problem with using lists is that they tend to be large and need to be kept up-to-date.

3.2.2 Greylisting

Greylisting is based on the reliability of the SMTP-protocol: SMTP-servers will try to resend an email if an attempt to do so fails. It is assumed that the software used by spammers has a lax implementation of the standards, so they might not resend an email if the first attempt to deliver the message did not work. Greylisting also makes use of black- and whitelists

and it has proven to be a quite effective way to protect email-servers against the flood of spam. Combining greylisting with black- and whitelisting appears to be a very effective way to prevent spam. Unfortunately it can also block regular email, e.g. when the sending host is part of an email-cluster with different IP-addresses.

3.2.3 Authentication

Authentication for email-delivery is often used by ISPs and freemail-providers. Until recently it was common to have open SMTP-servers which could be abused for sending emails. This has changed with SMTP-extensions like SMTP-AUTH (Myers 1999) which require the email-user to authenticate with a username/password combination before the SMTP-server can be used. Authentication is a successful method on the sender-side to prevent spammers from using an SMTP-server.

It is very popular for spammers to forge the sender's email-address in the email envelope. There was need for a technique to prevent this forgery and to make it impossible to abuse the email-addresses of unsuspecting victims. The solution for this problem is the sender policy framework (SPF), which prevents the forgery of email-addresses. It is possible to store the SPF-data in the Domain Name Service (DNS) TXT-entries. This allows receivers to find out which hosts are allowed to send emails for a domain by making a simple DNS-query.

3.2.4 Cryptographic authentication

In cryptographic authentication, a digital signature is added to an email-message. A popular approach is the Domainkeys identified Email (Allman et al. 2007) which attempts to prevent spammers from forging source-domains. This allows domain-based black- and whitelists to be more effective.

3.2.5 Challenge/response mechanisms

Challenge/response uses a form of verification mechanism to determine if or not the sender is legitimate. Incoming messages from unverified sources are held in a queue and a challenge is sent back to the sender. This could be a simple mathematical problem (e.g. $5 + 4 = ?$) or a CAPTCHA-picture. A legitimate user can respond to this challenge by sending a solution back to the receiver. While this method can be very effective against spam it might make email-communication confusing to some end-users as they don't expect to solve puzzles when sending an email.

Automated mailing-services like mailinglists, newsletters, etc. cannot respond to challenges. If both, sender and receiver use challenge/response mechanisms it could happen that challenges sent by the one result in challenges by the other and thus an endless loop of challenges is created.

3.2.6 Filtering

One of the most successful attempts to attack spam is the use of filters. There are numerous approaches to filter messages (Cormack 2006): Rule-based filters use a large set of freely configureable rules to determine if a given email contains spam. Bayesian filter systems calculate the probability of an email-message being spam while signature-based filters make use of methods such as hash-algorithms to find out if a message can be trusted or not. Modern spamfilters are highly sophisticated programs that use a combination of these three techniques and have a high rate of success at finding spam.

But spammers are always finding ways to prevent their messages from being filtered. The crux of the matter is that filtering means that we simply accept spam flooding our inboxes. Filtering might relieve the end users from huge numbers of unwanted messages but it still means that spam uses bandwidth as well as storage and CPU-time at the receiving mail-servers. It is desirable to prevent spam without brute-force filtering every incoming email-message.

4 A pull-based strategy to prevent spam

The biggest problem in successful spam-prevention is the SMTP-protocol itself as it fosters its own abuse. In most cases the spam-email has been received and the only thing to do is to limit its effects to the end-user by filtering or finding out if the sender is trustworthy. And even the methods that try to solve the problem at the beginning of the email-delivery process are attempts to compensate the deficiencies in SMTP. The big question is: why are people still using a nearly thirty year old protocol when it is the source of all the trouble?

SMTP does a good job at delivering emails, even if exactly this advantage is also abused by spammers to deliver their spam. The other reason SMTP is still in use is that it is one of the most used protocols on the internet. Given that email is a vital part of daily communication both for business and private users, it is difficult to imagine a world without email. The wide use of SMTP makes it hard to be replaced with a newer, more secure protocol. Billions of internet-users expect their emails to be delivered, regardless of which system is used to deliver the message. Planning a replacement would require a world-wide agreement that SMTP has to be replaced. Even if this task were successful there is still the question of how to replace it. There is no question that there is need of a transitional period of several years in which both the old and the new protocols would co-exist. A first step could be an agreement between large freemail-providers like Yahoo!, Google, Microsoft, etc. and the major ISPs to replace SMTP. The optimistic assumption is that after planning the transitional period more and more email-service-providers would jump on the bandwagon as they won't want to be locked out from global communication via email.

But what would a replacement to SMTP look like? A new protocol should have the same features as SMTP, but without its drawbacks:

- it should be easy to use for end-users
- it should be compatible with existing internet-email standards
- it should reliably deliver legitimate emails
- it should be difficult to abuse this system

the first feature is essential: email-users should not be bothered with any changes to the delivery-protocols. Users should be able to use their email-client with all the functionality they are familiar with. They should not be forced into using new email-applications just because the new standard is not supported by their preferred software. It would be advantageous if a new standard would allow legacy clients to use it - so this would mean no change for the end-user. There should also be no change to the way an email-message is presented to the end-user. Email-address of sender and recipient, subject and body of messages should look exactly the same as they looked in the past.

The other three features are of significant importance for the new protocol: the new protocol

should not introduce new, incompatible mechanics but should work with existing standards. SMTP provides a perfect set of commands to deliver email-messages, so a new approach should be based on existing functionality and extend it instead of using different techniques.

Like SMTP it should be possible to reliably deliver email messages to the recipient. The sender of an email-message should expect that a message will be delivered to the receiver and if this is not possible he should be informed that there was a problem in sending the message.

But unlike SMTP the new protocol should only deliver email-messages originating from a trusted source. Spammers should not be able to use the new protocol the same way they misuse SMTP.

4.1 Pull instead of Push

Internet-email, unlike many other internet-based protocols, is a push-based protocol. This means that communication is initiated by the sender as the receiver does neither know about a message he will receive nor when this message will be sent. But it means that the receiver has to accept all incoming messages, regardless of its content. During the whole delivery-process the control lies in the hands of the sender. The receiver has little influence in this email-delivery process - as long as the sender is a trusted source this procedure is acceptable. But it also makes the receiver vulnerable should the sender abuse this system. There are few methods that give the receiver control over the delivery-process (such as black- or whitelisting).

Pull-based protocols work the other way round. The receiver initiates the communication by requesting information. Thus he has more control over which information is received whereas the sender is only the provider of this information. There are numerous pull-based protocols; with HTTP the most obvious. Using a pull-based approach for internet-email empowers the receiver and gives him control over the delivery-process as he can decide when and what he wants to receive.

Pull-based email was first introduced with Internet Mail 2000 (Bernstein 2000). Instead of forwarding every email-message automatically a notification that there is email available is sent to the recipient. The receiver then decides if he wants to receive messages from this sender or not. In a positive case the receiving MTA pulls the email from the sender. If the recipient does not want to receive mail from the sender, the notification is simply ignored and not responded to. The difference between this approach and SMTP is that during the whole process the email-data is stored at the senders side. This is important as it becomes possible for the recipient to decide which email-messages he wants to receive (and which not) instead of blindly accepting and filtering every incoming message. Although Internet Mail 2000 is simply a conceptual idea there have been several attempts to implement it. The two most notable pull-based email-services are DMTP and Stubmail.

DMTP (Duan et al. 2007) makes use of a combination of classical SMTP functionality, black-/whitelisting and a pull-based approach. By classifying senders into the categories well-known spammers, regular contacts and unclassified senders it allows the receiver to process messages in different ways depending on the sender. While messages from well-known spammers are automatically rejected those from regular contacts (stored in a list on the receivers MTA) are received using the standard SMTP-push mechanism. Messages from unclassified senders (i.e. neither well-known spammers nor in the regular contacts list)

are not received, instead the sending MTA (SMTA) is notified to use the DMTP-protocol. This means the SMTA stores the message and sends a message-key and the subject of the message to the RMTA. The RMTA then generates a email-message, containing the information given by the SMTA and sends it to the end-user. Receiving this notification the end-user has to decide if he wants to receive the message or not. In the case he wants the message he responds to the RMTA. The RMTA then retrieves the message from the SMTA using the message-key and adds the SMTA to its whitelist. Future messages from this SMTA will be automatically accepted. Considering that it is unlikely that all possible spammers are stored in the blacklist and not all legitimate senders are in the whitelist means that many messages will be from a unclassified source. The result is that the end-users mailbox is flooded with email-notifications. There is a risk that the end-user accidentally accepts spam-messages (resulting in whitelisting a spam-source) or rejects legitimate mails. Also the sheer mass of notifications could be perceived as annoying as spam.

Stubmail (Wong 2006) also uses a pull-based email-retrieval approach by combining classical SMTP-based internet-mail with HTTP. It checks if the receiver supports Stubmail or not and then decides how to deliver the message. If the recipient supports the new protocol extension, a key is created and a notification (the so-called stub) is sent to the receiver. Should the receiver of the message decide to read the email-message he has to pull it from the senders server. To find the address of the server on which the downloadable messages are stored the receiver must make a special DNS request. Once these address is known the message can be retrieved by an HTTP-post request. Like DMTP the number of notification could be irritating to the end-user. By using HTTP to retrieve a message it could be possible to download malicious software to the receivers PC.

Despite there being working reference-implementations Internet Mail 2000 is not universally used as a replacement to SMTP. It seems as though the proposed approaches are not perceived as replacements for the classical internet-mail system. They require too much interaction by the end-user, making it awkward to receive messages without minimizing the chance to receive spam.

4.2 General Delivery: using SMTP for Internet Mail 2000

In our new approach to Internet Mail 2000 we introduce a system which works like the snailmail-approach of *poste restante* (or general delivery). The post-office notifies the receiver that there is mail waiting at the post-office. The notification includes an identification with which the receiver can retrieve the mail from his post-office. If the receiver doesn't gather his mail within a certain time-frame the mail is returned to the sender as undeliverable.

The new functionality can be implemented as an extension to SMTP which means that existing SMTP-services could be easily upgraded to use this approach. Email-communication usually involves at least four agents: the sender's mail transfer client (Outlook, Webmail, etc.), a SMTA, a RMTA and the receiver's mail transfer client (MTC). As the communication between the RMTA and the receiver's MTC uses POP or IMAP, only the first three agents use SMTP. Unlike other pull-based approaches the only SMTA and RMTA use pull-mechanics in their communication. The decision if a message should be retrieved or not is made by the RMTA (i.e. an SMTP-server). No interaction by the end-user is needed to make general delivery work. This means that the end-user can use

the email-service as before by using their preferred client-software.

The communication between SMTA and RMTA is split into two parts. In the first part the SMTA connects to the RMTA and notifies it that an email-message is available and sends a unique identifier for this message. After this notification the connection is closed. It is then up to the receiver to decide if the message should be retrieved or not. The second part happens in the case that the receiving host decides to retrieve the message. It connects to the sender and asks for the message by handing over the unique identifier. The sender then forwards the requested message.

The general delivery extension to SMTP introduces two new commands: GDEL and RETR. One is used for delivering the notification, the other one is used for retrieving the email. As usual these extensions are mentioned in a email-servers response to the EHLO command.

The time-frame of how long a message should be stored at the sending email-host is reasonable. It should be taken into consideration that the receiving host might not retrieve a message immediately. So a time-frame between 24 hours to 48 hours might be appropriate. If a message has not been retrieved after this predefined amount of time the SMTA should send a notification to the sender of the email that the delivery of the message was not successful. Likewise the time that passes between the notification and the retrieval can be determined by the RMTA.

It might occur that a RMTA gets a notification for an email that cannot be retrieved. The reason for that could be a temporarily unreachable sending host or a corrupted unique identifier for the message. In such a case the RMTA should try to retrieve the message a number of times (perhaps 3 to 5 times) over the next 48 hours. If retrieving the message continues to be unsuccessful the RMTA discards the notification and stops retrieving it. If the message to be retrieved was legitimate (and the non-delivery was just because of technical difficulties) there are mechanics to notify the sender that the delivery of the message was not successful, which works quite the same way as the classical SMTP.



Figure 2: GDEL-command

4.2.1 GDEL (General Delivery)

The GDEL command is used by the SMTA to notify the RMTA that there is a message available. Together with the GDEL command a unique identifier for the email-message to be delivered is sent. It is up to the SMTA how to generate this unique identifier - so it could possibly use a hash-value of the email-message. It is possible to send multiple GDEL commands in the case that several messages for the same receiver-host/domain are available. In any case a unique message identifier has to be generated for every message, regardless of the fact that two or more recipients might be of the same receiver-domain. This

is necessary because there is a chance that one message is retrieved while the other one is not.

When used in the response to an EHLO-command this command informs a SMTA that the receiving host is able to use the general delivery extension.

Syntax: GDEL uniqueId

Possible reply codes:

- 250 Requested mail action okay, completed
- 500 Syntax error, command unrecognized
- 501 Syntax error in parameters and arguments
- 502 Command not implemented

Figure 2 describes the usual sequence of commands of a successful notification using the GDEL-extension:

1. The SMTA opens a connection to the RMTA
2. The RMTA returns 220 OK
3. The SMTA sends the EHLO-command
4. The RMTA returns with 250 OK and a list of possible extensions supported, of which one is GDEL
5. The SMTA generates a unique identifier for the email-message and sends it using the GDEL-command
6. The RMTA stores this unique id and returns a 250 OK
7. The SMTA closes the connection by sending QUIT

It should be noted that it is possible to send several notifications in a sequence to the receiving host. After the last 250 OK the SMTA could send another notification using the GDEL-command.

4.2.2 RETR (Retrieve)

The RETR command is used by the RMTA to retrieve a message from the SMTA. The unique identifier of the email-message is passed as an argument. If the unique identifier is valid (ie. the email exists on this server) and the email is destined for the connecting host, the SMTA changes into sending mode and starts sending email using standard SMTP. It is possible to retrieve several emails in this way after one another. In the case the RMTA sends an invalid message-id, the SMTA should respond with a 550-error message. If an email-message is not destined for the connected RMTA, the SMTA also responds with a 550-error message, even if the provided unique identifier is valid.

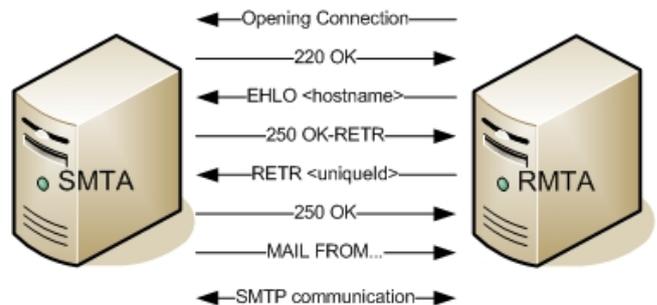


Figure 3: RETR-command

The communication for the retrieval is initiated by the receiver. When the RETR-command is sent

sender and receiver switch roles and the sender starts the SMTP-message-transfer sequence (MAIL FROM - RCPT TO - DATA). After sending the message the roles are switched again to allow the sender to send another RETR-command or to quit the connection. The original SMTP-standard had a similar command called TURN which allowed a sender to become the receiver and vice-versa (Postel 1982). However this functionality eventually became deprecated as it was possible for an unauthenticated client to retrieve messages.

When used in response to an EHLO this command informs the RMTA that the sending host is able to use the general delivery extension.

Syntax: RETR uniqueId

Possible reply codes:

250 Requested mail action okay, completed
 500 Syntax error, command unrecognized
 501 Syntax error in parameters and arguments
 502 Command not implemented
 550 Requested action not taken (e.g. given message-Id not available)

The second part of the protocol using the RETR-extension is shown on figure 3:

1. The RMTA opens a connection to the SMTA
2. The SMTA returns with 220 OK
3. The RMTA sends the EHLO-command
4. The SMTA returns with reply-code 250 OK and also sends a list of the supported extensions (e.g. RETR)
5. The RMTA uses the stored unique id of the email-message it wants to retrieve and sends it using the RETR-command
6. The SMTA compares the requested unique message-id to a list of stored messages, if the message is available it sends 250 OK
7. Both MTA change into SMTP-mode and the SMTA starts forwarding the email-message using the MAIL FROM-command
8. subsequent communication is classical SMTP

As soon as a message is retrieved by a client, there is no need for the SMTA to store it any longer. All subsequent requests for an already retrieved message should be denied with return-code 550 Requested action not taken. It is important for SMTP-clusters where a retrievable message could be requested from several servers is made invalid on all servers belonging to the same cluster. It should be not possible to retrieve a message from a server which has already been retrieved on a server in this cluster.

4.3 Combination with other techniques for spam-prevention

In order to prevent spam effectively it is important to combine several techniques. The general delivery extension to SMTP does not prevent the use of other methods to make the mailing process more secure. It can be combined with other techniques like white- and blacklists to make it easier to determine the senders credibility. By using IP-based lists at the initiation

of the communication process it is possible to prevent connections from illegitimate hosts while allowing host that are on the white-list to connect without problem. This could be used in a way to selectively decide if the general-delivery extension should be used for a connecting host or not. Using black- and white-lists that way is a convenient way for legitimate email-users as the flow of their messages is not disturbed while it is more difficult for spammers to get their emails through.

On the sender side, ISPs could use TCP-blocking to prevent outgoing email-connections to hosts outside of their own network as well as incoming retrieval requests. Many ISPs don't allow the operation of network-based servers for consumer-connections. Power-users and corporate customers could have special contracts with their ISPs that allow them to operate those services. This would make it very hard for infected zombie-PCs to provide SMTP-based services. However this method needs the cooperation of ISPs and it is questionable whether a worldwide agreement could ever be reached. Finally the use of general delivery extension does not exclude the use of filter-software on the receivers side. Although it makes it harder for spammers to get their mail through it does not prevent spam. And there is also the chance that legitimate and therefore trusted hosts might have been compromised and send spam. So the use of spam-filtering complements the mix of effective tools to prevent spam in this scenario.

4.3.1 Advantages

For spammers to be successful it is essential to send as many messages as possible in a short period of time. Using general delivery forces them to store all the messages that they want to send on their server until they are retrieved. Regardless of whether the messages are sent from an email-server in an offshore country or by a zombie-PC in a botnet this means that the SMTP-service must be provided for an undefined timeframe as the spammer does not know when the receiver will try to retrieve the message (if he ever does so). Especially hijacked PCs in a botnet need to stay longer online as they must provide the SMTP-service for incoming retrieval requests. Botnets tend to send their spam in bursts (Xie et al. 2008) – after a period of inactivity the bots are activated and start sending spam for a certain amount of time (usually a couple of hours) followed by another period of inactivity. This time of inactivity is used for botnet-maintenance during which bots might get new instructions, new lists of email-addresses and material for new spam-campaigns. Using the pull-based approach it becomes more awkward for spammers to use bots for propagating spam as they cannot use the hit-and-run tactic anymore. By providing the general delivery service the bots must stay active all the time. This could also become a problem for a consumer-based internet-connection when a lot of retrieval requests are incoming. So one could say that general delivery does not prevent spam, but it makes it more difficult for spammers to get their messages through. The separation of the delivery process into two parts gives the RMTA more time to decide if a message should be retrieved or not. During the time between notification and retrieval the RMTA could run processes to find out if the sender is a trustworthy source. Most notably general delivery is an extension to the standard SMTP-procedure, so both the new mail-service as well as the classical service can be serviced by the same mailing host. This is an advantage during the transitional period as only one service needs to be maintained instead of two parallel running services. After the transitional period

the classical SMTP-functionality could simple be deactivated. General delivery works on protocol-level between email-servers, so no interaction by the end-user is needed. This makes it very convenient as there is no change for the user.

4.3.2 Disadvantages

The new mechanism comes with some disadvantages. The most obvious is the protocol overhead. The communication between SMTA and RMTA produces more traffic than standard SMTP communication. There is the notification and a retrieval. In the case of a successful message-delivery the amount of data transferred between sender and receiver is larger than with the classical protocol. However it should be taken into consideration that not every email-message will be retrieved as the RMTA decides that it comes from an untrustworthy source. Considering the vast amount of spam that will not be transferred we believe it is more than a cheap payoff and therefore worth the additional traffic for legitimate email-communication.

Spammers using a botnet could provide SMTP services on hijacked PCs. However the time between notification and retrieval could be used by the RMTA to determine whether that the sending email-host is a legitimate email-server (i.e. by querying a blacklist, etc.) and thus decide not to retrieve the email.

Another argument against the general delivery extension is that it could be used to make DDOS attacks. This vulnerability will be discussed in detail in the next section.

4.4 Vulnerabilities

Using the pull-based approach for internet-mail has many advantages. The most important of which is that the responsibility for email-storage is moved from the receiver to the sender and that the receiver can decide if he wants to retrieve the messages. However these advantages do not come without a problem. Unlike ordinary SMTP-based services the vulnerability does not lie on the receivers side but on the senders. The pull-based approach makes it necessary for the SMTA to act as a client (when sending notifications) and a server (when providing services for incoming retrieval-requests). During the notification the SMTA has control over the process as it initiates it. On the other side the SMTA has no control over who connects during the retrieval-process. As long as a legitimate client connects everything is fine. However there is the possibility of incoming connections that might not be according to the protocol. The last problem might be a misconfigured RMTA that is repeatedly trying to retrieve a non-existing or already retrieved message.

As retrievable messages are identified by a unique id it is possible that third parties try to illegitimately attain messages by simply guessing the unique id. Particularly implementations with open sources could make this process possible as attackers could write scripts that send retrieve-requests with randomly generated id-keys that are according the key-generation-algorithm. Though the chance to retrieve a particular message is not high, there is still the possibility of generating a valid key. As a certain message can be only retrieved by the RMTA for which it is destined, an attacker needs to pretend to be the correct receiving host. This makes it extremely hard to randomly retrieve messages, but if the attacker has knowledge of emails on the SMTA that are destined for a specific receiver domain, it could be possible to retrieve messages by a brute force attack. Even if the chance of actually retrieving messages is not high it could easily lead to a performance problem as the SMTA

has to process a lot of unnecessary requests. Especially a combined brute-force attack of retrieval requests could lead to a denial of service as the SMTA is unable to process all the requests at once. For this reason it is suggested that further communication is delayed for a reasonable time before another attempt to retrieve a message is possible. There remains the possibility of adding hosts that continuously try to retrieve non-existent messages with wrong message-ids to a blacklist.

The greatest threat to the pull-based approach is that it could be used to intentionally attack a mail-server with a distributed denial of service attack. It is possible to use the message-notification mechanism to force a large number of RMTA to make retrieval-requests even if there is no message actually to be retrieved. So an attacker could send millions of notifications to different RMTA, notifying them that a message is available on the email-server of domain victim.com.

A botnet of tens of thousands of zombie-PCs might generate a tremendous amount of email-notifications. The RMTA will try to contact the mailing-host on which the message is apparently stored. Though there is no guarantee when (and if) RMTA will try to retrieve a message it is obvious that a huge number of requests could easily bring down an SMTA.

One effective way to reduce the vulnerability to DDOS-attacks is to make sure that the notifications are sent from a trusted source i.e. the notification comes from the same address as the message. This means the receiver of a notification has the responsibility to determine whether a notification is from a legitimate origin. So when there is an incoming notification, the receiver should query the IP-address of the connecting host and find out if it belongs to the number of hosts that are allowed to send emails for their domain.

The best way to find more information about the sender of a message is use the DNS. Many domains have an MX-record used to determine which hosts are responsible for mail-exchange. Though usually the MX-records are used to determine the mail-hosts for incoming email-traffic on a domain, the change to a pull-based approach brings to the mail-protocol a means by which they could be used for both in- and outgoing traffic. Using the MX-record would be a misuse of the DNS functionality, but it could be argued that the goal of the pull-based approach is to replace the traditional way to exchange email-messages and so it justifies its use.

But there is another way to determine the identity of the sender - the Sender Policy Framework (Wong & Schlitt 2006). The intention of the Sender Policy Framework (SPF) is to prevent the forgery of email-address senders by explicitly authorizing the hosts that are used for mail-transfer. Using SPF, receiving hosts can query to determine whether a connecting host is authorized to send emails (or notifications in the case of the pull-based approach). SPF makes use of the DNS-protocol TXT entry which is used to store arbitrary text-based attributes. It is possible to define which hosts are allowed to send emails on behalf of a domain. Hosts do not necessarily need to be in the same domain, SPF allows authorizing mail-hosts belonging to other domains. A possible SPF entry for using the pull-based mail-service could look like this:

```
example.com. TXT "v=spf1 mx
a:pullmail.example.com -all"
```

For the domain example.com all outgoing MX-servers are authorized to send (and provide) emails as well as the mailhost pullmail.example.com. All other

hosts are not allowed to send or provide emails on behalf of this domain. When getting notification from an SMTP, all the RMTA has to do is to query the SPF-record for the domain the notification is from and compare it to the IP-address of the connecting host. If the host is in the list of authorized mail-servers, the RMTA can proceed to retrieve the email. In any other cases the notification can simply be rejected. Using this technique is very effective against connections from a botnet because it is unlikely that a zombie-PC has a valid SPF-record. And even if there are SPF-entries for botnet-hosts, the SPF-query can still be combined with a blacklist-query and spam-filtering to make it more effective.

5 Conclusions

This paper discusses the current protocol for internet-email, SMTP and why its architecture (which is focused on reliability and simplicity) fosters the spread of unsolicited email. The various methods and techniques to recognize and prevent spam, both on the sender and the receiver side have been presented.

Especially in a time where huge amounts of spam-messages originate from hijacked botnet-PCs it is important to find new ways to make the distribution of unwanted messages harder. Therefore a pull-based approach to retrieve emails, which is in contrast to the classical push-based approach is suggested.

One advantage of the pull based approach is that the responsibility to store the email-messages is transferred from the receiver to the sender. As the receiver just gets a notification, that there is a message available to be retrieved at the sender's server, it is easy for him to decide if he trusts the sender and gets the email or to just ignore the notification. Therefore bandwidth can be saved as not every message has to be received. The pull-based approach, called general delivery, makes its use as well as its introduction very easy. By just adding two new commands to the set of existing SMTP-commands the new email-pull functionality is provided. As the pull-based approach needs no user-interaction, it can be introduced without end-users interaction.

However the pull-based approach comes not without disadvantages. Most notably is its vulnerability to distributed denial of service attacks — when an attacker sends notifications to various mail-servers that messages can be retrieved at a certain host. This vulnerability can only be reduced on the receiver's side by making sure that the notification has been made by the correct host and not by a third party pretending to be the sender. To make this possible, existing techniques like the DNS-entries of the Sender Policy Framework could be used. The new approach is a way to stem the flood of spam in emails. However it is clear that an effective solution to spam must be a cocktail of various anti-spam measures.

References

- Allman, E., Callas, J., Delaney, M., Libbey, M., Fenton, J. & Thomas, M. (2007), 'RFC 4870 Domainkeys Identified Mail (DKIM) Signatures', <http://www.ietf.org/rfc/rfc4870.txt>.
- Bernstein, D. J. (2000), 'Internet mail 2000', <http://cr.jp.to/im2000.html>.
- Cormack, G. V. (2006), 'Email spam filtering: A systematic review', *Foundations and Trends in Information Retrieval* **1**(4).
- Duan, Z., Dong, Y. & Gopalan, K. (2007), 'DMTP: Controlling spam through message delivery differentiation', *Computer Networks* **51**(10), 2616–2630.
- Ferris-Research (2005), 'The global economic impact of spam, 2005', <http://www.ferris.com/2005/02/24/the-global-economic-impact-of-spam-2005/>.
- Hayati, P. & Potdar, V. (2008), Evaluation of spam detection and prevention frameworks for email and image spam: a state of art, in 'Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services', ACM New York, NY, USA, pp. 520–527.
- Herley, C. & Florencio, D. (2008), 'Nobody sells gold for the price of silver: Dishonesty, uncertainty and the underground economy', Microsoft Research, <http://research.microsoft.com/pubs/80034/nobodysellsgoldforthepriceofsilver.pdf>.
- Kanich, C., Kreibich, C., Levchenko, K., Enright, B., Voelker, G., Paxson, V. & Savage, S. (2008), Spamalytics: An empirical analysis of spam marketing conversion, in 'Proceedings of the 15th ACM conference on Computer and communications security', ACM New York, NY, USA, pp. 3–14.
- Klensin, J. (2008), 'RFC 5321 Simple Mail Transfer Protocol', <http://tools.ietf.org/html/rfc5321>.
- Levy, E. & Arce, I. (2004), 'Criminals become tech savvy', *IEEE Security & Privacy Magazine* **2**(2), 65–68.
- MessageLabs/Symantec (2009), 'Message-labs intelligence: Q2/june 2009', http://www.message-labs.com/mlireport/MLIReport_2009.07_July_FINAL.pdf.
- Myers, J. (1999), 'RFC 2554 SMTP Service Extension for Authentication', <http://www.ietf.org/rfc/rfc2554.txt>.
- Postel, J. (1982), 'RFC 821 Simple Mail Transfer Protocol', [/urlwww.ietf.org/rfc/rfc821.txt](http://urlwww.ietf.org/rfc/rfc821.txt).
- Resnick, P. (2001), 'RFC 2822 Internet Message Format', <http://tools.ietf.org/html/rfc2822>.
- Savage, S., Wetherall, D., Karlin, A. & Anderson, T. (2000), 'Practical network support for IP traceback', *ACM SIGCOMM Computer Communication Review* **30**(4), 295–306.
- Spamhaus-Project (2009), 'The definition of spam', <http://www.spamhaus.org/definition.html>.
- Spammer-X (2004), *Inside the Spam Cartel - Trade secrets from the dark side*, Syngress Publishing.
- Symantec (2009), 'State of spam: A monthly report', http://go.symantec.com/spam_report.
- Wong, M. & Schlitt, W. (2006), 'RFC 4408 Sender Policy Framework (SPF) for Authorizing Use of Domains in E-Mail, Version 1', <http://tools.ietf.org/html/rfc4408>.
- Wong, M. W. (2006), 'Stubmail', <http://www.stubmail.com>.
- Xie, Y., Yu, F., Achan, K., Panigrahy, R., Hulten, G. & Osipkov, I. (2008), 'Spamming botnets: Signatures and characteristics', *ACM SIGCOMM Computer Communication Review* **38**(4), 171–182.

An Administrative Model for $UCON_{ABC}$

Farzad Salim

Jason Reid

Ed Dawson

Information Security Institute,
Queensland University of Technology,
GPO Box 2434, Brisbane Queensland 4001, Australia
Email: {farzad,reid,e.dawson}@isi.qut.edu.au

Abstract

$UCON_{ABC}$ is an emerging access control framework that lacks an administration model. In this paper we define the problem of administration and propose a novel administrative model. At the core of this model is the concept of attribute, which is also the central component of $UCON_{ABC}$. In our model, attributes are created by the assertions of subjects, which ascribe properties/rights to other subjects or objects. Through such a treatment of attributes, administration capabilities can be delegated from one subject to another and as a consequence $UCON_{ABC}$ is improved in three aspects. First, immutable attributes that are currently considered as external to the model can be incorporated and thereby treated as mutable attributes. Second, the current arbitrary categorisation of users (as modifiers of attributes), to system and administrator can be removed. Attributes and objects are only modifiable by those who possess administration capability over them. Third, the delegation of administration over objects and properties that is not currently expressible in $UCON_{ABC}$ is made possible.

Keywords: Access Control, Trust Management, Usage Control, Authorisation, Administration.

1 Introduction

The advent of the Internet and large scale Intranet has led to the emergence of a plethora of new applications (e.g., resource sharing, electronic commerce, health care systems) in which authorisation is significantly different from that of more traditional centralised or closed systems, from two main perspectives: the absence of a central administrator and the lack of prior knowledge held by resource providers and access requesters about each other.

$UCON$ is a new and emerging *abstract*¹ authorisation framework that attempts to combine features from traditional access control, trust management and digital rights management. The concept of $UCON$ was introduced by Park and Sandhu (Sandhu & Park 2003, Park & Sandhu 2002b) and further refined into a formal model of $UCON_{ABC}$ that specifically focuses on the authorisation, obligation and condition aspects of access control (Park & Sandhu 2004).

At the heart of $UCON_{ABC}$ lies the concept of *attributes* and the primary contribution of the model

is the manipulation of attributes throughout an access process. In this model, attributes are considered abstractly as the properties (e.g., role, classification, credit, clearance) of subjects or objects in the model. They are also conceptually divided into two categories, *immutable* and *mutable*. The former is left out of the $UCON_{ABC}$ model, as they are *admin-controlled*, meaning only an administrator can modify them. Mutable attributes are those whose value can change throughout an access process, usually as a consequence of subjects' actions. To eliminate complexities such as who has a right to modify the attributes and how rights are to be assigned and enforced, mutable attributes are considered to be modified by the *system* (i.e., *system-controlled*), an abstraction for a trusted process or user (Park & Sandhu 2004, Park et al. 2004).

The need for a clear definition of an administration and attribute management model for $UCON_{ABC}$ has been mentioned several times in the literature (Park & Sandhu 2002b, Sandhu & Park 2003, Park & Sandhu 2004, Park et al. 2004, Zhang et al. 2004) but left aside for future research. Park and Sandhu (Park & Sandhu 2004), the originators of the $UCON$ concept suggest that "delegation of rights is among the crucial issues that should be covered within $UCON$ framework. In addition, there should be a clear description of administration issues. We believe further studies on these issues will provide more comprehensive solution approaches for the area of usage control." This need has also been noticed by others (Zhang et al. 2007, Luo et al. 2008, Wang & Wang 2007) and attempts have been made to address it through annotating the $UCON_{ABC}$ model with ad-hoc delegation or administration elements. These approaches and their shortcomings will be discussed in more detail in Section 3.

We argue that *administration* is a separate matter, orthogonal to $UCON_{ABC}$ itself. Administration deals with the question of who is supposed to provide the required *policies* for an authorisation. We refer to this as the *Administrative Model*. The latter determines who must be *authorised/denied* given the relevant authorisation policies, which we refer to as the *Usage Model*. We believe that in the context of $UCON_{ABC}$, administration must be addressed through the management of attributes. Therefore, an administration model defines what attributes are, where they come from and who can manipulate them. Further, since objects have attributes, through the administration of attributes, such an administrative model will allow the administration of objects as well. Hence, the connection between the administration model and the $UCON_{ABC}$ model stems from the fact that the former defines attributes and the latter employs them for an authorisation process.

This paper proposes a novel *administrative model* for $UCON_{ABC}$, where the *properties/rights* of a sub-

Copyright ©2010, Australian Computer Society, Inc. This paper appeared at the Australasian Information Security Conference (AISC), Brisbane, Australia. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 105, Colin Boyd and Willy Susilo, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

¹Note that abstractness is a major design principle for $UCON$. By abstract we mean it must be independent of any specific existing language, policy or model.

ject or an *object* are defined by attributes which are formed through *assertions* made directly by a subject or indirectly by others to whom he/she *delegated* the administration capability. A key contribution of the proposed model is its introduction of a two layer structure which comprises a *peer model* and *authoriser model*. The peer model provides an expressive unrestricted environment where every subject may state their *beliefs* about properties and rights of other subjects/objects. Through this, attributes become source centric. Initially the *issuer* of an attribute is the sole administrator of the attribute, until they assert otherwise. The implication of this is the ability to identify who can modify or delegate properties and rights. The authoriser model, sits on top of the peer model and is responsible for determining whose assertions are to be taken into account for an access request. In order to make such an adjudication the model depends on the system/application wide *administrative policy* which we assume subjects operating within the model have consented to. We suggest no specific policy but provide some examples of well known policies such as *owner-based*, used in Digital Rights Management (DRM) or privacy aware applications, and *administrator-based*, for traditional access control systems.

Through such a formulation of the administration problem we make the adequacy of trust management techniques for the administration of $UCON_{ABC}$ evident. In order to communicate our ideas more concretely we use the syntax of the SecPal language (Becker et al. 2007). We use some of the basic constructs that exist in almost any trust management language to show its function in the proposed administrative model. However, the model is not defined or limited by SecPal, since it is not part of the proposed extensions. This is necessarily in keeping with the abstract nature of $UCON_{ABC}$.

Our administrative framework specifically improves $UCON_{ABC}$ in three aspects. First, it lifts the current assumption of a single administrator who issues attributes and the authorisation policies. Second, it removes the arbitrary division of attributes into *mutable* (those that are system-controlled) and *immutable* attributes (those that are admin-controlled). Hence, all attributes are conditionally mutable and can be treated within the current $UCON_{ABC}$ model. Consequently, it would allow the construction of a model with either top-down or bottom-up propagation of administration, in which, administration capabilities over attributes and objects are *delegatable* and the administration root is dynamically determined with respect to a specific application policy. Such dynamism is actually one of the strong advantages of the proposed model. Third, the arbitrary categorization of users as modifiers of attributes into *system* and *administrator* is removed. There only exist subjects, who can modify the attributes they administer or for which they have been delegated administration capabilities.

The rest of this paper is organised as follows. Section 2 outlines the main concepts of $UCON_{ABC}$, on which our paper is based. This is followed by Section 3 that reviews related approaches to the administration problem. Section 4 introduces a motivating example. The actual administration model is described in Section 5 and this is followed by Section 6, where we make concluding remarks.

2 $UCON_{ABC}$ Components

The $UCON_{ABC}$ model (Park & Sandhu 2004) extends traditional access control to address the problem of authorisation not only at the time of access to a re-

source but also during its usage. The main components of the model as shown in Figure 1, are the *Subjects* (S) that wish to use their *rights* (R) over certain *Objects* (O). Subjects and objects are endowed with *Attributes* (A) that capture the properties of these entities. *Authorisation* (AU) is a functional predicate that evaluates usage requests based on the subjects' and objects' attributes, the requested rights, the policy model, and returns either yes or no. In addition to authorisation there are two other decision factors, *obligation* (B) which is a functional predicate that ensures certain obligation actions are performed by the subject and *Condition* (C) predicates, where environmental requirements that have to be satisfied are checked as a part of the usage decision process. The question of how the propositional value of these predicates are determined is currently external to the model.

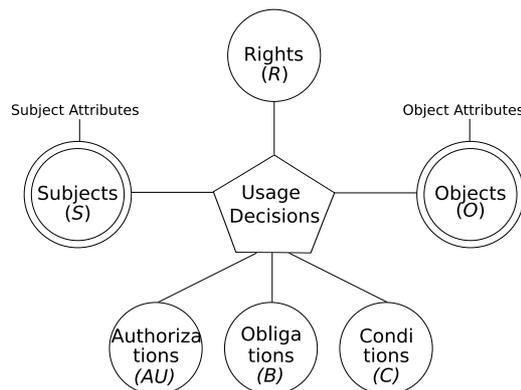


Figure 1: $UCON_{ABC}$ Model

Two innovations for $UCON_{ABC}$ are attribute *mutability* and *continuity*. Different from other access control models, subject or object attributes in this model can not only be modified by the administrator, but can also be changed as a side effect of a subject's usage of an object. However, there is a special user called *system* who is assumed to observe these actions and make proper updates. The concept of continuity proposes that the system repeatedly checks the validity of subjects' rights during the access. When a right is revoked, the access will be terminated on time. These two properties uniquely identify $UCON_{ABC}$ and are indispensable in an open network environment.

3 Related Works

In $UCON_{ABC}$ (Park & Sandhu 2004) attributes are the central component of the model as access decisions are based on the state of subjects' and objects' attributes, which may change and subsequently influence the usage control decision. Despite the important role of attributes in usage control, $UCON_{ABC}$ does not address challenges regarding the management of attributes, focusing instead on user authorisation issues. However, the need for an attribute management mechanism has been noted by $UCON_{ABC}$ designers (Sandhu & Park 2003, Park & Sandhu 2002b, 2004).

In the original $UCON_{ABC}$ papers of Park et al. (Park & Sandhu 2002b, 2004), attributes are divided into 'admin-controlled' and 'system-controlled'. The admin-controlled attributes are said to be immutable in that they are only assigned to subjects and objects by administrator actions and cannot be modified by the system automatically, whilst system-controlled attributes can be updated as side effects of user's us-

age of objects. Park et al. (Park et al. 2004) further refine admin-controlled attributes to be either ‘security-officer-controlled’ or ‘user-controlled’, where the user could be either the subject possessing the property, ‘self-controlled’, or some other user within the model, ‘non-self-controlled’. They leave further details on administration issues for future work. As we will discuss in Section 5, such a syntactic classification of attributes and users is rather arbitrary and meaningless without being able to define attributes and determine how and by whom they can be updated. Furthermore, the above papers always imply the need for a central administrator who assigns attributes and rights to users. This is inconsistent with one of the primary design goals of usage control, to address open environments. The issues regarding delegation are also not considered.

In addition to the above mentioned papers there are a number of other proposals that attempt to provide an administration/delegation for $UCON_{ABC}$. Zhang et al., (Zhang et al. 2007) proposed $UCON_D$, a delegation model for $UCON_{ABC}$. Their view of delegation is limited to access level permissions, where a user grants some of their own rights (e.g., read/write) to another user. They introduce several entities specifically for delegation purposes: ‘delegator’, ‘delegatee’, ‘delegation context’ and ‘permission’, as well as attributes, referred to as ‘delegation attributes’. The contribution of their proposal is to suggest the importance of expressing the relationships between subjects. However, the proposed extension arguably violates the generalised and abstract nature of the original $UCON_{ABC}$ model. Further, there is no clear link between the proposed model and the $UCON_{ABC}$ model.

Luo et al. (Luo et al. 2008) attempt to integrate $UCON_{ABC}$ with ideas from trust management to introduce a distributed delegation model, referred to as $UTCDM$. They use directed graphs as a representation tool to express delegation relationships and to provide a credential discovery algorithm. While credential discovery is one of the main areas in trust management, the link between $UCON_{ABC}$ and their proposed mechanism is unclear. Many well studied trust management and credential discovery frameworks already exist (Blaze et al. 1998, Li et al. 2003) that could be used to address the issues proposed in this paper. However, $UCON_{ABC}$ is meant to be an abstract framework, thus their proposal arguably violates the abstractness principle inherent in $UCON_{ABC}$.

Wang et al. (Wang & Wang 2007) use $UCON_{ABC}$ ’s abstract concept of attribute as ‘credibility’ to express the trustworthiness of subjects who attempt to access an object. From this perspective their proposal is trivial as the concept of attribute in $UCON_{ABC}$ is clearly abstract enough to express trust, credibility, role, etc. They also attempt to establish that, by using attributes as certificates, one can reduce certain complexities such as certificate revocation and certificate discovery that exist in certificate based access control models. However, it is not clear how their proposal would reduce such complexities. There are several theoretical and practical issues that the paper fails to address, such as: who is able to assign attributes to subjects regardless of what they represent, and, how and by whom these attributes are to be modified.

Originator Control (ORCON) is a proposal for an access control policy in which recipients of information need to gain the originator’s approval for the further re-dissemination of the information (Park & Sandhu 2002a). They put emphasis on the importance of adapting the originator control policy within $UCON$ as a means for it to go beyond tradi-

tional access control, trust management models and DRM. However, the paper leans toward implementation rather than an abstract theoretical work. They describe several ways in which licenses and delegation tickets could be used to control redistribution of a resource. As we will describe in Section 5, explicit description of ownership and owner-based control are two examples of a system policy that an abstract framework like $UCON$ must be capable of modelling, but it must not be limited only to this.

On a similar ground to our work, Firozabadi et al. (Firozabadi et al. 2002) and Wood et al. (Wood & Fernandez 1979) address the requirements of a decentralised administration/authorisation model and distinguish between two kinds of delegations: delegation of authority at management level and delegation of permission at request level. The delegation of authority allows an entity to hand off authorities to another entity such that the receiver can express authorisation policies on behalf of the sender. On the other hand, permissions are privileges to exercise the rights of a specific entity - their delegation allows the receiving entity to access resources on behalf of the other. Drawing this distinction allows them to introduce constraints required for each category. However, the main contribution of these proposals is in discussing the distinction between the delegation types rather than providing a formal language to express them. These works have inspired the main ideas that underlie our proposal.

In the following sections we will introduce a general administrative model that is policy agnostic and therefore adheres to the abstraction level inherent in $UCON_{ABC}$, while addressing its limitations in defining, issuing, delegating and dealing with the modification of rights and properties.

4 Motivating Example

To clarify the problem that our administrative model seeks to address, consider the following simple example, taken from (Park & Sandhu 2004) which addresses ‘DRM pay-per-use’ by using $UCON_{preA1}$ ². In this model, two attributes are assumed to exist:

- $credit(s)$: subject’s credit (measurement unit is money).
- $value(o,r)$: object’s value (the amount of money for a given right on object).

There are also two policies:

- $allowed(s,o,r) \Leftarrow credit(s) \geq value(o,r)$.
- $update(credit(s))$: $credit(s) = credit(s) - value(o,r)$.

The policies are intuitively read as: to accept a request the subject (requesting) must have enough credit. In that case the subject’s credit is modified by reducing the value of the object that is being requested and access is granted.

Given the above example, there are three main points that are assumed and left outside the $UCON_{ABC}$ model.

1. It is not clear *who* determines the properties such as subject’s *credit* and object’s *value*. Such properties (in $UCON_{ABC}$ term, attributes) belong to a subject or an object.
2. It is not clear *who* determines the rights for a subject on an object. In the above example, $allowed(s,o,r)$.

²The subscript refers to the pre-authorisation with pre-update policy - interested readers refer to the original paper.

- Although the modification of such properties is at the core of $UCON_{ABC}$, policy governing modification of properties is not explicit. Therefore it is not practical to determine who can modify these properties. In the context of the above example, it is not clear who can update the *credit* property of a subject (i.e., $update(credit(s))$).

$UCON_{ABC}$ simply assumes that $update(credit(s))$ is performed by *system*, an entity outside the model, trusted to do so. By the same token, rights that specify the relationship between subjects and objects are assumed to exist and the model does not care about the origin of the rights.

It is important to appreciate that properties and rights are subjective by nature. In the real world, attributes and rights are acquired from the sources that have the *authority* to provide them. For example, a subject may acquire credit from a bank or a driving license from the traffic authority; those dealing with the subject may trust the bank to honour the provided credit or not. However, the only entity capable of modifying a subject's credit must be the bank or those somehow appointed by the bank to do so. The objective of the administrative model for $UCON_{ABC}$ is to address these issues.

5 Administrative Model (\mathcal{M})

Here we introduce a novel approach in representing administrative functions. We conceptually divide the $UCON_{ABC}$ Administrative Model (\mathcal{M}) into, the *Peer Model* ($\mathcal{M}_{\mathcal{P}}$), that defines an unrestricted basis for establishing relationships between subjects and objects through assertions, and the *Authoriser Model* ($\mathcal{M}_{\mathcal{A}}$), that provides a means for selecting and honouring some of the existing assertions with respect to an administrative policy of the *system* that implements the $UCON_{ABC}$ model.

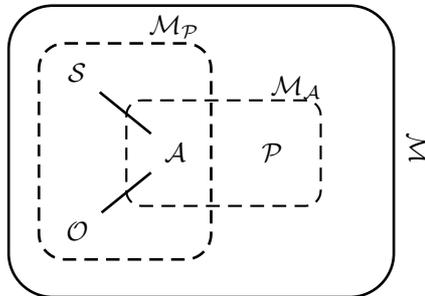


Figure 2: Administrative Model (\mathcal{M})

The administrative model, shown in Figure 2, is abstracted into four major components: *Subjects* (\mathcal{S}) that are inter-connected, *Objects* (\mathcal{O}) that are *administered, owned or used*³ by subjects, *Assertions* (\mathcal{A}) that specify the relationships between subjects and objects, and finally the *Administrative Policy* (\mathcal{P}), denoting *system* policy.

We use a specific name when we are addressing an element of the finite set of \mathcal{S} or \mathcal{O} . For example *Alice, Bob, Carol* $\in \mathcal{S}$ or *File* $\in \mathcal{O}$. Further, we use a lower-case letter (e.g., *s*) when we are addressing a variable.

5.1 Peer Model ($\mathcal{M}_{\mathcal{P}}$): Subjects, Objects, Assertions

At the heart of the $UCON_{ABC}$ model is the concept of attributes and at the center of $\mathcal{M}_{\mathcal{P}}$ is the concept of *assertion* shown in Figure 3. The relationship

³In this paper we consider “use” and “access” to be synonyms.

between assertions and attributes stems from the fact that assertions in our model generate the attributes of the $UCON_{ABC}$ model. Assertions are also used to express *rights, conditions* and *obligations* of $UCON_{ABC}$ as well.

We generalise an assertion to be a *belief statement* expressed by a *subject* about the *property* (\mathcal{T}) or the *Rights* (\mathcal{R})⁴ of another *subject* (including themselves) or an *object*. An assertion could state various things such as what is the subject's role, age, credit balance, clearance or object's classification, value or trustworthiness from the *issuer's perspective*.

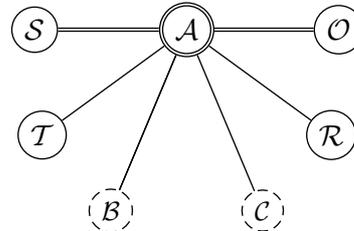


Figure 3: Assertions \mathcal{A}

In relation to expressing *conditions* and *obligations* within $\mathcal{M}_{\mathcal{P}}$, we argue that both of these are objects with properties under the control of a *specific* subject. This is a different view from the former access control models, including $UCON_{ABC}$ that consider these to be a special (type) entity, other than a common object and considers the subject modifying them to be external to the model. Their approach results in a simpler model as they do not need to deal with the manipulation of these entities by users (subjects).

For example, in $UCON_{ABC}$, *local time* is considered as a condition under the control of external subject, *environment*. However, in reality, the local time is the *property* of an object, *system clock*, which is under the control of a subject, *root* in a Unix operating system. By a similar token, in $UCON_{ABC}$, a user's assent to a privacy policy by *ticking* a policy form is considered as the discharge of an obligation. In our model, the *policy form* could be considered as an object with a property, *filled*, that could be assigned values *true* or *false* by a subject (user).

In all the above cases an assertion states that its issuer *believes*⁵ that a subject or an object has a property or a right. Through such a treatment we inherit a *subjective* view, where properties and rights are always formed from their issuer's perspective and are always *valid* from that aspect. We refer to such claims that a subject makes in an assertion as (subjective) *fact*. The grammar of facts are shown in Table 1.

There are two types of assertion that could be written over a fact: a *direct assertion* and an *indirect/delegation assertion*. Direct assertions are the basis of the model and take the form:

$$s \text{ says } fact \langle \text{if } fact_1 \dots fact_n \rangle$$

Informally, a direct assertion states that the subject believes (says) a fact. The assertion may also be conditional upon the existence of some other facts asserted by the issuer. In the above assertion $s \in \mathcal{S}$ is the *issuer* of the statement, **says** is a keyword, *fact* encodes a specific fact based on the grammar of facts.

⁴Note that rights, permission or capability can be viewed as a property; for example granting a right r to a subject can be viewed as making an assertion that the subject has property r . However, here for clarity and to be aligned with the $UCON_{ABC}$ approach we consider them to be separate.

⁵Note that in our paper, believe, assert, state are synonyms, all mean a subject has made an assertion.

$e_{[s]}$	$::=$	s	(variables)
		$s \in \mathcal{S}$	(subject)
$e_{[o]}$	$::=$	o	(variables)
		$o \in \mathcal{O}$	(object)
$pred_{[r]}$	$::=$	$canRead[o \in \mathcal{O}]$	(user-defined (rights) predicates)
		\dots	
$pred_{[t]}$	$::=$	$hasCredit[-]$	(user-defined (property) predicates)
		$hasClearance[-]$	
		\dots	
$fact$	$::=$	$e_{[s]} pred_{[r]}$	
		$e_{[s]} pred_{[t]}$	
		$e_{[o]} pred_{[t]}$	

Table 1: The Grammar of Facts

The *if* is an optional keyword, $fact_1 \dots fact_n$ are facts upon which the *fact* is conditional. The conditional facts must already exist within the *set of assertions* made by s , *Assertion Context* of s (\mathcal{AC}_s), in order to say: s **says** *fact*.

Following is an example of three direct assertions made by three subjects: *Bank*, *Administrator* and *Alice* in an imaginary *UCON* model. They intuitively mean *Administrator* believes that *Bob* has the clearance level 1; *Bank* states (believes) that *Bob* has a credit limit of \$50, and *Alice* believes the object *Book* has a value of \$10 for reading purposes.

- Admin* **says** *Bob* hasClearance[1] (1)
Bank **says** *Bob* hasCredit[50] (2)
Alice **says** *Book* hasValue[10,read] (3)

Direct assertions are less expressive as they are unable to capture the dependency of a subject on another in order to make an assertion that brings about the propositional content of a fact. For example, only using direct assertions, it is not possible for *Alice* to depend on *Administrator* to determine the *clearance* property for *Bob* by:

- Alice* **says** *Bob* hasClearance[1] **if**
Administrator **says** *Bob* hasClearance[1]

This is because the conditional facts in a direct assertion must be deducible from the issuer's assertion context, in this case \mathcal{AC}_{Alice} . Expressing such a dependency is the primary requirement in open environments, where subjects have limited knowledge about the properties or the rights of others. This need is addressed by delegation assertions that take the following form and allow a subject to state its willingness to believe certain types of facts asserted by other subjects:

$$s \text{ says } s' \text{ can say}_D \text{ fact } \langle \text{if } fact_1 \dots fact_n \rangle$$

The above delegation assertion introduces an extra keyword **can say** _{D} which introduces the willingness of the issuer for accepting (believing) the assertions made by another subject s' about the *fact*. The delegations have arbitrary but specified depth, where D defines the possible depth of the delegation and takes the values $n \in \mathbb{N} \dots \infty$ where $D = 0$ means no delegation and ∞ means an unbounded delegation. The depth of $D = 2$ means that a subject (e.g., s) may delegate the assertion of a fact to another subject (e.g., s_1) and allow s_1 to delegate to others but *not* allow these others to delegate further.

- Alice* **says** *Admin* **can say**₀ s hasClearance (4)
Alice **says** *Bank* **can say**₀ s hasCredit (5)

For example, given the direct assertions {1,2} and delegation assertions {4,5}, one (i.e., *Alice* or any one having access to assertions) can deduce:

- Alice* **says** *Bob* hasCredit[50] (6)
Alice **says** *Bob* hasClearance[1] (7)

As a result the assertion context of *Alice* \mathcal{AC}_{Alice} would consist of direct assertions {3,6,7}. Notice that here we assumed subjects (e.g., *Alice*) are informed about the assertions made by other subjects in the model. The details of how such knowledge is shared and complexities regarding chains of assertion are directly related to the application employing our model and several approaches exist to address these issues e.g., (Li et al. 2003, Blaze et al. 1998).

Notice that through the above treatment, subjects within a $UCON_{ABC}$ model are enabled to make assertions about properties and rights of other subjects and objects within the model. Further, they can delegate such assignments to other subjects. Here, despite $UCON_{ABC}$'s approach where attributes belong to subjects or objects, and where it is not clear who has specified the policy, properties and rights only exist from the perspective of their issuers. This does not necessarily mean that it is universally believed (by other subjects in the model) that the subject/object actually has the property/right in question. This leads us to the question, how could the $UCON_{ABC}$ model decide whose (subject) perspective should be relied upon for making an authorisation decision? This question is answered by the authoriser model.

5.2 Authoriser Model (\mathcal{M}_A): System policy, Assertions

The peer model introduces a flexible anarchic model where no authority is assumed. A subject may make an assertion about itself, other subjects or objects and these assertions could be honoured by others through delegation. However, such an anarchic model is meaningless if there are no means to determine whose assertions are to be taken into account for a $UCON_{ABC}$ usage control decision, which must ultimately involve one or more subjects who act as the *authority root*—whose set of assertions is denoted as \mathcal{AC}_{Root} .

The concept of *authority root* is inherent in, and the basis of, all the existing access control models. However, it is usually assumed fixed and defined outside the model itself. For example, in most RBAC models the authority root is a trusted central administrator who is assumed to assign permissions to roles and roles to users. Based on these assignments, the model determines whether an access request is

to be granted or denied. ORCON and privacy oriented models consider the owner of the resource as the authority root and make access control decisions based on this assumption. Trust Management systems consider the authority root to be the principal called *local*, who writes the local policies. Since these models are based on predefined views about the root of authority, none is general enough to model the others.

To ensure the flexibility of our administrative model we bring the concept of authority root into the model and allow it to be explicitly defined through an *Administrative Policy*. The authoriser is defined as a function shown in Figure 4:

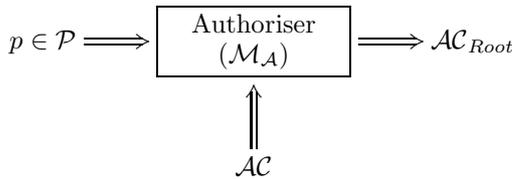


Figure 4: System View: Determining Authority Root

where \mathcal{P} is a set of possible administrative policies. An administrative policy $p \in \mathcal{P}$ precisely states the rules necessary to determine which subject(s) are to be considered as the authority root. The \mathcal{AC} is a set of all assertions made by subjects in $\mathcal{M}_{\mathcal{P}}$. In other words, it is the universal set of assertion contexts. Given p and \mathcal{AC} , $\mathcal{M}_{\mathcal{A}}$ determines the assertion context of the authority root, denoted as \mathcal{AC}_{Root} ⁶.

Note that when more than one authority root is specified by p , there is a potential for conflict between their assertions. The nature of such conflicts depend on the language used for expressing administrative policies. Such a language could allow the expression of hierarchies and would ideally provide mechanisms for detecting inconsistencies or redundancies. Whilst the discussion of such conflicts and their resolution is important for the applicability of our proposal, the focus of this paper is on introducing an abstract administrative framework, free from any specific language or enforcement mechanism. We therefore leave the introduction of a language to express administrative policies for future research. Here, we simply specify administrative policies using the notations used for expressing facts.

To illustrate the generality of our proposal, in the following we will provide two examples that correspond to two separate application settings employing $\mathcal{U}CON_{ABC}$. In one an administrator is the root of authority and in the other, owners of objects are the authority roots. The former covers a majority of traditional access control models and the latter addresses DRM, ORCON and privacy-aware systems.

5.2.1 Administrator-based Model

Let us consider an administrative policy for an organization's database that is under the control of a single administrator, *Clare*. Intuitively, the policy p could be trivially written as the following statements:

$$\begin{aligned} s \text{ isRoot if } s \text{ isAdministrator} \\ \text{Clare isAdministrator} \end{aligned}$$

Now, since *Clare* is considered as the root of authority for the given application, she may enforce a DAC policy through the following assertions, which state, anyone who can read a directory may allow others to read the files in that directory:

⁶Note that *Root* is simply an alias of the subject whom is to be the authority root, not a distinguished subject.

*Clare says s can say*_∞ *s'* canRead[*o*] if *s* canRead[*d*], *o* isElementOf[*d*]

Or in another instance, she may enforce a MAC policy to ensure that the users are only allowed to read the files for which their clearance dominates the file's classification:

*Clare says s canRead[*o*] if s isElementof[*S*], o isElementOf[*O*], level[*s*] ≥ level[*o*]*

Hence, given the administrative policy, $\mathcal{M}_{\mathcal{A}}$ determines the \mathcal{AC}_{Root} , which in this case is the assertion context of the administrator \mathcal{AC}_{Clare} .

Note that although in the above example we assumed a single administrator for the given application, in reality, an application could have many co-existing administrators and this could be reflected in the administrative policy. For example, consider an *operating system*, where the administrator (the root user) has privilege over the system objects (e.g. system directory, system files, printer, etc.) and the private directories of other users on the system are under their control - they can specify rules for sharing their own resources.

5.2.2 Owner-Based Model

Let us now consider another scenario where there is a data store through which subjects can share their documents with others. However, only the owner of the resources can express access control rules for them. A simple administrative policy for such an application may identify a subject as the authority root if the document carries the subject's signature, and this could also be simply expressed as:

$$s \text{ isRoot if } s \text{ signed}[o]$$

Again, assume that there is a subject, *Alice* who would like to allow her friends, and their friends to read the document, *Book*, she is sharing. To do so, *Alice* states her authorisation rules for her book as below:

$$\begin{aligned} \text{Alice says } s \text{ canRead[Book] if } s \text{ isFriend} & \quad (8) \\ \text{Alice says } s \text{ can say }_{\infty} y \text{ isFriend if } s \text{ isFriend} & \quad (9) \\ \text{Alice says } Rob \text{ isFriend} & \quad (10) \end{aligned}$$

Given the above assertions, *Alice* declares *Rob* as her friend and through assertions {8,9} allows *Rob's* friends to read the book that *Alice* is sharing. Now, if *Rob says Mary* isFriend, then *Mary* would also be able to read *Alice's* Book. Lets take this even further and assume that *Alice* would like to allow others who may not be friend to also read the book for a price.

$$\begin{aligned} \text{Alice says } s \text{ canRead[Book] if } s \\ \text{hasCredit[10], } s \text{ isPaying[10]} \end{aligned} \quad (11)$$

From the assertions {3, 5} we can see that *Alice* has determined the value of the book and decided to accept *Bank's* assertions regarding the subject's credit balance. Using the assertions {11}, she further ensures that the book can be read by anyone who has enough credit and (*Alice* believes) is paying the price of the book.

The above examples demonstrate how uncontrolled assertions in the peer model can be regulated and managed based on the policy of an application without restricting our administrative model to any specific policy.

6 Conclusion and Future Work

The contributions of this paper are three fold. First, we introduced a novel administrative model based on two layers of abstraction, capable of representing both centralised as well as distributed administrative requirements to address different application domains. One layer introduces an anarchic environment where every subject, in addition to being an access requester, can be an administrator and specify rights and properties for other subjects or objects. Another layer introduces constraints based on application requirements to identify who can actually administer and update rights or properties. In this aspect, we made the concept of *authority root* that is inherent but external to the existing access control models, explicit and internal. Through this, given an application's requirements, theoretically any number of authority roots could be defined, whom can specify any policy type or delegate such tasks. Thus the design delivers a desirable flexibility.

Second, we analysed the administrative problem in $UCON_{ABC}$ and showed how it can be addressed using existing trust management techniques, as a result, taking advantage of all the already developed functionalities (e.g., certificate delegation, revocation, etc.) that comes with them.

Finally, through concrete examples we showed how our proposed administrative model can address the specific problems identified within $UCON_{ABC}$ model. Precisely, the administrator who was considered a special entity external to the $UCON_{ABC}$ model can now be any subject(s) within the model. An arbitrary division of attributes into *mutable*, modified by subjects, and *immutable*, only to be modified by the administrator, is no longer necessary, since all attributes are made mutable. The arbitrary attribute modifier *system* is removed; attributes are source centric and can only be updated through their issuer or by those whom have been delegated relevant administrative authority over them.

We envisage the immediate future direction is to adopt one of the existing trust management languages to develop an administrative toolkit based on the concepts introduced in this paper. The developed framework is aimed towards a current application of $UCON_{ABC}$ in collaborative environments, such as the one proposed by Zhang et al., (Zhang et al. 2006). Further, we would like to examine the potential conflicts that may arise due to assertions made by multiple authority roots and introduce approaches to deal with them.

7 Acknowledgement

This research has been partially funded by the Australian Research Council - Project DP0773706.

References

Becker, M., Fournet, C. & Gordon, A. (2007), Design and semantics of a decentralized authorization language, in 'Computer Security Foundations Symposium, 2007. CSF '07. 20th IEEE', pp. 3–15.

Blaze, M., Feigenbaum, J. & Strauss, M. (1998), Compliance checking in the Policymaker trust management system, in 'Financial Cryptography', pp. 254–274.

Firozabadi, B. S., Sergot, M. & Bandmann, O. (2002), Using authority certificates to create management structures, in 'Revised Papers from the 9th International Workshop on Security Protocols', Springer-Verlag, London, UK, pp. 134–145.

Li, N., Winsborough, W. H. & Mitchell, J. C. (2003), 'Distributed credential chain discovery in trust management', *J. Comput. Secur.* **11**(1), 35–86.

Luo, X., Yang, Y. & Hu, Z. (2008), 'Controllable delegation model based on usage and trustworthiness', *Knowledge Acquisition and Modeling, International Symposium on* pp. 745–749.

Park, J. & Sandhu, R. (2002a), 'Originator control in usage control', *Policies for Distributed Systems and Networks, 2002. Proceedings. Third International Workshop on* pp. 60–66.

Park, J. & Sandhu, R. (2002b), Towards usage control models: beyond traditional access control, in 'SACMAT '02: Proceedings of the seventh ACM symposium on access control models and technologies', ACM, New York, NY, USA, pp. 57–64.

Park, J. & Sandhu, R. S. (2004), 'The $UCON_{ABC}$ usage control model', *ACM Trans. Inf. Syst. Secur.* **7**(1), 128–174.

Park, J., Zhang, X. & Sandhu, R. (2004), Attribute mutability in usage control, in 'In Proceedings of the Annual IFIP WG 11.3 Working Conference on Data and Applications Security', pp. 25–28.

Sandhu, R. & Park, J. (2003), Usage control: A vision for next generation access control, in 'Computer Network Security', Vol. 2776/2003, Second International Workshop on Mathematical Methods, Models, and Architectures for Computer Network Security, Springer Berlin / Heidelberg, pp. 17–31.

Wang, F. & Wang, F. (2007), The research and application of resource dissemination based on credibility and UCON, in 'CIS '07: Proceedings of the 2007 International Conference on Computational Intelligence and Security', IEEE Computer Society, Washington, DC, USA, pp. 584–588.

Wood, C. & Fernandez, E. (1979), 'Decentralized authorization in a database system', *Very Large Data Bases, 1979. Fifth International Conference on* pp. 352–359.

Zhang, X., Nakae, M., Covington, M. J. & Sandhu, R. (2006), A usage-based authorization framework for collaborative computing systems, in 'SACMAT '06: Proceedings of the eleventh ACM symposium on Access control models and technologies', ACM, New York, NY, USA, pp. 180–189.

Zhang, X., Park, J., Parisi-Presicce, F. & Sandhu, R. (2004), A logical specification for usage control, in 'SACMAT '04: Proceedings of the ninth ACM symposium on Access control models and technologies', ACM, New York, NY, USA, pp. 1–10.

Zhang, Z., Yang, L., Pei, Q. & Ma, J. (2007), Research on usage control model with delegation characteristics based on OM-AM methodology, in 'NPC '07: Proceedings of the 2007 IFIP International Conference on Network and Parallel Computing Workshops', IEEE Computer Society, Washington, DC, USA, pp. 238–243.

Impeding CAPTCHA Breakers with Visual Decryption

Simon R. Lang¹, Neville Williams²

^{1 2} School of Computer Science, Engineering and Mathematics
 Flinders University of South Australia,
 PO Box 2100, Adelaide, South Australia 5001,
 Email: simon.lang@flinders.edu.au, neville.williams@flinders.edu.au

Abstract

Abuse of free Internet resources and services from false account creation, to spam, to identity theft, excessive bandwidth usage, or even vote stuffing online polls is a big problem. The Completely Automatic Public Turing Test to tell Computers and Humans Apart (CAPTCHA) controls access to resources but automated systems are increasingly adept at overcoming them. In this paper a method of access control is introduced as an extra layer of security on top of existing CAPTCHA implementations. It uses visual encryption to encrypt images, which are presented to clients like a CAPTCHA. It's purpose is to compress many sub-images into a small image format that humans can decode visually but is hard for automated systems due to decrypting overhead, and having to process more images to find the hidden image. This paper introduces visual encryption as a viable method to encrypt CAPTCHAs, and tests a prototype to measure how efficiently users can find them. It also measures whether this method could impede a real CAPTCHA breaker. Results show humans detect images within 16-33 seconds, and deciphering images is almost 100%. Estimates on CAPTCHA breaking benchmarks show automated systems would be slowed significantly, even assuming the image is found and decoded. As sub-images increase, humans can process the visually encrypted images faster than automated systems can.

Keywords: Access Control, Authentication, Automated Attacks, CAPTCHA, Cryptography, Security, Spamming, Visual Decryption, Visual Processing.

1 INTRODUCTION AND MOTIVATION

There are many free resources on the Internet that malicious users wish to exploit. Email has seen much abuse over the years, such that legislation in a number of countries has been put in place to punish those that grossly abuse such resources. This in itself has only marginally stemmed the tide of unsolicited email that floods email accounts every day. Most, if not all of these attacks are automated by a computer, or even a network of compromised systems that automatically send unsolicited mail to millions of people worldwide. Other similar services are also vulnerable, and malicious users are quick to exploit any free service that is available to them. Forums, Email services, and social

networking sites are easy targets for spammers (Leyla et al. 2009).

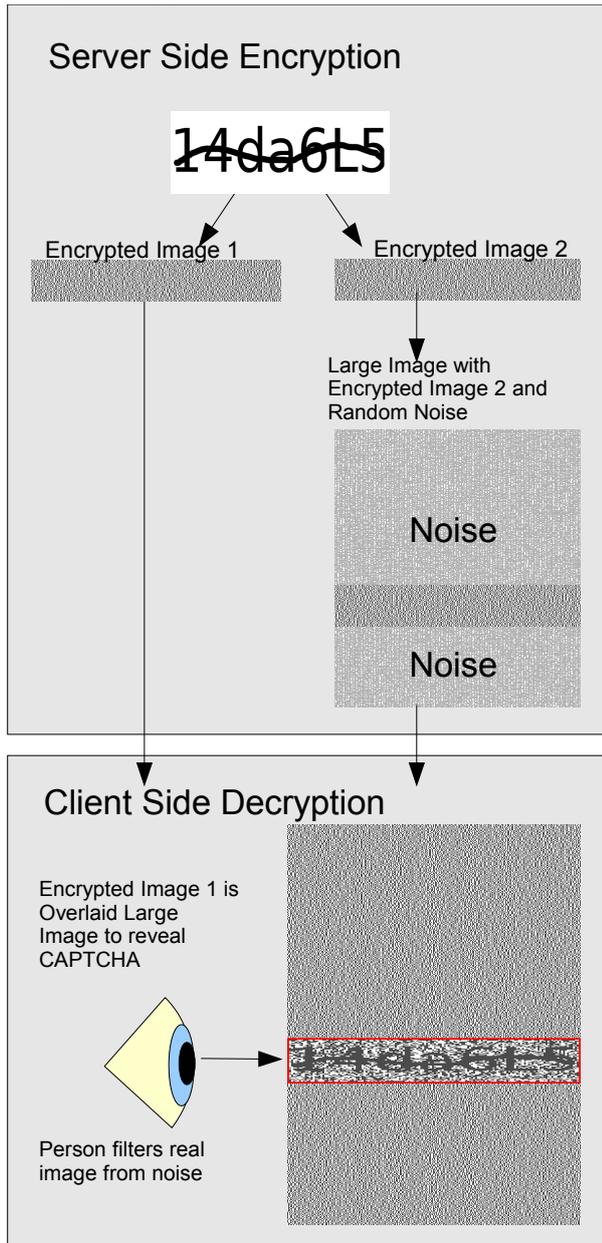
In the past, a weak form of authentication was to register an account with the site whose resources the person wished to use, but nowadays this process is outdated, and easily automated by a computer. An extra layer of security to overcome the limitations of account registration was to integrate into the account creation process a method that could authenticate the unknown party over an unsecured connection as a human, and not a computer. This way the host could be assured that an automated system was not trying to abuse the service.

Currently one of the most popular methods used to authenticate the existence of a human operator over an unsecured connection is the CAPTCHA. The CAPTCHA was first described by Luis von Ahn, Manuel Blum and John Langford (von Ahn et al. 2004). Its main purpose is to prevent computers masquerading as human operators, and to prevent abuse of publicly available services, illegal or otherwise, by automated computer systems. The most common situations in which CAPTCHAs are used are during account creation, comment postings on social networking sites or forums, and, to a lesser degree, file downloading. Malicious persons often try to hijack these services. This usually results in the abuse of these services by automated computer systems until the site operator finds a way to stop them. Common abuses include excessive bandwidth usage, Spam messages, unwanted spidering of content, false accounts being used illegally, theft of personal information, and similar.

This paper introduces an already existing encryption method that is generally useless as a conventional encryption algorithm, and uses it in such a way that it massively expands the search space of an image while harnessing a human's visual systems to quickly and relatively easily decrypt the encrypted CAPTCHA image. The encryption process is fast, low in bandwidth to transmit, easy to implement using browser scripting languages, generally low in client side processing overhead, and can be applied to varying degrees of complexity to further hinder automated attacks. It can easily encompass existing CAPTCHA methods or exist as a standalone implementation. The process involves taking an existing CAPTCHA image and encrypting it into two sub-images. One of the two encrypted images is hidden randomly in a larger sized image. It also includes random noise, the noise is indistinguishable from the original encrypted image both for humans and computers. Both sub-images are then transmitted to the computer or person that must solve the challenge. On the receiving end, a scriptable browser interface overlays the encrypted image without noise, over the larger image which contains the second encrypted image and noise. The person

Copyright ©2010, Australasian Information Security Conference. This paper appeared at the 8th Australasian Information Security Conference (AISC2010), Brisbane, Australia. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 105. Colin Boyd and Willy Susilo, Eds. Reproduction for academic, not-for profit purposes permitted provided this text is included.

Figure 1: Outlines the process a CAPTCHA would undergo to encrypt the image, and visually decrypt it.



must then find the encrypted CAPTCHA embedded in the noise to recover the original CAPTCHA image. To further highlight this, figure 1 outlines the process a CAPTCHA undergoes on server side encryption and client side decryption.

2 EXISTING APPROACHES

CAPTCHAs have encompassed a myriad of different forms. Their variation is limited only by the creator's imagination, which is indicative of the purpose, and that is to be as intuitive as possible for humans, but become as alien and unpredictable as possible for a computer system. This whole paper could easily be devoted to the types of CAPTCHAs that exist, which some have in fact done to better understand the flaws inherent in CAPTCHA design (Yan & Ahmad 2007, 2008b), so this paper will instead skim a few recent and promising variations outside of traditional CAPTCHA designs.

Leaving the character based CAPTCHA alto-

gether is a CAPTCHA based on an image's orientation (Gossweiler et al. 2009). A person is meant to gauge the rotation of a picture that is presented to them. This technique has the advantage that it is language independent, does not require the entry of text, and is more resistant to automated attacks. The sample of participants for this study however was small (16) and showed that even with a rotation error window of 16 degrees, it resulted in an eighty four to ninety four percent success rate when one to three rotation challenges were presented at a time (Gossweiler et al. 2009). The ability to thwart automated systems is also dependent on the repository of images created, though it is likely that randomised 3D scenes could be generated to avoid attacks that build repositories of previous images in order to improve learning, though this attack may be susceptible to image processing that looks for straight lines and other basic shapes to derive horizons, or find other patterns in the placement of objects within the scene.

Differentiation between animals (Philippe 2008) has also been used as a viable CAPTCHA, however this has shown to be easily cracked due to the fact that the features of dogs and cats could be learned by computers (Philippe 2008). Audio based CAPTCHAs for visually impaired persons (Jonathan et al. 2007), are also seen as a much better alternative compared to traditional CAPTCHAs (Jeffrey & Anna 2009). Though audio CAPTCHAs take longer to solve, they are much more accessible to visually disabled persons, but the process of solving an audio CAPTCHA is difficult for human users, and error rates are higher for a variety of reasons (Jeffrey & Anna 2009).

3D images that are generated on the fly at different angles is also proposed as a viable CAPTCHA alternative (Ngo 2009). The CAPTCHA works by presenting the user with a number of 3D images at randomly generated angles, the user is then asked to select the closest variation of a set of three images presented below the 3D image options presented. Whether this CAPTCHA will resist cracking attempts will remain to be seen. Though likely as uptake increases, so will efforts be taken by spammers to find attacks that cater to this different method.

Researchers have shown that most, if not all current conventional CAPTCHAs can be easily cracked with the right tools (Yan & Ahmad 2007, 2008b). Spammers are already finding ways to effectively overcome CAPTCHAs on even high profile sites owned by Microsoft (Keizer 2008, Yan & Ahmad 2008a, Philippe 2008, Prasad 2009), Google (Prasad 2008), Yahoo! (Broersma 2008), as well as a myriad of others that also use the same, or similar CAPTCHA methods.

3 VISUALLY ENCRYPTING CAPTCHAs

Visual encryption is a process developed by Naor and Shamir (Naor & Shamir 1995). The process takes an image and encrypts it into k images, all of the same size. All k images contain part of the information of the original image and only by having all k images can the original image be decrypted. Each part as a standalone image can't be decrypted in any way due to the fact that the original pixel of the image is randomly assigned to one of the k images, with the other $k-1$ images receiving a corresponding light or dark pixel depending on the colour of the original pixel. When the k images are overlaid, the corresponding pixels will emphasise the resulting images' pixels when they are dark and cancel out in others, resulting in a white pixel. A side effect of encrypting using Naor and Shamir's process is that

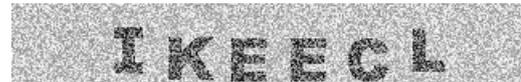
the quality of the image degrades, particularly in relation to pixel artifacts and contrast. This can however work to benefit the decryption process by further complicating matters for automated systems and CAPTCHA breaking software, while doing little to disrupt a human that visually decrypts the image. The noise inherent when the two images are overlaid means that pixel counting, where the characters in a CAPTCHA can be discerned simply by counting the number of similarly colored pixels is also avoided. This is because the random noise that is a side effect of encrypting the two images effectively changes each character's pixel count every time a CAPTCHA is encrypted.

The process of encrypting images into k pieces makes for very strong encryption, but lacks security when the k pieces are transmitted. Any party can intercept and decrypt the image, as long as all k images are found. Visual encryption on it's own is not very useful either. One simply needs to line up the k images and the encrypted image is revealed. In the method proposed in this paper, the use of two encrypted images makes it more resistant to decryption because it forces the system to accept a minimum processing overhead when decrypting the image. Another advantage is that the decrypted image is never stored on the computer as an image file at any time. Even when decrypted by the user, it is never stored or made easily accessible to automated systems. Other existing methods can be used to increase the quality of the encrypted image (Lin & Tsai 2003) though for this application it is not necessary. The visual encryption process can be applied immediately to any CAPTCHA implementation that uses an image file as the basis of their authorisation system. Due to the nature of the encryption, the total size of the images being sent is much smaller compared to the number of images that are created during decryption. This makes the transmission of larger, more difficult challenges possible while keeping usage of bandwidth low.

The real power of the encryption process is that a human can visually decode the image extremely fast. The human visual system can detect patterns and slight variations in pictures very quickly. This means that a human can immediately tell the difference between an image that contains meaningful patterns and an image filled with noise. This is coupled with the fact that decryption on the computer is easy to implement. The process does not require sophisticated encryption or decryption programs to encode or decode an image. All of this can be handled with browser scripting languages such as JavaScript. The encryption and decryption process is also very fast. This makes it much more viable for server systems that must generate and send hundreds or thousands of CAPTCHAs within a short space of time.

By harnessing the power of the human visual system and randomly hiding one of the images in a larger image filled with noise, we present a challenge to the human or computer. The human or computer must then find the original image within the larger image by overlaying the smaller encrypted image over the larger one. The human or computer must then traverse, pixel by pixel, row-wise and or column-wise, through every variation in the larger image to find the original encrypted image. For an automated computer system, it must decrypt and somehow capture the deciphered sub-image, run the image through a CAPTCHA breaker, then decide whether the image contains the CAPTCHA (if it decides incorrectly then it has to start with a completely new challenge), or disregard the image and move the overlay by one pixel to the next sub-image. Humans are essentially put on an even, or better level with a computer, since humans can detect even slight variations in an im-

Figure 2: An original phpbb CAPTCHA (a), an original Seed Peer CAPTCHA (b), and the same image when it has been visually decrypted (c)



(a) Normal phpbb CAPTCHA



(b) Normal Seed Peer CAPTCHA



(c) Visually Decrypted Seed Peer CAPTCHA

age much faster than a computer. It is very easy for a human to sift through a hundred sub-images in several seconds, find the variation as a sub-image switches to the CAPTCHA, refine the location using finer controls, process the characters within the deciphered CAPTCHA image, and enter the code.

3.1 The Downside of Visual Encryption and Decryption

Using a CAPTCHA with visual decryption carries with it a number of positives (as well as a few negatives). Negatives include the fact that a human operator must concentrate to properly find the CAPTCHA. Concentration on a seemingly trivial and time-wasting task may lead to boredom or frustration when trying to solve a visually encrypted CAPTCHA. Another negative aspect is the fact that decryption on the client side does involve a reasonable amount of CPU overhead. On desktop machines this is hardly noticeable but, netbooks, PDAs or mobile phones may consume most, if not all resources when a browser starts to rapidly decrypt the images. To solve this a server could do the processing and encode a video, or send all the images en masse, but this makes the approach less viable since an automated system has the server do a lot of the processing work for them. This results in more demand for bandwidth when transmitting images, and makes the sub-images that are produced much more accessible for automated systems. In the end, client side overhead is a necessary evil in many ways since an automated system is effectively forced to take on the overhead processing if they wish to find the CAPTCHA.

Another negative aspect of using visual decryption is that humans will find it difficult to navigate a large search space if it is done manually. Before testing, it was obvious that fine, pixel by pixel navigating could not be done by mouse or through keystrokes, and that realistic searching had to be fixed to one axis only. The most effective way to expose a human user to every image as quickly as possible was to automate the overlay such that all a user needed to do was start or stop the overlay moving. After the human had picked up a slight variation in the images being presented, then the user could use the mouse or keys to zero in on the location of the CAPTCHA. In the tests done the overlay was fixed on the x axis so it could only move up or down. The overlay itself was the same width as the large image. This helped avoid confusion and gave the user more control, at the cost of reducing the

search space that needed to be traversed. For testing purposes this was a fair trade off, but it is hoped that a much better user interface could be designed that can vastly improve the usability for users, expand the search space so that automated systems must process more images, and reduce the search space for humans.

3.2 Where is the CAPTCHA Image?

Encrypting CAPTCHAs using visual encryption means that the decrypted image is never stored as a file anywhere. The spammer must decrypt every sub-image and process each in order to find the CAPTCHA. Simply decrypting the sub-images results in processing overhead in itself that further slows down the processing of the CAPTCHA. The size of the encrypted images that are sent are also not representative of the images produced when the decrypted sub-images are created. A large 300x350pixel image combined with noise, and the smaller 300x50 pixel overlay image has a combined size in bytes a little over 11.5 KB as black and white png files. If all the processed sub-images were created and sent it would increase to nearly 400 KB. If this process were scaled up or used to substitute a traditional CAPTCHA implementation, and all sub-images were processed on the server side then sent to the client, the image size would significantly increase bandwidth usage, space and processing. Instead, sending over the large encrypted image and overlay results in a significantly smaller usage of bandwidth and very little processing. This makes the concept of using visual encryption much more appealing when it comes to implementation on a large scale. It also makes the encrypted contents a little more secure and forces CPU processing to be performed on the client side, which will further hinder automated systems.

3.3 Noise, artifacts, degradation of quality, not a bad thing after all

Another benefit of visual decryption is the fact that noise, as a result of encryption, is introduced into the image. This occurs because the encryption process requires random noise to be seeded into the original encrypted images so that neither image on it's own can be deciphered. Artifacts are not all bad in the case of CAPTCHAs. Artifact pixels further distort characters and actually make pixel counting attacks on these CAPTCHAs more difficult, as well as further distorting characters randomly each time to make optical character recognition software more difficult to use. The degradation of the image's quality may not be a bad thing since the random nature of the noise makes the decrypted image less predictable in it's structure and readability. Another positive aspect is that even if the same CAPTCHA were re-encrypted, the resulting noise and distortion will be completely different each time.

The use of visual encryption also makes the CAPTCHA much more resistant to man in the middle attacks where the CAPTCHA is stolen from a site and fed to another user on a spammer's site. These sites dupe unwitting users into validating CAPTCHAs for them. Since there is only one valid image among hundreds of sub-images, it means that conventional man in the middle attacks become more difficult to execute. The only way to continue to utilise man in the middle attacks, would be for the spammer to copy the interface and decryption implementation. This in itself complicates things for a malicious user that wishes to steal CAPTCHAs and use on their own sites because there are a number of ways in which an image can be visually encrypted. A spammer would have

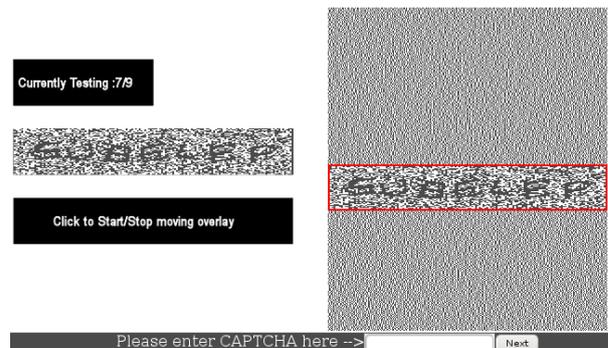


Figure 3: Testing screen interface

to copy the decryption process verbatim in order to properly decrypt the encrypted images.

4 METHOD AND TESTING

Two forms of testing were conducted to measure the usefulness of visual decryption for human users, and automated computer systems. The goal of human testing was to measure how easy, and how long a user would take to solve an encrypted CAPTCHA. The testing of automated computers was to get a general impression of how much longer the same tests performed by humans would delay an automated system. A browser client as shown in figure 3 was made to encrypt a set of nine CAPTCHA images using visual encryption methods outlined previously.

4.1 Visual decryption testing for human participants

The testing comprised three sets of speeds, 10fps, 15fps and 20 fps, and three different size categories, the heights of the size categories were, small (150px), medium (250px) and large (350px). The small image was 300x150 pixels and consisted of 100 sub-images being generated. The medium image was 300x250 pixels and consisted of 200 sub-images. The large image was 300x350px and subsequently generated 300 sub-images. The size of the CAPTCHA's encrypted image was 300x50 pixels and was not changed over any of the size categories. The CAPTCHAs were embedded at specific positions in each of the different size categories. In each size category, the CAPTCHAs were placed at positions of approximately one third to three quarters of the distance down from the top of the large encrypted image. In a real application the position of the CAPTCHA would be determined at random. Each test had the CAPTCHA placed in a slightly different position to avoid users predicting the position of the CAPTCHA, and given enough variance so that the user would not know where the CAPTCHA would be found. For each size category, the placement of the CAPTCHA was no more than 30 pixels distant from each other, this ensured that measuring the effect of using different frame rates did not noticeably distort comparisons between the time taken to find the CAPTCHA. The testing involved visually decrypting nine CAPTCHAs, and data on each test was collected.

The testing program, as shown in figure 3 measured the time it took from viewing the CAPTCHA, to entering the result into the program. It also recorded whether the CAPTCHA that was entered by the user was correct and what the CAPTCHA was if they entered it incorrectly. Already existing CAPTCHAs were taken from the site Seedpeer.com. CAPTCHAs that contained an I,J,D,Q,G or C were

avoided since image degradation while using visual encryption made the characters too ambiguous. The CAPTCHA images also had 5 pixels removed from the edges to improve the definition of the characters when it was encrypted.

The interface was designed initially so that the user could use the mouse, or up and down arrow keys to find the CAPTCHA. After consideration, the mouse was deactivated because it interfered too much with the overlay interface and had the potential to skew data that measured the overlay speed. This helped consolidate the reliability of speed measurements but subsequently put an emphasis on finding the CAPTCHA the first time, since the participant was no longer able to jump to different positions in the large image.

All participants that volunteered to test the interface were experienced computer users of various backgrounds, and familiar with CAPTCHAs and their purpose. Most were postgraduate, or undergraduate students. While testing, the participant was given time to familiarise themselves with the interface and then asked to visually decrypt each of the nine CAPTCHAs under varying speeds and size categories. The speed variation was linked directly to the animation of the overlay as it moved over the larger encrypted image. The participant's main method of detecting the CAPTCHA was through the use of the automated overlay. Once a variation in the images was found, the participant could stop the overlay and use the arrow keys to get finer control over the overlay position. The participant was told that the test was being timed, and was asked to try and finish the tests quickly.

A low end ASUS eeePC was used to conduct the testing. To ensure a fast enough environment, the net-book had to be over-clocked and the testing browser had to be given higher processing priority to ensure the animated overlay ran smoothly. On an Intel Q8400 PC with 3 gigabytes of RAM, the testing program used, at most, 4-7% of total CPU cycles. The overhead of decrypting the images on the fly is relatively low and would not generally impede normal activities on a modern PC, though this would not necessarily be the case for an automated system that attempted to process many CAPTCHAs at once.

A total of 14 participants took part in testing the visual encryption prototype. All participants' results were used as part of the final results, none were discarded except where all nine tasks weren't completed. The exception being one participant's timing for task one was incorrectly recorded so only task one's result was disregarded and the rest of task one's averages were recalculated based on only 13 participants. Mean and median values were calculated, the median values are preferred since it is less affected by large outliers in the data that was recorded.

4.2 Visual decryption testing for automated systems

To test the speed in which a CAPTCHA can be broken, a CAPTCHA breaking tool called "CAPTCHA Breaker" was downloaded from the Internet. The software is licensed as GPL-3.0 and written by Abram Hindle. It comes with the ability to break phpb, Pirate Bay, Seed Peer and Digg CAPTCHAs. Phpb CAPTCHAs were seen as the more convenient since it was the easiest to break, the CAPTCHA breaking was reasonably good, and could also be used to make conservative comparisons between automated and human testing. Using CAPTCHAs that were easy and quick to break meant that the results would be biased towards the CAPTCHA breaking software and not in favour of visual encryption. If visual encryption were

to have a significant impact on automated systems, then the impact would be from a conservative viewpoint.

Due to the nature of the testing program's browser implementation, the extraction of decrypted sub-images is a tedious and time consuming task. Most of the testing application, including encryption was implemented in JavaScript and extensively used JavaScript's new canvas object to perform the fine graphical implementations needed. As yet there is no easy method of extracting decrypted images from the browser unless specially customised code is written to automate print screen requests, or embedded JavaScript code is used. Coupled with this is the variant nature of CAPTCHAs and how CAPTCHA breaking software is usually tweaked according to the CAPTCHA methods used. To work around these time consuming issues, estimates were extrapolated from time measurements of the CAPTCHA breaking software successfully breaking CAPTCHAs. Linux's time command was used to measure how long a single CAPTCHA took to break.

A CAPTCHA was cracked several times and the processing time averaged. This was done on an Intel Q8400 PC with 3 gigabytes of RAM, while running Kubuntu 9.04 with minimal background processing and normal priority given to the CAPTCHA breaking software. The program took on average 0.153 seconds to successfully break a 320x50 pixel phpb CAPTCHA. The CAPTCHA is shown in figure 2a. The image was taken from Google's image indexing search results and was not tampered with. The original phpb image is 320x50 pixels, whereas the visually decrypted Seed Peer image shown in figure 2c is 300x50 pixels. At this time the visually decrypted image's CAPTCHA breaking time couldn't be measured because the software had not been tweaked to accommodate a visually decrypted CAPTCHA. Therefore it was decided the next best measure would be to use the original phpb image's average processing time as a benchmark since it was similar in size and also visually similar.

It is likely that malicious users will at some stage develop techniques to break a visually decrypted CAPTCHA. As such, estimates have been made to determine the processing time needed to process sub-images from the beginning of the large image (top), to the CAPTCHA locations that were used for testing with participants. The CAPTCHA position's approximate location was determined in the small, medium and large images used in tests with participants. This average determined the number of sub-images that needed to be decrypted and processed before the correct image could be found. Estimates were made of automated systems processing all images with the assumption that all sub-images had to be processed first before the system could decide which image was the decrypted CAPTCHA. Estimates were also made assuming that the automated system could detect the correct image immediately without having to decrypt and process all images, just like a human would. These assumptions and measurements allow a direct comparison with the results gathered from tests with human users.

5 ANALYSIS OF RESULTS

The initial implementation used during testing shows the practicality of the visually encrypted CAPTCHA. It demonstrates that the implementation using existing programming languages and hardware is viable. The implementation's search space was restricted to one axis, in this case the y axis, in order to help facilitate usability. This has the downside of also giving

an automated system cues that also limit the search space. On a 300 pixel by 350 pixel search space with a 300 by 50 pixel overlay, it means that a maximum of 300 sub-images must be individually processed in order to find the properly deciphered image. The increase in the number of images that must be processed, and the increase in time associated with it is a large improvement over existing CAPTCHA implementations. This can be made harder for computers in a number of ways by making the search space larger, encoding conventional CAPTCHA questions into the search space, or even seeding the encrypted large image with false positives in order to throw off automated systems. Another alternative is to span the search space over more axes in order to make the search space larger, but this also makes it more difficult for the human decrypter too. To take advantage of more than one axis, better methods need to be found to help humans disregard a large section of the search space, while making the computer still analyse the whole search space to find the answer.

Img Height /Speed	10fps	15fps	20fps
150px (100 sub-images)			
Correct Detection	100%	100%	100%
Mean Time	26.2s	21.1	17.4s
Median Time	24.3s	18.8s	15.9s
250px (200 sub-images)			
Correct Detection	92.8%	100%	100%
Mean Time	25.7s	20.2s	19.1s
Median Time	23.8s	19.3s	17.4s
350px (300 sub-images)			
Correct Detection	100%	100%	100%
Mean Time	34.0s	29.3s	27.4s
Median Time	33.5s	27.4s	27.0s

Table 1: Results of testing with 14 participants. Testing was conducted using small (150px), medium (250px), and large (350px) images at speeds of 10, 15, and 20fps. Times show the mean and median time taken to find and enter the CAPTCHA.

Total Sub-imgs	Time	Test Imgs	Time
100	15.3s	81/100	12.3s
200	30.6s	123/200	18.8s
300	45.9s	266/300	40.6s

Table 2: Estimates of processing 100, 200, and 300 320x50 pixel images using CAPTCHA breaking software. Test Imgs shows estimated processing if CAPTCHA were found in same approximate location as participants experienced during testing.

A comparison can be made between the average time taken by humans to decrypt and enter the CAPTCHA code, and an automated system that would need to run the CAPTCHA breaking software on all sub-images in the search space in order to find the image that yields the CAPTCHA. Table 2 shows that compared to humans in table 1, an automated system actually takes as long, or longer than a human to process 200 or 300 sub-images. It is highly likely that these processing times will be higher since the actual visually decrypted image couldn't be used to measure processing time due to the the processing software not being able to handle the visually decrypted CAPTCHA. The estimates also don't take into account the overhead of extracting every decoded sub-image from the large encrypted image before the CAPTCHA breaking software processes the image.

The first two columns in table 2 make the assumption that every sub-image will be processed by the CAPTCHA breaking software, though it is likely that an automated system would be tweaked to make conservative estimates about which sub-image is the right sub-image, (for example based on the number of characters detected) rather than process the entire set of sub-images.

The mean and median values in table 1 show that participants' time taken on each task were fairly consistent. The medians, which are more representative of the typical time taken for each task show that users solved the CAPTCHAs much faster as the overlay speed increased. Table 2 shows that when 100 images are being processed, compared to a human, the processing is faster on an automated system, but as the number of images increase, the human participants actually become the better performers. This is possibly due to the fact that a participant's timing of the task also took into consideration the time taken to type into the testing program the decrypted code. This minimum amount of time does not increase as more sub-images are needing to be processed. This highlights the fact that humans can disregard images faster than the automated system. A CAPTCHA broken in 0.153 seconds means the automated system could break 6.5 CAPTCHAs every second, whereas the number of images disregarded by participants were 10, 15, and 20 images per second. The correct detection row of each category in table 1 shows that nearly every CAPTCHA the participants entered was correct. Even at 20 frames per second, the participants could easily discriminate between a CAPTCHA and non-CAPTCHA.

After the participant had noticed a difference in image brightness, the overlay would be stopped. This almost always required the user to reverse the movement of the overlay using the arrow keys. In most cases the participant only needed to move the overlay back 5-8 pixels to re-find the properly decrypted image. It was observed during testing that participants found it was no trouble to find and read the decrypted images, even with distractions, or while talking. Though some commented that they would sometimes second guess themselves if the decrypted image did not appear after a long period of time. This sometimes happened when the image was positioned near the bottom of the 300x350 pixel large image, though this did not seem to impede their concentration.

5.1 Putting automated systems on the back foot

The results in table 1 and 2 lead to two informed conclusions, the first is that automated systems will be significantly impeded, from a processing and subsequently a time perspective. The impedance is not trivial, any slowing down of automated attacks through the use of visual decryption can have a significant effect on the amount of free resources automated systems get access to. If a single CAPTCHA's time to be cracked can be increased from 0.153 of a second to 15 or even 45 seconds as shown in table 2, this will slow down an automated system's efforts significantly. Since humans can easily filter through 20 images per second with no trouble, this means larger numbers of subimages will not necessarily impede humans as much as automated systems that can only process only 6.5 images on a fairly fast system.

This brings us to the second conclusion. The above impeding of automated systems is useless unless humans users can still get access via the visually decrypted CAPTCHA implementation. Table 1 shows that even when users are presented with the

same challenges as the CAPTCHA breaking software, the participants can on average detect the hidden CAPTCHA very quickly. Even as the speed of the animated overlay and number of sub-images increases, the participants actually find the CAPTCHA with very little chance of error. As the overlay speed increased, the participants were still able to correctly identify the decrypted CAPTCHA image, and because of the speed increase, in a shorter time. This is a strong indicator that most people's visual facilities will be able to spot a quick variation between a valid or invalid image while the overlay moves at 20fps, or perhaps even higher.

This process isn't without its drawbacks. The time taken to find and visually decrypt the CAPTCHA takes longer than conventional CAPTCHA implementations. It is hoped however that the interface could be further optimised to improve speed. As table 1 suggests, the percent of CAPTCHAs found at 20fps indicates that participants had no trouble finding the CAPTCHA while the overlay moved at 20fps. There is reason to believe therefore, that participants could have still found the CAPTCHA at a higher fps, hence reducing the time further.

5.2 Increasing the search space

The current implementation is a simple prototype and shows that users can use visually decrypted CAPTCHAs with little effort. It also shows that automated systems could be significantly slowed down due to the generation of extra sub-images that must be processed in order to find the original CAPTCHA image. To further capitalise on this, the number of sub-images can, in a number of ways, be increased substantially. In the prototype shown in this paper, the overlay is restricted to the y axis, but it is quite conceivable that the overlay could traverse along the x, y, and perhaps even the z axis. This is a double edged sword though, because even though the number of sub-images are now squared, the user will still have to search the entire search space to find the image. For example, if a 300 pixel by 300 pixel image that normally had 250 sub-images could be traversed on the x and y axis with a 50 pixel by 50 pixel overlay, the number of 50x50 sub-images would be 90,000. Restricting the number of axes searched along to only the y axis would only produce 300 sub-images consisting of 300x50 pixel overlays, so being able to exploit more than one axis while still making it easy for human users to use would be very desirable.

The key to making traversal along more than one axis viable is to limit the search space for a user, while still keeping the search space as large as possible for automated systems. Seeding the large encrypted image with false positives and embedding into the large image directions that humans can easily interpret could be ways in which the human operator can be guided, while still confounding the computer system. Cues could take the form of arrows, embedded textual, or verbal directions to guide the user along the x,y search space to the location of the encrypted CAPTCHA.

6 CONCLUSION

CAPTCHAs are becoming more of a hindrance than a help to users, yet they are the only way to stop the abuse of free resources on the Internet. CAPTCHAs are generally easy to break these days but there is no useful alternative to prevent automated systems from accessing free resources. This paper has introduced the concept of visual encryption as an extra layer of complication that can encrypt existing

CAPTCHAs. Visual encryption can impede conventional CAPTCHA breakers by hiding the real image inside a larger one which can be reliably found by humans yet also be hard to find for automated systems. Tests of this approach have found that humans can easily find and decode the image with near 100% accuracy. Human participants found and entered the CAPTCHA in 16-33 seconds, depending on the size of the image being searched and the speed at which the overlay moves to decrypt images for the user. Estimates of automated systems find that given the same position of the CAPTCHA that was used to test human participants, automated processing would take as long or longer than humans to decode and process. For images consisting of 100 sub-images the automated system performed better but as the number of images increased to 200 and 300, the automated system's processing time estimates took longer than a human doing the same tests on average. This demonstrates that visual encryption does not significantly impede human users and can significantly slow down conventional CAPTCHA breakers, even when the CAPTCHA is successfully found.

7 FUTURE WORK

The kinds of visual encryption implementations that can be used to make CAPTCHAs more secure has only really scratched the surface. The implementations of visual encryption and visual decryption could take many different forms and visual encryption is not just restricted to black and white implementations. Grey scale, as well as color visual encryption could be used. The testing conducted has still not found a ceiling for reliable detection of image variations as the animated overlay moves. The fastest at 20fps still resulted in the fastest detection times, to find out just how fast the overlay can move without participants missing the image would be useful to know. At this point we may be under utilising brain cycles that could instead be used for faster image detection.

There are also approaches that could be tried to reduce the search space for humans while expanding the search space for automated systems. This probably holds the most promise as far as impeding automated systems while at the same time making it more difficult for malicious users to program around. The introduction of false positives, expanding the search space along more than one axis, using human understandable cues, encrypting the CAPTCHA question into the large encrypted image itself, or perhaps even leveraging the latent visual encryption abilities to encrypt images in a variety of ways could be of great benefit.

References

- Broersma, M. (2008), 'Researcher claims to crack yahoo's antiscam filter'. http://www.pcworld.com/article/141545/researcher_claims_to_crack_yahoos_antiscam_filter.html.
- Gossweiler, R., Kamvar, M. & Baluja, S. (2009), 'What's up captcha?: a captcha based on image orientation'.
- Jeffrey, P. B. & Anna, C. C. (2009), 'Evaluating existing audio captchas and an interface optimized for non-visual use'.
- Jonathan, H., Jonathan, L., Jinjuan Heidi, F. & John, D. (2007), 'Developing usable captchas for blind users'.

- Keizer, G. (2008), 'Spammers' bot cracks microsoft's captcha'. http://www.computerworld.com/s/article/9061558/Spammers_bot_cracks_Microsoft_s_CAPTCHA_.
- Leyla, B., Thorsten, S., Davide, B. & Engin, K. (2009), 'All your contacts are belong to us: automated identity theft attacks on social networks'.
- Lin, C.-C. & Tsai, W.-H. (2003), 'Visual cryptography for gray-level images by dithering techniques', *Pattern Recogn. Lett.* **24**(1-3), 349–358. 641713.
- Naor, M. & Shamir, A. (1995), Visual cryptography, in 'Advances in Cryptology—Proceedings of Euro-crypto 94', pp. pp. 1–12.
- Ngo, D. (2009), '3d-based captchas become reality'. http://news.cnet.com/8301-17938_105-10204300-1.html.
- Philippe, G. (2008), 'Machine learning attacks against the asirra captcha'.
- Prasad, S. (2008), 'Google's captcha busted in recent spammer tactics'. <http://securitylabs.websense.com/content/Blogs/2919.aspx>.
- Prasad, S. (2009), 'Microsoft's captcha revolutions busted by spammers - again and again'. <http://securitylabs.websense.com/content/Blogs/3306.aspx>.
- von Ahn, L., Blum, M. & Langford, J. (2004), 'Telling humans and computers apart automatically', **47**(2), 56–60.
- Yan, J. & Ahmad, A. S. E. (2007), 'Breaking visual captchas with naive pattern recognition algorithms', *Computer Security Applications Conference, Annual* pp. pp. 279–291.
- Yan, J. & Ahmad, A. S. E. (2008a), 'A low-cost attack on a microsoft captcha'.
- Yan, J. & Ahmad, A. S. E. (2008b), 'Usability of captchas or usability issues in captcha design'.

Information security culture: A Behaviour Compliance Conceptual Framework

Salahuddin Alfawaz¹

Karen Nelson¹

Kavoos Mohannak²

¹ Faculty of Science and Technology, Queensland University of Technology
Brisbane, Australia

Email: s.alfawaz@isi.qut.edu.au

Email: kj.nelson@qut.edu.au

² School of Management, Queensland University of Technology
Brisbane, Australia

Email k.mohannak@qut.edu.au

Abstract

Understanding the complex dynamic and uncertain characteristics of organisational employees who perform authorised or unauthorised information security activities is deemed to be a very important and challenging task. This paper presents a conceptual framework for classifying and organising the characteristics of organisational subjects involved in these information security practices. Our framework expands the traditional Human Behaviour and the Social Environment perspectives used in social work by identifying how knowledge, skills and individual preferences work to influence individual and group practices with respect to information security management. The classification of concepts and characteristics in the framework arises from a review of recent literature and is underpinned by theoretical models that explain these concepts and characteristics. Further, based upon an exploratory study of three case organisations in Saudi Arabia involving extensive interviews with senior managers, department managers, IT managers, information security officers, and IT staff; this article describes observed information security practices and identifies several factors which appear to be particularly important in influencing information security behaviour. These factors include values associated with national and organisational culture and how they manifest in practice, and activities related to information security management.

Keywords: information security management, conceptual framework, information security culture, information security behaviour and compliance.

1 Introduction

Studies have shown that non-technical issues are at least as important as technical issues in safeguarding an organisation's sensitive information (Dhillon and Torkzadeh, 2006; Siponen and Oinas-Kukkonen, 2007). The importance of non-technical issues related to security management, however, has been emphasised in many studies by virtue of their quantitative nature (Siponen and Oinas-Kukkonen, 2007).

Copyright ©2010, Australian Computer Society, Inc. This paper appeared at the Australasian Information Security Conference (AISC), Brisbane, Australia. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 105, Colin Boyd and Willy Susilo, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

As a result, little attention has been paid to the role of human factors (e.g. individual choice and behaviour) or to organisational factors such as national and organisational culture, environment, and levels of information security awareness, and how these factors relate to attitudes about information security and its management. However, studies have shown that these factors are crucial to successfully safeguarding organisational information assets and that user input is imperative in addressing information security management strategies or issues (Vroom and von Solms, 2004).

There is general consensus that the purpose of information systems security is to ensure business continuity and minimise damage by preventing and /or minimising the business impact of security incidents. Dhillon et al. (2007) argue that "computer crime committed by internal employees is essentially a rational act" that may result from internal or external factors (e.g personal factors, work situation and available opportunities). These authors assert that behavioural security holds the key to successful information system security management (Dhillon et al., 2007).

Information security compromised by organisational insiders (employees and other stakeholders who have physical and/or logical access to organisational assets) can pose an enormous threat to an organisation's information systems. The risk posed to data by insiders needs to be closely monitored and managed. This risk can take two forms. The first form of risk is that posed by malicious insiders who deliberately leak sensitive data for personal financial gain or other criminal purposes. The second form of risk is from insiders who unintentionally expose data. Both these forms can result from carelessness or attempts to work around security measures. Information security management theorists assert that the behaviour of users needs to be directed and monitored to ensure compliance with security requirements (Vroom and von Solms, 2004; Dhillon et al., 2007; von Solms and von Solms, 2004). This view suggests that the success of an information security program depends on users' behaviour related to information security. Therefore, we contend that a better understanding of the characteristics of users' information security behaviours, will assist in assessing, improving and auditing individual information security behaviours, particularly in dynamic security environments.

Fishbein and Ajzen (1975) present the theory of reasoned action (TRA). TRA seeks to explain that an individual's behaviour or action is determined by his or her intention to perform such behaviour. Thus,

TRA considers that behaviour is determined by intention - which is in turn influenced by the individual's attitude towards performing that behaviour, and subjective norms (social pressures to perform the behaviour). The theory of reasoned action and its extension the Theory of Planned Behaviour (TPB) (Ajzen, 1985) have been applied in several studies relating to information security issues. More specifically, in risk perception, and security-related behaviour, both theories suggest that "ease of use" is an important factor affecting human behaviours. Siponen (2000) finds that the issues associated with "ease of use" of security solutions (e.g. techniques and adherence to procedures) has not been well addressed in the security literature. He suggests that a qualitative research approach would be appropriate to investigate this topic.

Earlier research has suggested several factors are crucial to information security policy adherence and user awareness. For example, Straub et al. (1993) applied the deterrence argument that information security actions will deter users from committing unauthorised acts. The deterrence argument has also been applied to improving the quality of information security policies (von Solms and von Solms, 2004), promoting security awareness (Straub and Nance, 1990), developing structures of responsibility (Dhillon et al., 2007) and protecting assets by motivation (Workman et al., 2008). Each of these studies provides important insights into specific issues relating to users adherence with security policies. To some extent, these studies all draw on the theory of reasoned action (TRA) and the theory of planned behaviour (TPB) (Fishbein and Ajzen, 1975; Ajzen, 1985), to understand and test constructs related to individuals' information security behaviours.

However, most of these studies, have paid little attention to the influence of national and /or organisational culture on employees attitudes, beliefs, and behaviours, or to the interactions between the individuals and their context. These interactions, may also contribute to an individual's beliefs and values about information security and its management.

2 The Analytic Framework

While there are some normative models for information security behaviour which are reported to work for one or two firms, there is little in the way of general guidance. The research reported here thus represents a preliminary attempt to identify a descriptive measure of information security related behaviours that are applicable for different types of organisations.

Classification theory suggests that classifying perceptions is crucial to human survival and adaptation, and attempts to explain the nature of concepts (categories/ classes) and why humans classify things (Smith and Medin, 1981; Parsons, 1996). Stanton et al. (2005) suggest that it is important to have a systematic view of end users security behaviour to facilitate accurate auditing and assessment of this behaviour. Therefore a classification that emphasises the characteristics of the organisational subjects who may perform authorised or unauthorised actions is proposed as helpful to understanding individual information security behaviour. Such a classification may serve two purposes for an organisation. Firstly, categorising a phenomenon makes systematic studies possible, and secondly, classification may assist organisations prioritise their information security efforts.

The term "knowing-doing gap" refers to people who have knowledge but do not take action or behaviour consistent with that knowledge (Pfeffer and Sutton, 2000). Workman et al. (2008) used this concept to investigate people's security behaviour referring to "people who have been trained but then do not use their new knowledge or skills as management expects". Following this analogy, we propose other possible patterns of an individual's behaviour with respect to information security practices. We choose to call these patterns modes (where mode means a "manner or way of acting, doing, or being; method or form") (Webster's New World Dictionary).

Based on an individual's acknowledgment of the security rules and possession of the essential skills for performing certain actions, we identify four modes to categorise individual security behaviours: Knowing-Doing mode, Knowing-Not doing mode, Not knowing-Doing mode and Not knowing-Not doing mode. Table 1 summarises the four modes. Figure 1 depicts these modes and their inter-relationships. The arrow lines connecting each mode represent the dynamic movement of each mode, which draws influences from individual's skills, knowledge and values based on change across the internal and external environment. Each mode is defined, theoretically justified and supported with relevant example/s as follows.

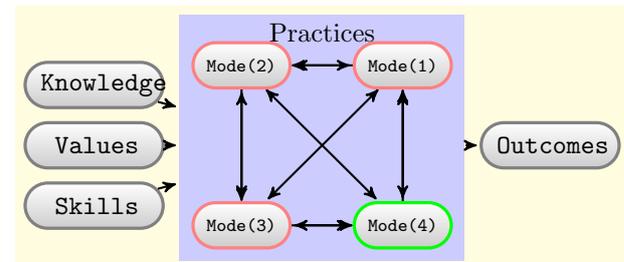


Figure 1: Information Security Behaviour Modes

Mode(1): Not Knowing-Not Doing In this mode, which falls into the upper right corner of the model of information security behaviour modes, the subject does Not Know the organisation's requirements for information security of behaviour and does not have security knowledge. As a result, they are Not Doing the right behaviour (violation of the security rules for behaviour - and security is compromised).

An example is a user who is not aware of the existence of organisational information security policies; he/she cannot be expected to follow them. Regardless of the presence of the necessary resources and the motivation to succeed, he/she can still fail to comply because they lack the knowledge of requirements/rules. An employee who has just joined the organisation or a manager who just been promoted to a new position may belong to this mode. This mode is a type of cognitive failure. Cognitive failures include issues such as: misunderstanding the security policy, missing an update of the policy, and poor decision-making.

Mode(2): Not Knowing-Doing: This second mode falls into the upper left corner of the model. The subject does Not Know the information security requirements/rules of behaviour and does not have security knowledge but is nevertheless Doing the right security behaviour (following the rules - security is not compromised).

Table 1: Information Security Behaviour Modes

Modes of individuals' behaviour	Description	Example of related information security behaviour
Mode(1): Not Knowing-Not Doing	In this mode the subject does Not Know the organisation's requirements for information security of behaviour and does not have security knowledge. As a result, they are Not Doing the right behaviour (violation of the security rules for behaviour - and security is compromised).	-Information security policy is not in place or is not properly communicated to the user: -sharing passwords -downloading internet software -visiting harm web contents.
Mode(2): Not Knowing-Doing	The subject does Not Know the information security requirements/rules of behaviour and does not have security knowledge but is nevertheless Doing the right security behaviour (following the rules - security is not compromised).	-Although there is no means provided to the users but they are voluntarily: -reporting valuations. -sharing related information and knowledge
Mode(3): Knowing-Not Doing	The the subject Knows the rules of behaviour and has the required knowledge and skills, but is Not Doing the right behaviour (violation of the rules of behaviour - security is compromised).	-Even though there was a policy at place and well communicated, users intentionally violating the related rules. -users using shortcuts to accomplish risky task. -users ignoring related procedures and rules.
Mode(4): Knowing-Doing	In this mode the subject Knows the rules of behaviour and has the knowledge/skills and they are Doing the right behaviour (following the rules - security is not compromised).	-Information security at place and well communicated and users are abiding by the rules.

A subject who is not aware of organisation information security policies, but asks supervisors or co-workers before taking certain actions, is an example of this mode. Some people may exercise more caution than others when they are uncertain how to act. This prudent behaviour demonstrates the conventional economic concept of being risk averse. The concept of risk being averse suggests that, when facing choices with the same outcomes, subjects tend to choose the less-risky one (Friedman and Savage, 1948). To some extent, this mode is also traceable to the self-regulatory model, which identifies rule-following as "originating within an individual's intrinsic desire to follow organisational rules" (Tyler et al., 2007).

Mode(3): Knowing-Not Doing: In this third mode, which takes the lower left corner of the model, the subject Knows the rules of behaviour and has the required knowledge and skills, but is Not Doing the right behaviour (violation of the rules of behaviour - security is compromised).

Given their knowledge, skills and sometimes authority over others, it seems reasonable to expect that employees will comply with the requirements/rules. However, this is sometimes not the case. An example of this mode is a person who has been trained but then does not use his new knowledge or skills as management expects (Workman et al., 2008) or a top manager or IT staff member who takes advantage of his position to compromise the rules (Dhillon, 2001). This mode suggests that while knowledge and skills are a key contributor to users behavioural output they are not the only ones. Theories of cognitive psychology explain why people may irrationally behave. One explanation is that a person's set of beliefs, or culture, may influence their actions. This suggests that if a person has a tendency to perform an authorised act and this tendency needs to be influenced, one has to focus on changing their primary belief system

(Dhillon, 2001). In this regard, Dhillon suggests that exposing employees to information about the consequences of their action may produce a change in their behaviour.

Mode(4): Knowing- Doing: In this mode, which takes the lower right corner of the model, the subject Knows the rules of behaviour and has the knowledge/skills and they are Doing the right behaviour (following the rules - security is not compromised).

This mode is based on the assumption that employees are rational actors who will comply with requirements because they have the necessary knowledge and skills. This mode is based on the view that people follow rules as a function of cost-benefit analyses (Stout et al., 2001). As in the case of mode 2, mode 4 is also linked to the self-regulatory model.

While mode 4 appears to be the "perfect mode" for management to target, there are at least two reasons why it is risky to rely on this mode alone. The first reason is that the information system security discipline is rapidly evolving as is the threat environment, and the required level of knowledge and skills. Yet, Mode 4 assumes that actors are able to keep their knowledge and skills current. This has always been a major challenging and costly task. The second reason is that it is not enough to secure the system by relying on those subjects who have the knowledge/skills and are Doing the right behaviours. Mode 4 requires the same level of planning, monitoring and managing as the previous modes. Furthermore an employee's behaviour may change from one mode to another, depending on their organisational role, the state of technology development, and the status and availability of security training.

3 Method

This article presents the findings from three exploratory case studies conducted in organisations in Saudi Arabia. Three methods were used to collect qualitative case data: semi-structured interviews, field notes and document analysis. For the purposes of this study, efforts were made to select diverse case organisations to allow for different of business, organisational size and approaches to information systems security management. The organisations selected for the study are Case A - a private organisation in the over 5000 employees category, Case B - represent participants from public organisations and Case C - a non-profit organisation which employs approximately 3,600 people, organisations in Saudi Arabia. The research was conducted in three phases over three years (January 2007-December 2009). This paper presents the preliminary result of this study. The first phase involved gathering data on the case organisations information security management approaches and practices to establish a baseline for later research. The sources of data for the interviews were senior managers, information security managers, functional managers, IT specialists, and IT users in each of the case study organisations.

In total, 13 interviews were conducted in case A, 16 in case B and 11 in case C with a further 7 interviews being carried out with Saudi PhD students who hold a related IT position in their work. Each interview lasted on average one and a half hours. The interview data was supplemented by a range of documentary evidence. This evidence was acquired from sources such as field notes, annual reports, organisational charts, official policy statements, and corporate Web sites.

4 Analysis and Findings

In all three cases, participants were asked to identify three main causes of security incidents as well as the obstacles to achieving improved information security compliance in their organisation. The interview data from the three cases revealed that behavioural issues associated with users' security compliance behaviour were the most common concern. These issues include password sharing, using shortcuts, downloading internet software, surfing potentially harmful content, ignoring relevant procedures, not sharing information and knowledge relevant to information security practices, not reporting security violations, and not enforcing security-related rules.

The first main cause of security incident was cited as users' errors and non-compliance. One IT manager pointed out that user error was the main cause of many of the information security incidents.

“all of the analyses we conducted on the various aspects of security incidents have identified careless and violation of policy rules as the main causes of accidents.”

The second cause identified in all three cases may arise from first and was identified as attacks from viruses and malicious software. In Cases A and C, the third factor identified was hardware failure, while in Case B the third factor was system administrator errors and non-compliance. This variation may reflect that both Cases A and C had issues relating to

budget constraints. In other words, these case organisations cannot afford to implement effective security mechanisms and procedures to protect themselves or they have other more important budgetary priorities. Another possible explanation is that both Cases were lacking information security staff or their current staff did not have the required level of skills. Whereas in Case B the issue seems to be more related to IT staff not following the right procedures and using shortcuts rather than lacking the required skills.

In terms of the obstacles to achieving improved security compliance, the cross-case analysis presented in Table 2 indicates that the participants in Case B and C saw the lack of clear direction in security procedures and roles as the major obstacle. In Case A, the lack of awareness and training programs was identified as the first obstacle, while the lack of clear direction in security procedures and roles came as a second. This is followed by the lack of motivation programs as the third obstacle in all three cases.

The variation between the cases appears to indicate existence and implementation of an organisation-wide information security policy in Case A. Whereas in both cases B and C information security procedures and rules were embedded in other organisational policies. Nevertheless, in Case A, participants identified “lack of awareness” as the second obstacle which indicates that that communicating the information security policy to the users is an issue of concern of Case A. Table 2 shows the main causes of security incidents and obstacles to achieving improved security compliance in the three cases.

Table 2: The main causes of security incidents and obstacles to achieving improved security compliance in the three cases

	The main causes of security incidents	The obstacles to achieving improved security compliance
Case A	1)The users' errors and non-compliance. 2)Viruses and malicious software. 3)The hardware failure.	1)Lack of awareness and training programs. 2)Lack of clear direction in security procedures and roles. 3)The lack of motivation programs.
Case B	1)The users' errors and non-compliance. 2)Viruses and malicious software. 3)The system administrator's errors or non-compliance.	1)Lack of clear direction in security procedures and roles. 2)Lack of awareness and training programs. 3)The lack of motivation programs.
Case C	1)The users' errors and non-compliance. 2)Viruses and malicious software. 3)The hardware failure.	1)Lack of clear direction in security procedures and roles. 2)Lack of awareness and training programs. 3)The lack of motivation programs.

In order to build in-depth inferences from the case studies, further data analysis was conducted to visualise and identify patterns and relationships between individuals' information security related behaviours. The aim was to determine whether or not the conceptual model (illustrated in figure 1) and four modes comprehensively describe individuals' information security behaviours that occur in the course of conducting their daily work.

The results presented in Table 3 seem to suggest the plausibility of the four modes, for Cases A, B,

and C. While there were similarities in terms of all four modes of information security behaviour being present in the three cases, variations were found in the behaviours related to each of the modes. Based on our findings from three case studies we placed each case on a grid chart (see Figure 2), as red, green and blue circles representing Case A, Case B and Case C respectively. The case study findings are reported below through an exploration of the framework's four modes as follows.

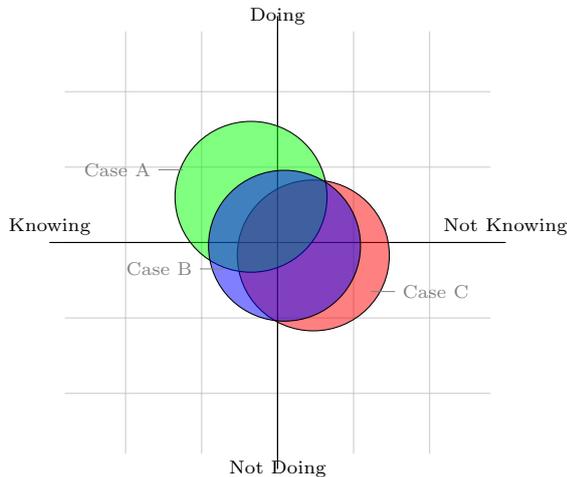


Figure 2: Information Security Behaviour Modes at the three cases

Mode(1): Not Knowing-Not Doing mode:

In Cases B and C, individuals were not aware of their organisations' information security policies; hence, they could not be expected to act to follow them. As noted earlier, regardless of having the necessary resources and the motivation to do so, if an individual lacks knowledge of the requirements or rules he/she may not exhibit appropriate information security behaviours. This is a type of cognitive failure that also includes issues such as misunderstanding the security policy or missing an update of the policy. In Cases B and C there was no evidence to show that unified and / or clearly articulated information security policies had been communicated to users. The lack of understanding about policy appeared to be the main contributor to most of the non-compliance issues reported by Case B and C.

For instance, respondents from both Cases B and C raised the importance of organisational policies to the development of information security (e.g. policy that seeks to standardize managerial procedure). It also appeared that the lack of clarity about what kind of procedures needed to be followed and enforced contributed to the lack of information security compliance in Case Organisation C.

One IT staff member explained:

“mostly individuals are taken for granted to do the right thing [following information security rules and procedures] but, unfortunately, individuals in most cases doing the wrong things”

Managers from different departments also supported IT staff views that the absence of clear information

security procedures and directions had contributed considerably to information security system incidents. All the interviewees in Case C, indicated that they were not familiar with the information security policy, although the IT manager made the following observation:

“It [the information security related procedures] has provided us with some guidance in different cases, initially. Now whether all departments would have followed and enforced them is another question”

Mode(2): Not Knowing-Doing mode:

The data collected from the case the interviews showed that most of the participants in three Cases were risk averse which although they do not know predisposes them to act conservatively. This aversion was mainly attributed to the belief that taking risks could affect their organisation's information assets. In Case A, and to a some extent, in Case B, a combination of self-consciousness as a member of the organisation and a willingness to abide by the organisation's rules indicated two aspects of both Case's organisational cultures. The first aspect was a sensitivity to losing information, knowing that they will be questioned about. The other was the hope for a reward, through the KPIs systems, as well as group bonus schemes, which were linked to organisational performance in the case of case A.

In a similar vein, all participants pointed out that cultural values can influence employees' information security related behaviours. For example one participant noted:

“certain cultural values could make people do the right thing but other values may not”

One can infer from the last part of this statement that certain individual cultural values may have a positive or negative influence on employees' security behaviour. Most of the case data appears to support this claim, for instance respondents indicated that there is a cultural influence on individuals' security-related behaviour which poses challenges, although managers may overcome these challenges by extended exposure to managerial activities such as training and/or awareness. However, some respondents did not see all personal cultural values as having a negative influence, especially in the context of individual security related behaviour. One manager summarised his thoughts:

“There are different components of the personal culture. Some of these values are good in promoting good behaviour”

He went on to illustrate his point using an example showing that some cultural values are useful in positively influencing individual security behaviour:

“in some cases religion values dictate where one is going. These religion values may hold one from visiting prohibited sites which usually have some viruses or spywares that could cause security related problems.”

Table 3: Modes of individuals' behaviour of information security culture in the three Cases

Modes	Case A	Case B	Case C
Mode(1)Not knowing-Not doing	Some IT staff were not sharing related information and knowledge because they were not aware of the right mechanism.	Most of employees were not aware of the information security policy and there was no clear instructions provided for them by IT department.	Most of the employees were not aware of the information security policy and there was no clear instructions provided for them by IT department. Individuals' non-compliance behaviour was seen as a result of the lack of existence and clarity of related rules and consequences of taking information security risks.
Mode(2) Not knowing-Doing	Voluntary sharing culture of information and knowledge related to information security between IT staffs.	As in public organisation, employees rely on the managers to solve work issues, most of non-compliance behaviour were prevented. Some national culture values prevented users from visiting illegal web contents and sharing between technical staffs takes informal approach.	Sharing information and knowledge between technical staffs takes informal approach and some culture values dictated users actions.
Mode(3)Knowing-Not doing	Although users aware of the information security procedures, some users intentionally conducting a non-compliance behaviours, example, using shortcuts, downloading internet software.	Employees were ignoring related procedures by downloading internet software, and some employees may have a tendency to not report colleagues' violations for the sake of saving the group's image.	Users were using shortcuts, downloading internet software, and some function managers may have a tendency to not enforce the rules to discipline their subordinates for a sympathy or protection concerns.
Mode(4)Knowing-Doing	The level of information security culture indicted that majority of members in all cases fit in this mode.		

This data indicates that some cultural values may impact on an individuals security-related behaviour and ultimately influence information security culture in a positive way.

This last point can be further examined by understanding aspects of the relationship between managers and employees. As is common for national cultures that score high on Hofstede's (1984) Power Distance dimension such as Saudi Arabia, executives and managers at upper-levels are sought out for advice and guidance (Hofstede, 1984). In a high Power Distance culture, employees usually rely on managers to solve work issues, because managers often attain the role of problem-solver. The impact of the Power Distance dimension is reflected in some of the information security managers' comments:

"When they [employees] face problem they come to me and I do my best.."

"We direct them [employees]."

In these cultures IT staff report issues to the IT Manager, who provides guidance and directives, which are then actioned by the IT department staff. As one IT staff member commented:

"We follow the organisation's procedures by getting the decision from high management."

These comments suggest that people may lack the experience to resolve problems since managers deal

with issues in the absence of explicit procedures. Under these conditions, undesirable employee information security behaviour and actions may be minimised as most activities have to be approved by immediate managers or work supervisors.

Although negatively affected by the lack of sharing and motivation mechanisms, some employees have adopted informal means for sharing information and knowledge related to information security systems. Members of Case A, for example, meet after work and the conversation usually turns to something that happened during their work hours. Whenever the group are together they discuss issues and problems encounter in their daily work. As one participant explained:

"Yes, we discuss some [ISM]issues in our lunch brakes or at the informal meetings. It is a good opportunity to ask for opinions or share some experience with colleagues.... Not only with IT people but with others as well, such as HR people.."

Mode(3):Knowing-Not Doing mode:

The data revealed that there was careless risk taking by individuals who used shortcuts, downloaded internet software, and surfed harmful internet content. These practices, as noted, varied between the three cases. In Cases B and C these behaviours can be mostly attributed, to the lack of and poor clarity about the rules and consequences of taking information security risks. Whereas in Case A, the data indicated intentional incidents relating to non-compliant behaviour. For example, Case A's Intranet

sites are updated regularly with security information, and employees are encouraged to access these sites on a regular basis. However, there was a perception that many of the organisation's members did not take these routine information security awareness programs seriously. One participant commented on security warning e-mails:

“the IT department sends a lot of warning e-mails related to security issues...almost every day...but I'm sure not every one takes them seriously.”

Another participant admitted that:

“Because some people do not have enough time they delete warning e-mails without even bothering to look at them...”

Mode(4):Knowing-Doing mode:

The level of information security culture in all three cases indicated that majority of information security related behaviours fit into this mode. Data showed that members in all three cases believed that the organisation's dependence on information systems is “very high and security is an integral part of this equation”. Most participants indicated that there was a certain level of comfort with the progress that their IT department was making in information security related areas. For example, in all cases, the data showed that top management commitment to information security was exemplified by allocating the necessary resources and adopting technical solutions to enhance information security programs.

The influence of national culture traits (for example, Hofstede's Power Distance dimension) may be seen in the practices associated with this mode. Saudi Arabia is a high Power Distance society, and data from all three cases indicated that individuals intended to follow the expectations of management and they are more likely to approve actions that they perceive to be supported by functional managers and work supervisors. These traits appear to be having a substantial influence on individuals' information security related behaviour in all three case studies.

Furthermore, the data indicated that a combination of self-consciousness as a member of the organisation and a willingness to abide by the organisation's rules was present in the organisational culture of the three cases. The sensitivity of losing information, knowing that they will be questioned about and the hope for rewards for reporting security incidents were also key factors in individuals' compliance with information security requirements.

However, as previously discussed, we should expect organisations' actors to keep their knowledge and skills current and it is not enough to secure the system by addressing the concern of those who have the knowledge/skills to do the right things alone. Organisations are going through a rapid and costly change as they seek to adjust and perform in the changing environment (e.g. new regulations, new technology and new threats). Therefore, mode 4 requires the same level of planning, monitoring and managing as the previous modes. An employee's behaviour may alter from one mode to another, depending on the organisational role the subject happens to be in, the state of technology deployment, and the relevance and availability of the suitable training.

5 Discussion and Next Steps

The findings supported the proposed model of the four modes of information security behaviour. A number of factors appeared to be interrelated. These inter-related factors included organisational cultural values manifest in practices and activities related to information security management, and factors related to the national culture, particularly the influence of power-distance on individual. The most important factors identified in this study were top management commitment, the level of training and IT skills, security awareness programs, organisational IT structures, the appointment of information security managers, type of motivation system utilised, existence of information security policy, and adoption of information security standards. Other factors were related to the influence of national culture on values in decision making, compliance, risk taking, sharing culture, collaboration, enforcement, reporting, and communication. Hence, these findings are consistent with the view that an individual actor's decision to comply with security requirements is not only a function of their knowledge and skills or the perceived cost-benefit of the behaviour as described in economic theories, but also, a function of the factors arising from the users' psychology and the social setting in which the actor is situated. Therefore it is crucial to understand how aspects of organisational and national culture inform employees' practices in order to achieve a high level of information security culture.

The complexity inherent in contemporary organisations suggests that organisations will have individuals who do not share a view of information security, and yet are expected to participate in the information security culture of that organisation. These disparate views may be attributed to the different assumptions, attitudes and values towards the information system implementation and use processes held by each of employee. Variation may also be related to, rapid technological advances bringing about an increase in the range of tools used for conducting unauthorised behaviours. Another noteworthy point, is that most employees assume the security of their organisation is not their responsibility and that only IT staff are responsible. Therefore, it is important to understand, what underlying principle values, beliefs and assumptions drive users behaviour. This is further complicated by the rate of change in the information systems environment with respect to security threats, which makes it unwise to assume that individual knowledge/skills will be current and that individual behaviour will remain as expected.

The challenge is now to determine the parts of an organisation's environment that facilitate and enable sustainable approaches to information security adherence. This is a complex issue with no easy answers. One aspect emphasised in the literature is the notion of creating a security culture, which is emerging as a goal for governments and corporations in their attempts to safeguard their information assets. We contend that a culture that encourages ethical conduct and commitment to compliance with information security requirements is a desirable organisational attribute. Many researchers have addressed the importance and the need for an information security culture in organisations (Chia et al., 2002; Ruighaver et al., 2007; Schlienger and Teufel, 2002, 2003; Zakaria and Gani, 2003; Zakaria, 2004). They all suggest that organisations must take affirmative steps to create an environment where security is “everyone's responsibility” and doing the right thing is the norm.

These observations provide a basis for us to propose “the information security culture mode”. In this mode organisations would work towards developing an information security culture where all employees adhere to its information security policy and rules even when no one is around and when their behaviour is not being monitored. Practices in mode 5 would also include cooperative information security, such as taking action against acts that would jeopardise the information security system for example, reporting unauthorised acts, and sharing security-related information and knowledge through the appropriate formal and informal channels.

In order to achieve the mode of information security culture, two things will need to occur. Firstly, the environmental factors that influence behaviour and encourage or inhibit individual employees and managers from doing the right thing, even when they know what the policy says, should be identified. Secondly, an effective management strategy that handles both internal and external factors critical to information security should be implemented.

This paper provides some insights but, clearly, additional investigations are required. Hence, we propose that a multi-methodological approach will be required to capture the richness of the information security management systems (ISMS) implementation processes in developing countries and the influence of both organisational and national culture values on information security culture development. More specifically, this second phase will explore information security management related activities within organisations in the Saudi Arabia context and how individual manager and employee personal values may affect the transition towards an information security culture. The study will use an integrated framework that incorporating information security culture into existing cultural models. Further, the study will adopt change management as an effective management strategy that manages both internal and external changes.

6 Conclusion

Based on evidence from three exploratory case studies, we populated a framework of information security practices that could contribute to information security management by identifying behaviours related to four modes of information security practice. The aim was to classify individual information security behaviours in organisations to ensure the development of high quality information security cultures. The information security modes described in this paper provide a sound basis that can be used to evaluate individual organisational members’ behaviour and the adequateness of existing security measures.

Although this approach does not deliver completely new measures, it leads to a more consistent set of security parameters which aim to protect against individuals non-compliant behaviour. The main strength of our approach is that it takes into account the complexity of human behaviour and their corresponding actions.

We conclude this paper with three remarks. First, although individual knowledge and skills are important, they alone are not enough to assure a positive contribution towards information security culture reliant on employee behaviours. Second, a person’s set of beliefs, or personal culture, plays a major role in

influencing their personal attitude towards their security behaviour. Hence, understanding their underlying beliefs is crucial in the process of behavioural change. Third, the influence of technology, social environment, regulation and self-interest all contribute to employees security-related behaviours. As a result members of an organisation will could exhibit behaviours from different modes at different points in time. This continuous movement makes it hard to secure an organisation’s information system by addressing a single mode in isolation. Hence, future research efforts should concentrate on investigations of these factors. The research findings and the model described in this paper may serve as resources for further investigating the human, (organisational and individual) aspects of effective information security systems.

References

- Ajzen, I. (1985). *From Intentions to Actions: A Theory of Planned Behavior*. Springer, Heidelberg, Germany.
- Chia, A., Ruighaver, B., and Maynard, B. (2002). Understanding organizational security culture. *Proc. of PACIS2002, Japan*.
- Dhillon, G. (2001). Violation of safeguards by trusted personnel and understanding related information security concerns. *Computers & Security*, 20:165–172.
- Dhillon, G., Tejay, G., and Hong, W. (2007). Identifying governance dimensions to evaluate information systems security in organizations. *Proceedings of the 40th Annual Hawaii International Conference on System Sciences (HICSS’07), computer security, IEEE*.
- Dhillon, G. and Torkzadeh (2006). Value-focused assessment of information system security in organizations. *Information Systems Journal*, 16:293–314.
- Fishbein, M. and Ajzen, I. (1975). *Belief, attitude, intention and behavior: an introduction to theory and research*. Addison-Wesley, Reading, MA.
- Friedman, M. and Savage, L. (1948). The utility analysis of choices involving risk. *Journal of Political Economy*, 56:279–304.
- Hofstede, G. (1984). *Cultures consequences: International differences in work-related values*. Beverly Hills, CA: Sage Publications.
- Parsons, J. (1996). An information model based on classification theory. *Management Science*, 42(10):1437–1453.
- Pfeffer and Sutton (2000). *How Smart Companies Turn Knowledge into Action*. Harvard Business Press.
- Ruighaver, A., Maynard, S., and Chang, S. (2007). Organisational security culture: Extending the end-user perspective. *Computers & Security*, 26(1):56–62.
- Schlienger, T. and Teufel, S. (2002.). Information security culture the socio-cultural dimension in information security management. *FIP TC11 international conference on information security, Cairo, Egypt; 7-9 May 2002*.

- Schlienger, T. and Teufel, S. (2003). Analyzing information security culture: increased trust by an appropriate information security culture. *Database and Expert Systems Applications, 2003. Proceedings. 14th International Workshop on*, pages 405–409.
- Siponen, M. (2000). Critical analysis of different approaches to minimizing user-related faults in information systems security: implications for research and practice. *Information Management & Computer Security*, 8(5):197–209.
- Siponen, M. and Oinas-Kukkonen, H. (2007). A review of information security issues and respective research contributions. *SIGMIS Database*, 38(1):60–80.
- Smith, E. and Medin, D. (1981). *Categorization and Concepts*. Harvard University Press, Cambridge, MA.
- Stanton, M., Stam, R., Mastrangelo, P., and Jolton, J. (2005). Analysis of end user security behaviors. *Journal of Computers and Security*, 24:124–133.
- Stout, L., Blair, and Margaret, M. (2001). Trust, trustworthiness, and the behavioral foundations of corporate law. *SSRN eLibrary*.
- Straub, D. W., Carlson, P. J., and Jones, E. H. (1993). Deterring cheating by student programmers: A field experiment in computer security. *Journal of Management Systems*, 5:33–48.
- Straub, D. W. and Nance, W. D. (1990). Discovering and disciplining computer abuse in organizations: A field study. *MIS Quarterly*, 14:45–62.
- Tyler, T., Patrick, C., and Jeffrey, F. (2007). Armed, and dangerous (?): Motivating rule adherence among agents of social control. *Law & Society Review*.
- von Solms, B. and von Solms, R. (2004). The 10 deadly sins of information security management. computers and security. *Computers and Security*, 23:371–376.
- Vroom, C. and von Solms, R. (2004). Towards information security behavioral compliance. *Computers & Security*, 23(3):191–198.
- Workman, M., Bommer, W. H., and Straub, D. (2008). Security lapses and the omission of information security measures: A threat control model and empirical test. *Computers in Human Behavior*, 24:2799–2816.
- Zakaria (2004). Understanding challenges of information security culture: a methodological issue. In *the second Australian information security management conference, Perth, Australia; 26 November 2004*.
- Zakaria, O. and Gani, A. (2003). A conceptual checklist of information security culture. In *in 2nd European Conference on Information Warfare and Security, Reading, UK*.

Multi-Factor Password-Authenticated Key Exchange

Douglas Stebila¹Poornaprajna Udupi²Sheueling Chang³

¹ Information Security Institute
Queensland University of Technology
Brisbane, Queensland, Australia
Email: douglas@stebila.ca

² Sun Microsystems Laboratories
Santa Clara, California, United States
Email: poornaprajna.udupi@sun.com

³ Email: sheueling.shantz@gmail.com

Abstract

We consider a new form of authenticated key exchange which we call *multi-factor password-authenticated key exchange*, where session establishment depends on successful authentication of multiple short secrets that are complementary in nature, such as a long-term password and a one-time response, allowing the client and server to be mutually assured of each other's identity without directly disclosing private information to the other party.

Multi-factor authentication can provide an enhanced level of assurance in higher-security scenarios such as online banking, virtual private network access, and physical access because a multi-factor protocol is designed to remain secure even if all but one of the factors has been compromised.

We introduce a security model for multi-factor password-authenticated key exchange protocols, propose an efficient and secure protocol called MFPAK, and provide a security argument to show that our protocol is secure in this model. Our security model is an extension of the Bellare-Pointcheval-Rogaway security model for password-authenticated key exchange and accommodates an arbitrary number of symmetric and asymmetric authentication factors.

Keywords: multi-factor authentication, passwords, key exchange, cryptographic protocols

1 Introduction

Phishing and spyware are two of the major security problems on the Internet today. *Phishing*, or server impersonation, occurs when a malicious server convinces a user to reveal sensitive personal information, such as a username and password, to a malicious server instead of the real server. Additionally, many users' computers are compromised with *spyware*, which can record users' keystrokes (and thus passwords) and transmit this information to a malicious party. These attacks are possible not because of the break of any cryptographic protocol but because of externalities such as social engineering and software bugs.

In theory, these attacks can be addressed in part by using trusted cryptographic devices that can store private keys and perform cryptographic operations, but such devices are difficult to deploy and use. Years

of experience have shown that passwords are a much more popular and easy-to-use form of authentication, but are more susceptible to phishing and spyware attacks. In this work, we focus on the use of passwords for authentication, since they are easier for users to use and carry between computers than long private keys.

Phishing can be combated by protocols that provide strong, easy-to-use server-to-client authentication. Password-authenticated key exchange (PAKE) can make server-to-client authentication easier and resistant to offline dictionary attacks, and additionally provides a secure key for encryption.

Spyware is more difficult to defend against. If a user's computer is compromised by passive spyware that records keystrokes and occasionally transmits this information to an attacker's server, then the use of *one-time passwords* may be effective, since a previously used one-time password can not be used again. Active spyware – that frequently communicates with the attacker's server and actively alters the user's computer – is nearly impossible to defend against without additional trusted hardware.

To reduce the damage caused by compromising an authentication factor, many organizations with high security requirements – such as financial institutions, governments, and corporate virtual private networks (VPNs) – are deploying *multi-factor authentication*, which depends on a variety of attributes, or *factors*. The factors could include: a long-term password, a set of one-time passwords, a private key, or a biometric. To be effective in practice, factors should have different, complementary natures of compromise. For example, one-time passwords cannot all be compromised unless one obtains the sheet of paper listing all the one-time passwords or the device generating the one-time passwords, whereas a biometric read by a trusted device (such as a secure fingerprint reader) should not be able to be reproduced without the presence of the person in question (or at least their finger).

Contributions. Our goal is to design a framework for multi-factor authentication protocols that provides flexibility in the number and nature of factors. Protocols secure in this framework should provide strong mutual authentication, convey the authentication secrets in a secure manner, and remain secure even if all but one of the authentication factors is compromised. The authentication secrets can be low-entropy secrets, such as passwords. Using multiple low-entropy secrets can allow for passwords that may have different modes of compromise, such as a memorized long-term password and a one-time password generated from a hardware device or transmitted over a mobile phone text message.

First, we define a security model which is an exten-

Copyright ©2010, Australian Computer Society, Inc. This paper appeared at the Australasian Information Security Conference (AISC), Brisbane, Australia. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 105, Colin Boyd and Willy Susilo, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

sion of the Bellare-Pointcheval-Rogaway model (Bellare et al. 2000) for PAKE. Our model allows for an arbitrary number of authentication factors, which can be either symmetric or asymmetric. Our security definition formalizes the notion that a multi-factor protocol should remain secure even if all but one of the factors has been compromised.

Next, we present an efficient multi-factor protocol that is secure in this model under standard cryptographic assumptions in the random oracle model. Our protocol combines facets of the PAK protocol (MacKenzie 2002) for symmetric factors and the PAK-Z+ protocol (Gentry et al. 2005) for asymmetric factors. We discuss how many different types of factors – long-term passwords, one-time passwords, biometrics, and even private keys – can be used in our protocol.

Our work differs from previous work in PAKE because it uses multiple authentication factors and maintains security even if some are compromised. Others have considered some aspects of multi-factor authentication, but these have either used at least one factor that is a long cryptographic secret (Yang et al. 2006, Park & Park 2004, Yoon & Yoo 2006, Pointcheval & Zimmer 2008), or have not provided strong server-to-client authentication resistant to man-in-the-middle attacks.

Outline. The rest of our paper proceeds as follows. In Section 2, we describe the security model for multi-factor PAKE. In Section 3, we present our protocol MFPAK and discuss its efficiency; we show through a formal analysis that the MFPAK protocol is secure and discuss how various types of factors can be used. Section 4 concludes the paper with what we believe are interesting directions for future research. Appendix A presents the one of the cases for our security proof for the MFPAK protocol; the rest appear in the full version of the paper (Stebila et al. 2009).

1.1 Related work

Password-authenticated key exchange was first introduced by Bellare and Merritt in 1992 (Bellare & Merritt 1992) as the encrypted key exchange (EKE) protocol, in which the client and server shared the plaintext password and exchanged encrypted information to allow them to derive a shared session key. A later variant by Bellare and Merritt, Augmented EKE (A-EKE) (Bellare & Merritt 1993), removed the requirement that the server have the plaintext password, instead having a (non-trivial) one-way transformation of the password, which alone is not sufficient to impersonate the user. The former is called a *symmetric* password-based protocol, because both client and server share the same plaintext password (or a trivial transformation of it), whereas the latter is called *asymmetric*. The dominant model for the security of PAKE protocols was proposed by Bellare, Pointcheval, and Rogaway (Bellare et al. 2000) and extended by Gentry, MacKenzie, and Ramzan (Gentry et al. 2005) to accommodate asymmetric protocols.

Many PAKE protocols have been developed, including PAK (Boyko et al. 2000, MacKenzie 2002) and PAK-Z+ (Gentry et al. 2005) which are relevant to our construction. Although universally composable constructions are attractive to consider when combining primitives, the existing work on universally composable PAKE (Canetti et al. 2005) is only symmetric, not asymmetric, and thus unsuitable for our approach.

A number of two-factor authentication schemes have been proposed that rely on a short password and a long cryptographic secret (Park & Park 2004, Yang et al. 2006, Yoon & Yoo 2006). Pointcheval and

Zimmer (Pointcheval & Zimmer 2008) presented a multi-factor authentication scheme using a password, a long cryptographic secret, and biometric data; their scheme has a formal security argument in a variant of the BPR model that shares some features with ours.

There are also non-cryptographic approaches to multi-factor authentication, but these do not provide as strong protection for the authentication factors. In a *multi-channel* system, the second factor is delivered over a separate channel (for example, via an SMS text message on a mobile phone), which the user then inputs into their web browser along side their password. In a *multi-layer* system, software installed on the server evaluates additional attributes such as an HTTP cookie, IP address, and browser identification string to heuristically analyze whether the user is likely to be authentic. Some multi-layer systems try to offer additional reassurance to the user of the server's identity by presenting the user with a customized image or string. While these multi-channel and multi-layer approaches can offer some increased assurance, they can be defeated by non-cryptographic means such as sophisticated man-in-the-middle attacks and spyware, and have been shown to be easily ignored by users (Schechter et al. 2007).

2 Security for multi-factor protocols

In a multi-factor PAKE protocol, multiple authentication secrets of complementary natures, such as a long-term password and a one-time password, are used. We support two general types of authentication factors: symmetric and asymmetric.

The authentication secrets must be used in a way that the client can convince the server that it knows all the authentication secrets, and that the server can convince the client that it knows all the authentication secrets: this provides mutual authentication. However, the protocol must be carefully designed to not reveal any information about the authentication secrets to a passive or even active adversary.

Secure communications often involve both authentication and encryption so, in addition to providing authentication, we want protocols that establish an ephemeral shared secret key between client and server that can be used, for example, for bulk encryption.

Informal security criteria. The general security criteria we use for multi-factor PAKE is that the protocol should remain secure even if all but one authentication factor is known to an adversary. We identify four security properties such a protocol should have:

1. Strong multi-factor server-to-client authentication: without knowledge of all of the authentication factors, a server cannot successfully convince a client of its identity.
2. Strong multi-factor client-to-server authentication: without knowledge of all of the authentication factors, a client cannot successfully convince a server of its identity.
3. Authentication secrets protected: no useful information about the authentication secrets is revealed during the authentication process.
4. Secure session key establishment: at the end of the protocol, an honest client and an honest server end up with a secure shared session key suitable for bulk encryption if and only if the mutual authentication is successful; otherwise no session is established.

2.1 Security model

We define a model for the security of multi-factor PAKE that allows one to argue that a protocol is secure by giving upper bounds on the probability that an adversary can break server-to-client or client-to-server authentication, or determine the session key

established; the authentication secrets are protected from offline dictionary attacks as well.

This model is an extension of the model for PAKE proposed by Bellare, Pointcheval, and Rogaway (Bellare et al. 2000) and modified by Gentry, MacKenzie, and Ramzan (Gentry et al. 2005). The model allows for an arbitrary number of authentication factors, and each factor can be either symmetric or asymmetric.

Participants. In this model, each interacting party is either a *client* or a *server*, is identified by a unique fixed length string, and the identifier is a member of either the set *Clients* or *Servers*, respectively, with $\text{Parties} = \text{Clients} \cup \text{Servers}$.

Each authentication factor can be one of two types: *symmetric* or *asymmetric*. Suppose there are n factors; let I_s denote the indices of symmetric factors and I_a denote the indices of asymmetric factors. For each client-server pair $(C, S) \in \text{Clients} \times \text{Servers}$, n authentication factors exist. The ℓ th authentication factor $\text{pw}_{C,S}^\ell$ is chosen uniformly at random from the set Passwords^ℓ and is stored by the client. For symmetric factors, the server also stores $\text{pw}_{C,S}^\ell$; for asymmetric factors, the server stores a *verifier* $\overline{\text{pw}}_{C,S}^\ell$, which is some non-trivial transformation of $\text{pw}_{C,S}^\ell$. (The notion of “non-trivial transformation” will be clear in the freshness definition below, but intuitively the transformation should be such that compromise of the verifier alone should not be sufficient to impersonate the user without performing a dictionary attack.)

Execution of the protocol. During execution, a party may have multiple instances of the protocol running. Each instance i of a party $U \in \text{Parties}$ is treated as an *oracle* denoted by Π_i^U .

In a protocol, there is a sequence of messages, called *flows*, starting with a flow from the client instance, responded to by a server instance, and so on. After some number of flows, an instance may *accept*, at which point it holds a *session key* sk , *partner id* pid , and *session id* sid . Subsequently, it may *terminate*. Two instances Π_i^C and Π_j^S are said to be *partnered* if they both accept, hold $(\text{pid}, \text{sid}, \text{sk})$ and $(\text{pid}', \text{sid}', \text{sk}')$, respectively, with $\text{pid} = S$, $\text{pid}' = C$, $\text{sid} = \text{sid}'$, and $\text{sk} = \text{sk}'$, and no other instance accepts with session id equal to sid . Alternatively, an instance may *reject* at any point in time, meaning it is no longer accepted or terminated.

Queries allowed. The protocol is determined by how participants respond to inputs from the environment, and the environment is considered to be controlled by the adversary, which is formally a probabilistic algorithm that issues queries to a *challenger* which simulates parties’ oracle instances. For a protocol P , the queries that the adversary can issue are defined as follows (where clear by the setting, we may omit the subscript P):

- $\text{Execute}_P(C, i, S, j)$: Causes client instance Π_i^C and server instance Π_j^S to faithfully execute protocol P and returns the resulting transcript.
- $\text{Send}_P(U, i, M)$: Sends message M to user instance Π_i^U , which faithfully performs the appropriate portion of protocol P based on its current state and the message M , updates its state as appropriate, and returns any resulting messages.
- $\text{Test}_P(U, i)$: If user instance Π_i^U has accepted, then the following happens: the challenger chooses $b \in_R \{0, 1\}$; if $b = 1$, then return the session key of Π_i^U , otherwise return a random string of the same length as the session key. This query may only be asked once.

- $\text{RevealSK}_P(U, i)$: If user instance Π_i^U has accepted, then returns session key sk held by Π_i^U .
- $\text{RevealFactor}_P(C, S, \ell)$: Returns the ℓ th factor $\text{pw}_{C,S}^\ell$ held by client C with server S .
- $\text{RevealFactorV}_P(S, C, \ell)$: If ℓ is an asymmetric factor: returns the ℓ th factor’s verifier $\overline{\text{pw}}_{C,S}^\ell$ held by server S with client C .

The RevealFactor and RevealFactorV queries model the adversary learning the authentication secrets, which corresponds to weak corruption in the Bellare-Pointcheval-Rogaway model. We do not allow the adversary to modify stored authentication secrets (also called strong corruption).

Definition 2.1 (Freshness) An instance Π_i^U with partner id U' is fresh in the ℓ th factor (with forward-secrecy) if and only if none of the following events occur:

1. a $\text{RevealSK}(U, i)$ query occurs;
2. a $\text{RevealSK}(U', j)$ query occurs, where $\Pi_j^{U'}$ is the partner instance of Π_i^U , if it exists;
3. if $U \in \text{Clients}$: $\text{RevealFactor}(U, U', \ell)$ (and/or $\text{RevealFactorV}(U', U, \ell)$ if the ℓ th factor is asymmetric) occurs before the Test query, and $\text{Send}(U, i, M)$ occurs for some string M ;
4. if $U \in \text{Servers}$: $\text{RevealFactor}(U', U, \ell)$ occurs before the Test query, and $\text{Send}(U, i, M)$ occurs for some string M .

This notion of freshness accommodates the idea that an instance should remain fresh even if all but one of the authentication factors has been fully compromised. If an instance is fresh in all of its factors, then it is also fresh in the original notion of freshness for PAKE.

Adversary’s goals. For *session key security*, the goal of an adversary is to guess the bit b used in the Test query of an instance that is fresh in at least one of its factors; this corresponds to the ability of an adversary to distinguish the session key from a random string of the same length. Let $\text{Succ}_P^{\text{ake-fl}}(\mathcal{A})$ be the event that the adversary \mathcal{A} makes a single Test query to some fresh in the ℓ th factor instance Π_i^U that has accepted and \mathcal{A} eventually outputs a bit b' , where $b' = b$ and b is the randomly selected bit in the Test query. The *ake-fl advantage* of \mathcal{A} attacking P is defined to be $\text{Adv}_P^{\text{ake-fl}}(\mathcal{A}) = 2 \Pr(\text{Succ}_P^{\text{ake-fl}}(\mathcal{A})) - 1$.

We can define similar notions for *client-to-server*, *server-to-client*, and *mutual* authentication. For the security experiments involving authentication, the Test query is prohibited. We define $\text{Adv}_P^{\text{c2s-fl}}(\mathcal{A})$ to be the probability that a server instance Π_j^S with partner id C terminates without having a partner oracle before the RevealFactor query in point 4 of the definition of freshness in the ℓ th factor. We define $\text{Adv}_P^{\text{s2c-fl}}(\mathcal{A})$ to be the probability that a client instance Π_i^C with partner id S terminates without having a partner oracle before the Reveal* queries in point 3 of the definition of freshness in the ℓ th factor. Finally, we define $\text{Adv}_P^{\text{ma-fl}}(\mathcal{A}) = \max\{\text{Adv}_P^{\text{c2s-fl}}(\mathcal{A}), \text{Adv}_P^{\text{s2c-fl}}(\mathcal{A})\}$.

We overload the Adv (and corresponding $\text{Pr}(\text{Succ})$) notation: $\text{Adv}_P^N(t, q_{\text{se}}, q_{\text{ex}}, q_{\text{ro}}) = \max_{\mathcal{A}}\{\text{Adv}_P^N(\mathcal{A})\}$, where the maximum is taken over all adversaries running in time at most t , making at most q_{se} and q_{ex} queries of type Send_P and Execute_P , respectively, and at most q_{ro} random oracle queries.

Definition 2.2 (Secure multi-factor protocol)

Let κ be a security parameter. A protocol P is

a secure multi-factor password authenticated key agreement protocol *if there exists a negligible (in κ) ϵ and small constants δ_ℓ , $\ell \in \{1, \dots, n\}$, such that, for all polynomially-bounded adversaries \mathcal{A} ,*

$$\text{Adv}_P^{\text{ake-fl}}(\mathcal{A}) \leq \begin{cases} \frac{\delta_\ell q_{se}}{|\text{Passwords}^\ell|} + \epsilon, & \text{if the } \ell\text{th factor is symmetric,} \\ \frac{\delta_\ell((1-b_{co}^\ell)q_{se}+b_{co}^\ell q_{ro})}{|\text{Passwords}^\ell|} + \epsilon, & \text{if the } \ell\text{th factor is asymmetric,} \end{cases}$$

and the corresponding bound applies for $\text{Adv}_P^{\text{ma-fl}}(\mathcal{A})$, where, for asymmetric factors ℓ , $b_{co}^\ell = 1$ if \mathcal{A} makes a $\text{RevealFactorV}(\cdot, \cdot, \ell)$ query and 0 otherwise.

Intuitively, this notion of security says that any polynomially-bounded adversary can only do negligibly better than doing an online dictionary attack at any unknown factors and can gain no advantage by doing an offline dictionary attack. Ideally, δ_ℓ would be 1, indicating the adversary can only rule out one password with each online guess; however, a protocol can still be secure as long as δ_ℓ is small compared to $|\text{Passwords}^\ell|$.

Since an instance that is fresh in all of its factors is also fresh in the original ake notion of PAKE, we have that

$$\text{Adv}_P^{\text{ake}}(\mathcal{A}) \leq \min_{\ell \in \{1, \dots, n\}} \left\{ \text{Adv}_P^{\text{ake-fl}}(\mathcal{A}) \right\}.$$

By providing bounds for each factor, we can provide greater granularity in relating the security of factors to their risks of compromise. For example, lower entropy factors (represented by smaller values of $|\text{Passwords}^\ell|$) may be physically distributed and secured in different ways than higher entropy factors, or may be used for a shorter period of time. This contrasts with the approach of (Pointcheval & Zimmer 2008), in which there is a single notion of freshness and a single bound over all factors.

2.2 Using one-time passwords

The model presented in Section 2.1 uses long-term authentication secrets that do not change over time. However, multi-factor authentication may include a factor that varies, such as a one-time password. Such a factor may be the response to a challenge, or may vary with time. The benefit of a one-time password is that the compromise of a single one-time password should not affect the security for a different one-time password. One-time passwords offer some protection against passive spyware, as previously compromised one-time passwords are useless.

Although at first glance it may seem impractical for a user to store a large number of passwords, this is actually quite practical and is already being done in the real world: for example, some European banks issue paper lists of one-time passwords to users (Nordea Bank 2009), and corporations issue hardware devices for pseudorandomly generating one-time passwords for virtual private network (VPN) access (RSA Security Inc. 2009) or electronic commerce (Blizzard Entertainment 2009). Even though a user may be carrying as much data as in a cryptographic key, one-time passwords offer usability benefits: carrying a cryptographic key requires a hardware interface or carefully managed private key files, whereas one-time passwords can be easily entered in only a few keystrokes.

Abdalla *et al.* (Abdalla et al. 2005) present a protocol for the use of one-time passwords in an authenticated key exchange protocol but do not alter

the security model from the standard BPR setting. Paterson and Stebila (Paterson & Stebila 2009) do present an alteration to the BPR security model that accommodates the compromise of previous (and future) one-time passwords and we apply their ideas to allow for symmetric factors using one-time passwords as follows.

Adjusting the model. We can alter the security definition of a multi-factor protocol to allow a symmetric factor that corresponds to a one-time password by applying the ideas of Paterson and Stebila (Paterson & Stebila 2009). Let ℓ be the index of a symmetric factor for which we wish to use one-time passwords. Let Indices^ℓ be the set of indices of one-time passwords, and let $\text{ch} \in \text{Indices}^\ell$. When a party is activated, they are activated with the index of the one-time password to use for that instance; a party can only be activated once for each $\text{ch} \in \text{Indices}^\ell$. Let $\{\text{pw}_{C,S,\text{ch}}^\ell\}$ be the set of one-time passwords between C and S , indexed by ch ; each such password is chosen uniformly at random from Passwords^ℓ . We add an additional parameter ch to the RevealFactor query:

- $\text{RevealFactor}_P(C, S, \text{ch}, \ell)$: Returns the ℓ th factor $\text{pw}_{C,S,\text{ch}}^\ell$ held by client C with server S for one-time password indexed by ch .

The definition of freshness in the ℓ th factor of Π_i^U is adjusted as well, replacing points 3 and 4 in Definition 2.1 with:

3. if $U \in \text{Clients}$: $\text{RevealFactor}(U, U', \text{ch}, \ell)$ occurs before the Test query, and $\text{Send}(U, i, M)$ occurs for some string M , where ch is the index of the one-time password with which Π_i^U was activated;
4. if $U \in \text{Servers}$: $\text{RevealFactor}(U', U, \text{ch}, \ell)$ occurs before the Test query, and $\text{Send}(U, i, M)$ occurs for some string M , where ch is the index of the one-time password with which Π_i^U was activated.

The definitions of authentication are adjusted analogously as well.

Paterson and Stebila go on to show that any secure PAKE protocol can be used in the natural way to build to a secure one-time PAKE protocol, by using the one-time password in place of the password. This holds even when the one-time passwords are pseudorandomly generated or time-dependent. This means that our MFPAK protocol in the next section can easily accommodate one-time passwords as authentication factors.

3 MFPAK: a multi-factor password-authenticated key exchange protocol

MFPAK is the first PAKE protocol that uses multiple low-entropy authentication factors. It allows for an arbitrary number of factors which can be asymmetric or symmetric, and these factors can be independently changed as users need to change their passwords. Our approach is much more efficient, in terms of number of expensive operations, than the naïve approach of combining existing PAKE protocols as black boxes: we add no expensive operations for each additional symmetric factor, and only one additional expensive operation (signature generation/verification) for each party for each asymmetric factor.

3.1 Design ideas

We designed MFPAK by considering two existing one-factor protocols as our building blocks: the asymmetric password protocol PAK-Z+ for asymmetric factors, and the symmetric password protocol PAK for symmetric factors. These two protocols are similar in structure which allows us to gain some efficiency improvements. All factors are tightly inte-

grated into the authentication and key exchange processes. The underlying session key agreement comes from a hashed Diffie-Hellman construction. Authentication for asymmetric factors is done using a digital signature scheme, while for symmetric factors it is done using hash functions.

Shielded ephemeral key. One of the main efficiency and security gains in the MFPK protocol comes in the first flow from the client to the server. In this flow, the client shields its ephemeral public key by multiplying it by (the hash of) each factor. The client is made to commit to those values, thereby preventing a malicious client from making an offline dictionary attack later on. Moreover, the server must use the same values to unshield the client's ephemeral public key or Diffie-Hellman key agreement will fail, thereby committing the server to its choice of values. By doing this multiple shielding operation, the client and server achieve mutual authentication, the client saves expensive operations compared to running multiple protocols separately, and the authentication secrets are protected.

Digital signature for asymmetric factors. Authentication for asymmetric factors comes from using a digital signature scheme, where the (short) authentication secret is used to shield the digital signature private key which is stored on the server. During the login stage of the protocol, the server returns the shielded private key, which the client can unwrap only if she knows the correct password. The client uses the private key to perform a signing operation which the server verifies using the public key. This allows for asymmetry: the compromise of the server's database is not enough to impersonate the client to the server without a dictionary attack. This technique, as used in PAK-Z+ (Gentry et al. 2005), is an instantiation of the generic technique proposed by Gentry *et al.* (Gentry et al. 2006) for asymmetric password-based authentication. It is important to note that the digital signature scheme is not used in its normal sense with published or certified public keys, but simply as a convenient asymmetric construction.

Hash function for symmetric factors. The hash of a symmetric factor is stored on the server. The server proves its knowledge of a symmetric factor by hashing it with the session key; the client does the same.

3.2 Protocol specification

The MFPK protocol, like many other protocols, contains two stages: a user registration stage, completed once per client-server pair, and a login stage, completed each time a user attempts to login. For convenience in presentation of the login stage, we assume there is at least one symmetric factor and one asymmetric factor; however, the protocol can be altered in the natural way to deal with exclusively symmetric or exclusively asymmetric factors. The number and type of factors are fixed and publicly known.

Ingredients and notation. Let κ be a cryptographic security parameter. The notation $z \in_R Z$ denotes an element z selected uniformly at random from a set Z . Angle brackets $\langle \cdot \rangle$ denote a list, and $\cdot || \cdot$ denotes concatenation. The protocol operates over a finite cycle group G of order q , generated by g , for which the Computational Diffie-Hellman (CDH) assumption holds. The function $\text{Acceptable}(\cdot)$ tests whether an element is in G (or, for efficiency reasons, a group containing G ; see (MacKenzie 2002, §4)). It makes use of a number of random hash functions based on random oracles (Bellare & Rogaway 1993): H_1 maps $\{0, 1\}^*$ to group elements (such as (Coron & Icart 2009) for hashing into elliptic curve groups), while all other hash functions H_i map $\{0, 1\}^*$ to $\{0, 1\}^\kappa$. We also employ a signature scheme $\mathcal{S} = (\text{Gen}, \text{Sign}, \text{Verify})$ that is existentially unforgeable under chosen message at-

tacks (Goldwasser et al. 1988). Let $(v, V) \leftarrow \text{Gen}(1^\kappa)$, where v is a private key and V is the corresponding public key. Recall that $\text{pw}_{C,S}^\ell$ denotes client C 's password for server S for the ℓ th factor, and $\overline{\text{pw}}_{C,S}^\ell$ denotes the corresponding value held by the server, which may be equal to $\text{pw}_{C,S}^\ell$ for symmetric factors and is some non-trivial transformation of $\text{pw}_{C,S}^\ell$ for asymmetric factors.

The *user registration stage* of MFPK is given in Figure 1. This stage should be completed over a private, authentic channel. The user registration stage can be altered in the obvious way to have authentication secrets chosen by the server and supplied to the client, if necessary.

The *login stage* of MFPK is given in Figure 2. This stage can be completed over a public, untrusted channel. A client C initiates the login stage with a server S .

3.3 Nature of the factors

The MFPK protocol can accommodate a wide variety of authentication secrets using either symmetric or asymmetric factors, as we note below. Our approach offers improved functionality compared with the naive way of combining multiple authentication secrets by simply concatenating them into one long string: with concatenation, one cannot easily combine passwords that change over time (symmetric factors) with long-term passwords (asymmetric factors) because the server does not store the plaintext password.

Long-term passwords. Long-term passwords are best accommodated as an asymmetric factor, but can be treated asymmetrically as well. Since long-term passwords do not change frequently (or at all), we should reduce the damage that can be caused by compromise of the server database containing data for these factors. Although we can never prevent dictionary attacks against the server's database, we can raise the amount of work an attacker needs to do by using asymmetric factors.

One-time passwords. One-time passwords are usually best accommodated as symmetric factors. Asymmetric factors could be used, but the costs for asymmetric factors may not be worth it for one-time passwords. It may be more efficient to generate one-time passwords from a seed using a challenge-response mechanism or a time-dependent generator. For factors that employ a challenge-response mechanism, an initial message from the server to the client conveying the challenge can be added to the beginning of the login stage of the protocol.

Cryptographic keys. Although our primary motivation has been the use of short strings as authentication secrets so users can easily carry their authentication secrets between computers, there is nothing preventing a password-based protocol from using high-entropy secrets (that is, cryptographically large keys) as opposed to low-entropy secrets. We can directly use a cryptographic key as $\text{pw}_{C,S}^\ell$ in either the symmetric or asymmetric case. In the asymmetric case, it would be possible to further streamline the protocol by having the user store the private key v_ℓ from the digital signature scheme, and adjust the remainder of the protocol as follows: set $\text{pw}_{C,S}^\ell \leftarrow v_\ell$; in the registration stage, the server stores $\overline{\text{pw}}_{C,S}^\ell \leftarrow \langle \tau_\ell, \tau_\ell^{-1}, V_\ell \rangle$; in the login stage, the server omits steps 15 and 16 for this factor and the client omits steps 22–25 for this factor. We recommend, however, that situations using exclusively cryptographically large keys should consider traditional authenticated key exchange protocols as the security models (Canetti & Krawczyk

MFPAK User Registration	
Client C	Server S
for $\ell \in \{1, \dots, n\}$:	
1. store $\text{pw}_{C,S}^\ell \in_R \text{Passwords}^\ell$	
2. $\tau_\ell \leftarrow H_1(C, S, \ell, \text{pw}_{C,S}^\ell)$	
for $\ell \in I_a$:	
3. $(v_\ell, V_\ell) \leftarrow_R \text{Gen}(1^\kappa)$	
4. $v'_\ell \leftarrow H_2(C, S, \ell, \text{pw}_{C,S}^\ell) \oplus v_\ell$	
5. $v''_\ell \leftarrow H_3(\ell, v_\ell)$	
6.	$C, \{\tau_\ell\}, \{V_\ell\}, \{v'_\ell\}, \{v''_\ell\}$
7.	for $\ell \in I_s$: store $\overline{\text{pw}}_{C,S}^\ell \leftarrow \langle \tau_\ell, \tau_\ell^{-1} \rangle$
8.	for $\ell \in I_a$: store $\overline{\text{pw}}_{C,S}^\ell \leftarrow \langle \tau_\ell, \tau_\ell^{-1}, V_\ell, v'_\ell, v''_\ell \rangle$

Figure 1: The user registration stage of the MFPAK protocol.

2001, LaMacchia et al. 2007) are stronger and offer resistance to ephemeral key leakage in addition to static key leakage.

Biometrics. Pointcheval and Zimmer (Pointcheval & Zimmer 2008) describe in detail the use of biometric templates in an authenticated key exchange protocol. They use secure sketches and fuzzy extractors to safely see if two biometric templates match.

An alternative approach is to use *fuzzy vaults*, which were introduced by Juels and Sudan (Juels & Sudan 2002). They allow a secret to be embedded in a *vault* which is locked by a set of fuzzy values, such as the minutiae of a fingerprint. Fuzzy vaults could for example be used in a multi-factor protocol as follows: the user receives the fuzzy vault, uses her biometric to unlock the vault, and then uses the embedded secret value as another factor in the multi-factor protocol.

Because of the privacy issues surrounding biometrics, we are not suggesting that biometrics naïvely be used in our construction immediately, as there are numerous issues to consider. For example, should the fuzzy vault be transmitted unencrypted or encrypted under the session key derived from the other factors? Should the secret embedded in the vault contain error correcting information, as suggested in (Juels & Sudan 2002), or not? (We think not, as error correcting information allows an offline “dictionary” attacker to detect whether it has the right input, whereas lack of error correction information would ideally mean the attacker needs to do an online “dictionary” attack.) The use of biometrics in authenticated key exchange merits further study.

3.4 Efficiency

In many e-commerce and online banking situations, the performance-limiting factor is the number of connections a server can handle, and this is in turn limited by the number of expensive operations required by the cryptographic protocol. MFPAK can increase security without a substantial additional computational burden on the server.

Figure 3 compares the number of expensive operations (group exponentiations and signature generation / verification) performed by a naïve combination of PAK and PAK-Z+ versus the MFPAK protocol. MFPAK has a fixed overhead of two group exponentiations each on client and server side. For each symmetric factor, there are no additional expensive operations (only multiplications and hashes, not exponentiations); for each asymmetric factor, there is one additional expensive operation on each side (signature generation by the client, signature verification by the server). This makes MFPAK much more efficient, in terms of number of expensive operations, than if one were to make a multi-factor scheme sim-

ply by running PAK and PAK-Z+ in parallel independently.

3.5 Security analysis

The main idea of the security argument is that, if one factor, say the ℓ^* th factor, remains uncompromised, then the difficulty of breaking MFPAK is related to the difficulty of breaking the corresponding one of either PAK (for a symmetric factor) or PAK-Z+ (for an asymmetric factor), each which is in turn related to solving the Computational Diffie-Hellman problem.

For both symmetric and asymmetric factors, we describe a procedure (specified by a modifier \mathcal{M}) to transform an adversary \mathcal{A} against MFPAK with the ℓ^* th factor uncompromised into an adversary \mathcal{A}^* against the corresponding one of the two underlying protocols (PAK and PAK-Z+, respectively). The transformations are such that, if the oracle instance in MFPAK against which the Test query is directed is fresh in the ℓ^* th factor, then the corresponding oracle instance is also fresh in the corresponding attack on PAK (resp., PAK-Z+). This is possible because of the design of the MFPAK protocol: it essentially runs both PAK and PAK-Z+ together while still capturing the security of each independently. This design characteristic allows the relatively straightforward (although lengthy) security argument.

Our formal argument proceeds by considering four cases, two corresponding to an asymmetric factor being uncompromised and two corresponding a symmetric factor being uncompromised. The cases are:

1. Asymmetric factor uncompromised, $U^* \in \text{Clients}$: no $\text{RevealFactor}_{\text{MFPAK}}(U^*, U^*, \ell^*)$ or $\text{RevealFactorV}_{\text{MFPAK}}(U^*, U^*, \ell^*)$ query.
2. Asymmetric factor uncompromised, $U^* \in \text{Servers}$: no $\text{RevealFactor}_{\text{MFPAK}}(U^*, U^*, \ell^*)$ query.
3. Symmetric factor uncompromised, $U^* \in \text{Clients}$: no $\text{RevealFactor}_{\text{MFPAK}}(U^*, U^*, \ell^*)$ query.
4. Symmetric factor uncompromised, $U^* \in \text{Servers}$: no $\text{RevealFactor}_{\text{MFPAK}}(U^*, U^*, \ell^*)$ query.

These four cases are combined probabilistically to give the overall result. The details are provided in Appendix A. Throughout, we assume passwords are uniformly distributed. The resulting security statement is as follows:

Theorem 3.1 *Let κ be a security parameter. Let G be a finite cyclic group generated by g and let S be a signature scheme. Let \mathcal{A} be an adversary that runs in time polynomial in κ , and makes at most q_{se} and q_{ex} queries of type Send and Execute, respectively, and at most q_{ro} queries to the random oracle. If ℓ is an asymmetric factor, then let $b_{\text{co}} = 1$ if \mathcal{A} makes a $\text{RevealFactorV}(\cdot, \cdot, \ell)$ query to a server, and 0 otherwise. Then MFPAK is a secure multi-factor PAKE*

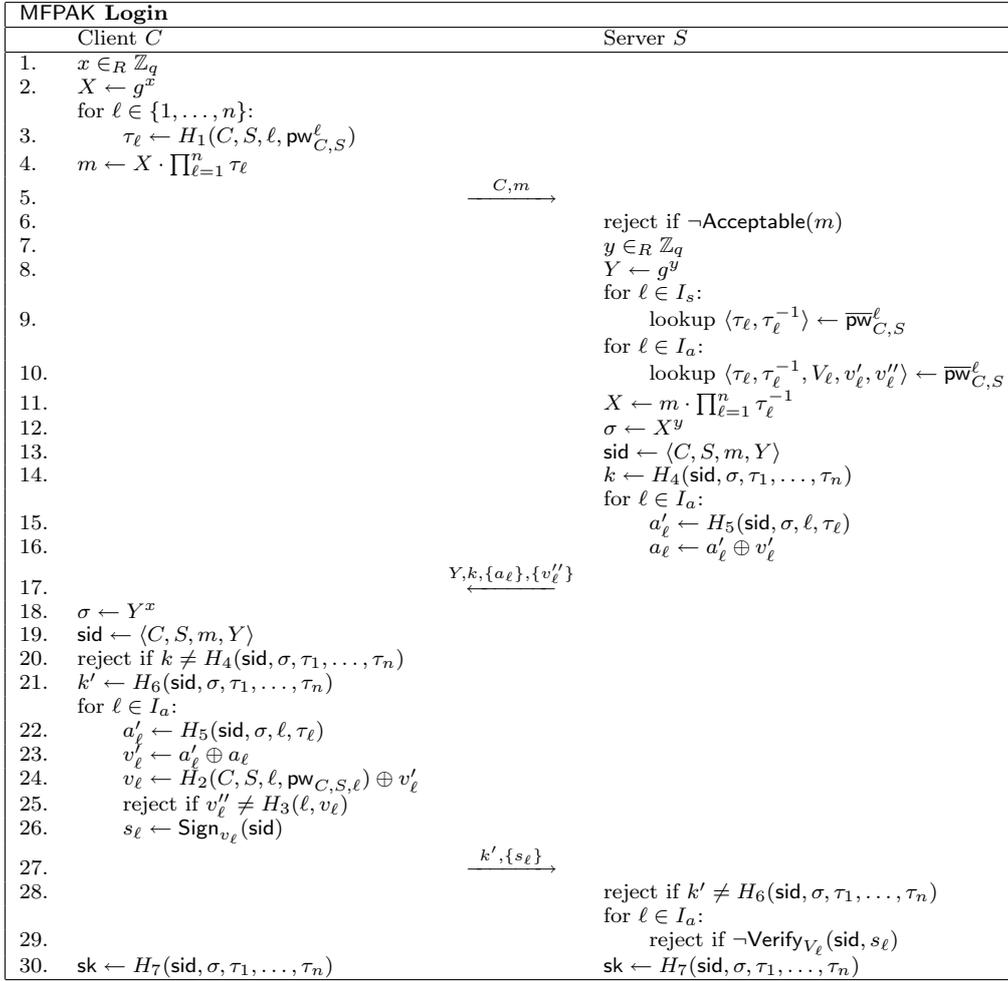


Figure 2: The login stage of the MFPAK protocol.

Operation	PAK & PAK-Z+		MFPAK	
	Client	Server	Client	Server
exponentiations	$2 I_s + 2 I_a $	$2 I_s + 2 I_a $	2	2
signature generation	$ I_a $	0	$ I_a $	0
signature verification	0	$ I_a $	0	$ I_a $
total	$2 I_s + 3 I_a $	$2 I_s + 3 I_a $	$2 + I_a $	$2 + I_a $

Figure 3: Comparison of expensive operations for combined PAK & PAK-Z+ and MFPAK.

protocol, with

$$\text{Adv}_{\text{MFPAK}}^{\text{ake-fl}}(\mathcal{A}) \leq \begin{cases} \frac{16\delta((1-b_{\text{co}})q_{\text{se}} + b_{\text{co}}q_{\text{ro}})}{|\text{Passwords}^\ell|} + \epsilon, & \text{if the } \ell\text{th factor is symmetric,} \\ \frac{4\delta q_{\text{se}}}{|\text{Passwords}^\ell|} + \epsilon, & \text{if the } \ell\text{th factor is asymmetric,} \end{cases}$$

where ϵ is a negligible function of κ , and $\delta = |\text{Clients}| \cdot |\text{Servers}|$; a similar bound exists for $\text{Adv}_{\text{MFPAK}}^{\text{ma-fl}}(\mathcal{A})$.

As with any formal security argument, a proof of security does not imply security against all forms of attack. A protocol may be vulnerable to attack methods not described by the security model. Nonetheless, a security proof is valuable as a heuristic that the protocol is resistant to at least some types of attacks.

4 Conclusion and future work

We have presented a security model for multi-factor password-authenticated key exchange protocols that can accommodate an arbitrary number of factors. We

have provided a security argument showing that our new protocol, MFPAK, is secure in this model. Our multi-factor authentication protocol offers enhanced authentication protection through the use of complementary factors, such as a long-term password and a one-time challenge/response. The construction is quite efficient in terms of the number of operations per factor; for example, a two-factor version of our protocol using a long-term password and one-time challenge/response has the same efficiency as the one-factor protocol PAK-Z+. The protocol remains secure even if all but one of the authentication factors is fully known to an adversary. Our multi-factor protocol is resistant to man-in-the-middle and impersonation attacks, providing enhanced authentication in the face of more complex threats like phishing.

Other recent work in the field of PAKE protocols has focused on protocols where the sequence of flows fits existing network protocols such as SSL/TLS. An open question is to design a provably secure multi-factor PAKE protocol with support for asymmetric factors that fits within the message flow of SSL/TLS.

Additionally, multi-factor protocols supporting an

arbitrary number of factors could be designed where some factors are optional and the number of factors used corresponds to differing levels of access depending on the application situation: one factor could be used for read-only access, two factors for small-value transactions, and three factors for large-value transactions.

An interesting future direction would be to further investigate the use of biometric information in a multi-factor authenticated key exchange protocol. We have outlined some ideas involving fuzzy vaults, but consideration of the privacy and security requirements requires further research.

Acknowledgements

This research performed while D.S. was at the University of Waterloo and S.C. was at Sun Microsystems Laboratories. D.S. was supported in part by an NSERC Canada Graduate Scholarship. The authors gratefully acknowledge helpful discussions with Alfred Menezes, Bodo Möller, Michele Mosca, and Berkant Ustaoglu, and appreciate the feedback of anonymous referees.

References

- Abdalla, M., Chevassut, O. & Pointcheval, D. (2005), One-time verifier-based encrypted key exchange, in (Vaudey 2005), pp. 47–64. Full version available as **URL:** <http://www.di.ens.fr/~mabdalla/papers/ACP05-letter.pdf>
- Bellare, M., Pointcheval, D. & Rogaway, P. (2000), Authenticated key exchange secure against dictionary attacks, in (Preneel 2000), pp. 139–155.
- Bellare, M. & Rogaway, P. (1993), Random oracles are practical: a paradigm for designing efficient protocols, in *Proc. 1st ACM Conference on Computer and Communications Security (CCS)* (1993), ACM, pp. 62–73.
- Bellovin, S. M. & Merritt, M. (1992), Encrypted key exchange: Password-based protocols secure against dictionary attacks, in *Proceedings of the 1992 IEEE Computer Society Conference on Research in Security and Privacy*, IEEE.
- Bellovin, S. M. & Merritt, M. (1993), Augmented encrypted key exchange: a password-based protocol secure against dictionary attacks and password file compromise, in *Proc. 1st ACM Conference on Computer and Communications Security (CCS)* (1993), pp. 244–250.
- Blizzard Entertainment (2009), Blizzard authenticator. **URL:** http://eu.blizzard.com/support/article.xml?locale=en_GB&articleId=28152
- Boyko, V., MacKenzie, P. & Patel, S. (2000), Provably secure Password-Authenticated Key exchange using Diffie-Hellman, in (Preneel 2000), pp. 156–171. Full version available as **URL:** <http://eprint.iacr.org/2000/044>
- Canetti, R., Halevi, S., Katz, J., Lindell, Y. & MacKenzie, P. (2005), Universally composable password-based key exchange, in R. Cramer, ed., *Advances in Cryptology – Proc. EUROCRYPT 2005*, Vol. 3494 of *LNCS*, Springer, pp. 404–421.
- Canetti, R. & Krawczyk, H. (2001), Analysis of key-exchange protocols and their use for building secure channels, in B. Pfitzmann, ed., *Advances in Cryptology – Proc. EUROCRYPT 2001*, Vol. 2045 of *LNCS*, Springer, pp. 453–474. Full version available as **URL:** <http://eprint.iacr.org/2001/040>
- Coron, J.-S. & Icart, T. (2009), A random oracle into elliptic curves. **URL:** <http://eprint.iacr.org/2009/340>
- Gentry, C., MacKenzie, P. & Ramzan, Z. (2005), PAK-Z+. Contribution to the IEEE P1363-2000 study group for Future PKC Standards. **URL:** <http://grouper.ieee.org/groups/1363/WorkingGroup/presentations/pakzplusv2.pdf>
- Gentry, C., MacKenzie, P. & Ramzan, Z. (2006), A method for making password-based key exchange resilient to server compromise, in C. Dwork, ed., *Advances in Cryptology – Proc. CRYPTO 2006*, Vol. 4117 of *LNCS*, Springer, pp. 142–159.
- Goldwasser, S., Micali, S. & Rivest, R. L. (1988), A digital signature scheme secure against adaptive chosen-message attacks, *SIAM Journal on Computing* **17**(2), 281–308.
- Juels, A. & Sudan, M. (2002), A fuzzy vault scheme, in *Proc. IEEE International Symposium on Information Theory (ISIT) 2002*, IEEE Press, p. 408. Full version available as **URL:** <http://www.rsa.com:80/rsalabs/node.asp?id=2061>
- LaMacchia, B., Lauter, K. & Mityagin, A. (2007), Stronger security of authenticated key exchange, in W. Susilo, J. K. Liu & Y. Mu, eds, *First International Conference on Provable Security (ProvSec) 2007*, Vol. 4784 of *LNCS*, Springer, pp. 1–16.
- MacKenzie, P. (2002), The PAK suite: Protocols for password-authenticated key exchange, Technical Report 2002-46, DIMACS Center, Rutgers University. **URL:** <http://dimacs.rutgers.edu/TechnicalReports/abstracts/2002/2002-46.html>
- Nordea Bank (2009), Netbank security. **URL:** <http://www.nordea.ee/Private+customers/E-channels++Netbank/Netbank/Netbank+Security/936612.html>
- Park, Y. M. & Park, S. G. (2004), Two factor authenticated key exchange (TAKE) protocol in public wireless LANs, *IEICE Transactions on Communications* **E87-B**(5), 1382–1385.
- Paterson, K. G. & Stebila, D. (2009), One-time-password-authenticated key exchange. **URL:** <http://eprint.iacr.org/2009/430>
- Pointcheval, D. & Zimmer, S. (2008), Multi-factor authenticated key exchange, in S. M. Bellovin & R. Gennaro, eds, *Applied Cryptography and Network Security (ACNS) 2008*, Vol. 5037 of *LNCS*, Springer, pp. 277–295.
- Preneel, B., ed. (2000), *Advances in Cryptology – Proc. EUROCRYPT 2000*, Vol. 1807 of *LNCS*, Springer.
- RSA Security Inc. (2009), RSA SecurID. **URL:** <http://www.rsa.com/node.aspx?id=1156>
- Schechter, S., Dhamija, R., Ozment, A. & Fischer, I. (2007), The emperor’s new security indicators: An evaluation of website authentication and the effect of role playing on usability studies, in *Proc. IEEE Symposium on Security and Privacy (S&P) 2007*, IEEE Press, pp. 51–65.
- Stebila, D., Udipi, P. & Chang, S. (2009), Multi-factor password-authenticated key exchange (full version). **URL:** <http://eprint.iacr.org/2008/214>
- Vaudenay, S., ed. (2005), *Public Key Cryptography (PKC) 2005*, Vol. 3386 of *LNCS*, Springer.
- Yang, G., Wong, D. S., Wang, H. & Deng, X. (2006), Formal analysis and systematic construction of two-factor authentication scheme (short paper), in P. Ning, S. Qing & N. Li, eds, *Proc. 8th International Conference on Information and Communications Security (ICICS) 2006*, Vol. 4307 of *LNCS*, Springer, pp. 82–91. Full version available as **URL:** <http://eprint.iacr.org/2006/270>
- Yoon, E.-J. & Yoo, K.-Y. (2006), An optimized two factor authenticated key exchange protocol in PWLANs, in V. N. Alexandrov, G. D. van Albada, P. M. Sloot & J. Dongarra, eds, *Computational Science – ICCS 2006*, Vol. 3992 of *LNCS*, Springer, pp. 1000–1007.

A Security analysis

This section contains the details of the security analysis supporting Theorem 3.1.

It is helpful to be able to refer to the action of a party upon receipt of a message. We use the notation CLIENTACTION_{iP} and SERVERACTION_{iP} to refer to the portion of the protocol P performed by the client or server, respectively, after the i th flow. Thus, MFPAK as described in Figure 2 specifies $\text{CLIENTACTION0}_{\text{MFPAK}}$, $\text{SERVERACTION1}_{\text{MFPAK}}$, $\text{CLIENTACTION2}_{\text{MFPAK}}$, and $\text{SERVERACTION3}_{\text{MFPAK}}$.

A.1 Ingredients

Computational Diffie-Hellman assumption. MFPAK operates over a finite cycle group G for which the Computational Diffie-Hellman (CDH) assumption holds. Let G be a finite cyclic group of order q , let g be a generator of G , and let t_{exp} be the time it takes to perform an exponentiation in G . Let $\text{Acceptable} : \overline{G} \rightarrow \{\text{true}, \text{false}\}$ such that $\text{Acceptable}(z) = \text{true}$ if and only if $z \in \overline{G}$, where \overline{G} is a specified abelian group which has G as a subgroup. For two values X and Y , define $\text{DH}(X, Y) = X^y$, if $\text{Acceptable}(X)$ and $Y = g^y$, or $\text{DH}(X, Y) = Y^x$, if $\text{Acceptable}(Y)$ and $X = g^x$. Let \mathcal{A} be a probabilistic algorithm with input (G, g, X, Y) that outputs a subset of G , and define

$$\text{Adv}_{G,g}^{\text{cdh}}(\mathcal{A}) = \Pr(\text{DH}(X, Y) \in \mathcal{A}(G, g, X, Y) : \\ (x, y) \in_R \mathbb{Z}_q, X = g^x, Y = g^y) .$$

Let $\text{Adv}_{G,g}^{\text{cdh}}(t, n) = \max_{\mathcal{A}} \{\text{Adv}_{G,g}^{\text{cdh}}(\mathcal{A})\}$ where the maximum is taken over all algorithms running in time t and outputting a subset of size at most n . The CDH assumption is that, for any probabilistic polynomial time algorithm \mathcal{A} , $\text{Adv}_{G,g}^{\text{cdh}}(\mathcal{A})$ is negligible.

Random hash functions. MFPAK makes use of a number of random hash functions based on random oracles (Bellare & Rogaway 1993). A *random hash function* $H : \{0, 1\}^* \rightarrow \{0, 1\}^k$ is constructed by selecting each bit of $H(x)$ uniformly at random and independently for every $x \in \{0, 1\}^*$. We make use of a number of independent random hash functions H_1, H_2, \dots , which can be constructed from a single random hash function H by setting $H_\ell(x) = H(\ell \| x)$. Constructing a hash function that outputs elements of a group instead of $\{0, 1\}^*$ is also possible and efficient, and in fact all of the hash functions used in MFPAK are into the group G .

Signature scheme. MFPAK makes use of a signature scheme $\mathcal{S} = (\text{Gen}, \text{Sign}, \text{Verify})$ that is existentially unforgeable under chosen message attacks (Goldwasser et al. 1988). Let $(v, V) \leftarrow \text{Gen}(1^\kappa)$, where v is a private key and V is the corresponding public key. Let t_{Gen} be the runtime of $\text{Gen}(1^\kappa)$, and t_{sig} be the runtime of Sign and Verify . A forger \mathcal{F} is given a public key V and must forge signatures; it can query an oracle that returns $\text{Sign}_v(m)$ for any messages m of its choice. It succeeds if it can output a forgery (m, σ) such that $\text{Verify}_V(m, \sigma) = \text{true}$, where m was not queried to the signing oracle. Let $\text{Succ}_{\mathcal{S}, \kappa}^{\text{eu-cma}}(\mathcal{F}) = \Pr(\mathcal{F} \text{ succeeds})$, and $\text{Succ}_{\mathcal{S}, \kappa}^{\text{eu-cma}}(t, q_{\text{Sign}}) = \max_{\mathcal{F}} \{\text{Succ}_{\mathcal{S}, \kappa}^{\text{eu-cma}}(\mathcal{F})\}$ where the maximum is taken over all forgers running in time t and making at most q_{Sign} queries to the signing oracle. A signature scheme \mathcal{S} is *existentially unforgeable under chosen message attacks (eu-cma)* if, for any probabilistic polynomial time algorithm \mathcal{F} , $\text{Succ}_{\mathcal{S}, \kappa}^{\text{eu-cma}}(\mathcal{F})$ is negligible.

A.2 Case 1: Attacking a client instance, asymmetric factor uncompromised

This case addresses impersonation of the server when the instance being attacked is a client instance and the uncompromised ℓ^* th factor is asymmetric.

The modifier \mathcal{M} first uniformly at random guesses $U^* \in_R \text{Clients}$ and $U'^* \in_R \text{Servers}$ as its guess of who the adversary \mathcal{A} will end up attacking. If the attacker ends up attacking the pair of users the modifier has guessed, then we will show how to transform the attack into an attack on PAK-Z+.

Let GuessCS be the event that the modifier \mathcal{M} correctly guesses U^* and U'^* . Then

$$\Pr(\text{GuessCS}) = \Pr((U^* \text{ correct}) \wedge (U'^* \text{ correct})) \quad (1) \\ \geq \frac{1}{|\text{Clients}| \cdot |\text{Servers}|} . \quad (2)$$

For this case, we assume that no $\text{RevealFactor}_{\text{MFPAK}}(U^*, U'^*, \ell^*)$ or $\text{RevealFactorV}_{\text{MFPAK}}(U^*, U^*, \ell^*)$ query is issued against \mathcal{M} : this case models server impersonation in the ℓ^* th factor, which is why no $\text{RevealFactorV}_{\text{MFPAK}}(U'^*, U^*, \ell^*)$ query is allowed. Furthermore, no $\text{RevealFactor}_{\text{MFPAK}}(U^*, U'^*, \ell^*)$ is allowed because an adversary can easily recover the verifier $\overline{\text{pw}}_{U^*, U'^*}^{\ell^*}$ from the secret $\text{pw}_{U^*, U'^*}^{\ell^*}$ and one interaction with U'^* .

The modifier \mathcal{M} does the following to convert an MFPAK adversary \mathcal{A} into a PAK-Z+ adversary \mathcal{A}^* .

Password preparation. For each $(C, S, \ell) \in \text{Clients} \times \text{Servers} \times \{1, \dots, n\} \setminus \{(U^*, U'^*, \ell^*)\}$, \mathcal{M} sets $\text{pw}_{C,S}^\ell \in_R \text{Passwords}^\ell$ and constructs the corresponding $\overline{\text{pw}}_{C,S}^\ell$. Of all the authentication secrets, only $\text{pw}_{U^*, U'^*}^{\ell^*}$ and $\overline{\text{pw}}_{U^*, U'^*}^{\ell^*}$ remain unknown to \mathcal{M} at this point. Compute the corresponding τ_ℓ , for $\ell \neq \ell^*$, and set $\pi \leftarrow \prod_{\ell=1, \ell \neq \ell^*}^n \tau_\ell$.

Instantiation of PAK-Z+ simulator. We instantiate the PAK-Z+ simulator $\mathcal{S}_{\text{PAK-Z+}}$ with the following random oracles: $H_i^*(C, S, \text{pw}_{C,S}) := H_i(C, S, \ell^*, \text{pw}_{C,S})$ for $i = 1, 2$; $H_3^*(v) := H_3(\ell^*, v)$;

$$H_4^*(\langle C, S, m, Y \rangle, \sigma, \tau^{-1}) \\ := H_4(\langle C, S, m \cdot \pi, Y \rangle, \sigma, \tau_1, \dots, \tau, \dots, \tau_n) \\ \parallel_{\ell \in I_a, \ell \neq \ell^*} H_5(\langle C, S, m \cdot \pi, Y \rangle, \sigma, \ell, \tau_\ell) ;$$

$H_5^*(\langle C, S, m, Y \rangle, \sigma, \tau^{-1}) := H_5(\langle C, S, m \cdot \pi, Y \rangle, \sigma, \ell^*, \tau)$; and $H_7^*(\langle C, S, m, Y \rangle, \sigma, \tau^{-1}) := H_7(\langle C, S, m \cdot \pi, Y \rangle, \sigma, \tau_1, \dots, \tau, \dots, \tau_n)$.¹ These 'starred' functions are independent random oracles if the corresponding unstarred functions are. The above construction is possible since $\{\tau_\ell\}_{\ell \neq \ell^*}$ and π are fixed and known to \mathcal{M} because of the guesses made at the beginning of this case. By using a concatenation of random oracles, the PAK system computes the values we need in \mathcal{M} 's handling of Execute and Send queries.

Further, $\mathcal{S}_{\text{PAK-Z+}}$ is instantiated with the following signature scheme $(\text{Gen}, \text{Sign}^*, \text{Verify}^*)$:

$$\text{Sign}_v^*(\langle C, S, m, Y \rangle) := \text{Sign}_v(\langle C, S, m \cdot \pi, Y \rangle) \\ \text{Verify}_V^*(\langle C, S, m, Y \rangle, s) := \text{Verify}_V(\langle C, S, m \cdot \pi, Y \rangle, s).$$

Since the transformation that sends $\langle C, S, m, Y \rangle \mapsto \langle C, S, m \cdot \pi, Y \rangle$ is just a permutation, it follows that $(\text{Gen}, \text{Sign}^*, \text{Verify}^*)$ is an eu-cma signature scheme whenever $(\text{Gen}, \text{Sign}, \text{Verify})$ is.

¹Note that we do not need to instantiate H_6^* because this oracle is not used by PAK-Z+.

\mathcal{M} 's handling of \mathcal{A} 's queries. The modifier \mathcal{M} performs the following modifications to the queries of \mathcal{A} . The main goal is for \mathcal{M} to simulate all queries except for ones that are related to the U^* and U'^* guessed at the beginning of the case: these queries are passed to the underlying PAK-Z+ simulator $\mathcal{S}_{\text{PAK-Z+}}$.

RevealFactor(C, S, ℓ):

1. If $(C, S, \ell) \neq (U^*, U'^*, \ell^*)$:
Return $\text{pw}_{C,S}^\ell$.
2. If $(C, S, \ell) = (U^*, U'^*, \ell^*)$:
Reject; if this query occurs, then \mathcal{M} 's guess of U^* and U'^* at the beginning of this case was incorrect.

RevealFactorV(S, C, ℓ):

1. If $(C, S, \ell) \neq (U^*, U'^*, \ell^*)$:
Return $\overline{\text{pw}}_{C,S}^\ell$.
2. If $(C, S, \ell) = (U^*, U'^*, \ell^*)$:
Reject; if this query occurs, then \mathcal{M} 's guess of U^* and U'^* at the beginning of this case was incorrect.

Test(U, i):

1. If $U = U^*$:
Send a $\text{Test}_{\text{PAK-Z+}}(U, i)$ query to PAK-Z+ simulator $\mathcal{S}_{\text{PAK-Z+}}$ and return the result to \mathcal{A} .
2. If $U \neq U^*$:
Reject; if this query occurs, then \mathcal{M} 's guess of U^* at the beginning of this case was incorrect.

RevealSK(U, i):

1. If $U = U^*$ or $U = U'^*$:
Send a $\text{RevealSK}_{\text{PAK-Z+}}(U, i)$ query to PAK-Z+ simulator $\mathcal{S}_{\text{PAK-Z+}}$ and return the result to \mathcal{A} .
2. Otherwise:
Return sk for instance Π_i^U .

Execute(C, i, S, j):

1. If $(C, S) \neq (U^*, U'^*)$:
 \mathcal{M} performs $\text{Execute}_{\text{MFPK}}(C, i, S, j)$ with all the values it has and returns the transcript.
2. If $(C, S) = (U^*, U'^*)$:
 \mathcal{M} will use the PAK-Z+ simulator $\mathcal{S}_{\text{PAK-Z+}}$ to obtain a transcript for this query.
 - (a) Send an $\text{Execute}_{\text{PAK-Z+}}(C, i, S, j)$ query to $\mathcal{S}_{\text{PAK-Z+}}$ and receive $\langle C, m, Y, k, a, v'', s \rangle$.
 - (b) Set $\hat{m} \leftarrow m \cdot \pi$.
 - (c) Set $\hat{k}' \in_R \text{range}(H_6)$.
 - (d) Extract \hat{k} as the first component of k .
 - (e) Extract $\{a'_\ell\}_{\ell \in I_a, \ell \neq \ell^*}$ from the remaining $|I_a| - 1$ components of k .
 - (f) Compute $\{a_\ell\}_{\ell \in I_a, \ell \neq \ell^*}$.
 - (g) Set $a_{\ell^*} \leftarrow a$.
 - (h) Set $v''_{\ell^*} \leftarrow v''$.
 - (i) Compute $\{s_\ell\}_{\ell \in I_a, \ell \neq \ell^*}$.
 - (j) Set $s_{\ell^*} \leftarrow s$.
- (k) Return $\langle C, \hat{m}, Y, \hat{k}, \{a_\ell\}, \{v''_\ell\}, \hat{k}', \{s_\ell\} \rangle$ to \mathcal{A} .

Send(U, i, M):

If M is not a valid protocol message in a meaningful sequence, then reject as would be done in MFPK.

1. If $M = \langle \text{"start"}, S \rangle$ and $(U, S) \neq (U^*, U'^*)$:
Perform $\text{CLIENTACTION0}_{\text{MFPK}}$ and return $\langle U, m \rangle$.
2. If $M = \langle \text{"start"}, S \rangle$ and $(U, S) = (U^*, U'^*)$:
 - (a) Send a $\text{Send}_{\text{PAK-Z+}}(U, i, M)$ query to $\mathcal{S}_{\text{PAK-Z+}}$ and receive $\langle U, m \rangle$.
 - (b) Set $\hat{m} \leftarrow m \cdot \pi$.
 - (c) Return $\langle U, \hat{m} \rangle$.
3. If $M = \langle C, m \rangle$ and $(C, U) \neq (U^*, U'^*)$:
Perform $\text{SERVERACTION1}_{\text{MFPK}}$ and return $\langle Y, k, \{a_\ell\}, \{v''_\ell\} \rangle$.

4. If $M = \langle C, m \rangle$ and $(C, U) = (U^*, U'^*)$:
 - (a) Set $\hat{m} \leftarrow m \cdot \pi^{-1}$.
 - (b) Send a $\text{Send}_{\text{PAK-Z+}}(U, i, \langle C, \hat{m} \rangle)$ query to $\mathcal{S}_{\text{PAK-Z+}}$ and receive $\langle Y, k, a, v'' \rangle$.
 - (c) Extract \hat{k} as the first component of k .
 - (d) Extract $\{a'_\ell\}_{\ell \in I_a, \ell \neq \ell^*}$ from the remaining $|I_a| - 1$ components of k .
 - (e) Compute $\{a_\ell\}_{\ell \in I_a, \ell \neq \ell^*}$.
 - (f) Set $a_{\ell^*} \leftarrow a$.
 - (g) Set $v''_{\ell^*} \leftarrow v''$.
 - (h) Return $\langle C, \hat{m}, Y, \hat{k}, \{a_\ell\}, \{v''_\ell\} \rangle$.
5. If $M = \langle Y, k, \{a_\ell\}, \{v''_\ell\} \rangle$ and $(U, U') \neq (U^*, U'^*)$, where U' is the partner of U :
Perform $\text{CLIENTACTION2}_{\text{MFPK}}$ and return $\langle k', \{s_\ell\} \rangle$.
6. If $M = \langle Y, k, \{a_\ell\}, \{v''_\ell\} \rangle$ and $(U, U') = (U^*, U'^*)$, where U' is the partner of U :
 - (a) Verify $\{v''_\ell\}_{\ell \in I_a, \ell \neq \ell^*}$.
 - (b) Set $\hat{k}' \leftarrow k \|_{\ell \in I_a, \ell \neq \ell^*} a'_\ell$.
 - (c) Send a $\text{Send}_{\text{PAK-Z+}}(U, i, \langle Y, \hat{k}, a_{\ell^*}, v''_{\ell^*} \rangle)$ query to $\mathcal{S}_{\text{PAK-Z+}}$ and receive $\langle s \rangle$.
 - (d) Set $\hat{k}' \in_R \text{range}(H_6)$ and store.
 - (e) Compute $\{s_\ell\}_{\ell \in I_a, \ell \neq \ell^*}$.
 - (f) Set $s_{\ell^*} \leftarrow s$.
 - (g) Return $\langle \hat{k}', \{s_\ell\} \rangle$.
7. If $M = \langle k', \{s_\ell\} \rangle$ and $(U', U) \neq (U^*, U'^*)$, where U' is the partner of U :
Perform $\text{SERVERACTION3}_{\text{MFPK}}$.
8. If $M = \langle k', \{s_\ell\} \rangle$ and $(U', U) = (U^*, U'^*)$, where U' is the partner of U :
 - (a) Reject if k' is not the same as the \hat{k}' generated in Case 6 above.
 - (b) Verify $\{s_\ell\}_{\ell \in I_a, \ell \neq \ell^*}$.
 - (c) Send a $\text{Send}_{\text{PAK-Z+}}(U, i, \langle s_{\ell^*} \rangle)$ query to $\mathcal{S}_{\text{PAK-Z+}}$.

Differences from MFPK simulator. We must now analyze the differences between a true MFPK simulator and the view presented to the MFPK adversary \mathcal{A} by the modifier \mathcal{M} .

First we note that the distributions of generated passwords exactly match the MFPK specifications. Furthermore, all the generated passwords exactly match the PAK-Z+ specifications.

Next, we note that \mathcal{M} 's handling of \mathcal{A} 's queries precisely matches what an MFPK simulator would do except in a small number of cases. The messages received from and forwarded from the use of the PAK-Z+ simulator $\mathcal{S}_{\text{PAK-Z+}}$ can by inspection be seen to match what the MFPK simulator would do because $\mathcal{S}_{\text{PAK-Z+}}$ is using the specially constructed random oracles H_i^* . The differences between \mathcal{M} and what a true MFPK simulator would do are as follows:

- $\text{RevealFactor}(C, S, \ell)$ when $(C, S, \ell) = (U^*, U'^*, \ell^*)$, $\text{RevealFactorV}(S, C, \ell)$ when $(C, S, \ell) = (U^*, U'^*, \ell^*)$, and $\text{Test}(U, i)$ when $U \neq U^*$:

The modifier \mathcal{M} rejects here, while a true MFPK simulator should not. If \mathcal{M} correctly guessed U^* and U'^* at the beginning of this case, then none of these queries would occur, for if one did then the instance in which a Test query is directed to $\Pi_i^{U^*}$ would not be fresh in the ℓ^* th factor.

- $\text{Execute}(C, i, S, j)$ when $(C, S) = (U^*, U'^*)$, $\text{Send}(U, i, M)$ when $M = \langle Y, k, a, v'' \rangle$ and $(U, U') = (U^*, U'^*)$, where U' is the partner of U , and $\text{Send}(U, i, M)$ when $M = \langle k', s \rangle$ and $(U, U') = (U^*, U'^*)$, where U' is the partner of U :

The modifier \mathcal{M} generated a random value \hat{k}' for this instance instead of generating $k' = H_6(\text{sid}, \sigma, \tau_1, \dots, \tau_n)$. Since H_6 is a random oracle, this substitution is distinguishable by the adversary \mathcal{A} if and only if \mathcal{A} queries H_6 on the arguments $\text{sid}, \sigma, \tau_1, \dots, \tau_n$. But if that occurs, then \mathcal{A} must know τ_{ℓ^*} . These are the same inputs to the H_7^* oracle used to compute the session key in the PAK-Z+ simulation $\mathcal{S}_{\text{PAK-Z+}}$, so the same adversary could distinguish the output of $\text{Test}_{\text{PAK-Z+}}(U^*, i)$ received from $\mathcal{S}_{\text{PAK-Z+}}$. The latter event corresponds to the event $\text{Succ}_{\text{PAK-Z+}}^{\text{ake}}$, and so the substitution is distinguishable with probability at most $\Pr(\text{Succ}_{\text{PAK-Z+}}^{\text{ake}}|\mathcal{A})$.

Let $\text{Dist}_1|\text{GuessCS}$ be the event that the simulation \mathcal{M} is distinguishable from a real MFAK simulator from \mathcal{A} 's perspective given that the modifier correctly guessed U^* and U'^* at the beginning of this case. Then $\Pr(\text{Dist}_1|\text{GuessCS}) \leq 3\Pr(\text{Succ}_{\text{PAK-Z+}}^{\text{ake}}|\mathcal{A})$ by the argument above.

Result for case 1. Let $U^* \in \text{Clients}$, $U'^* \in \text{Servers}$ and let E_1 be the event that neither $\text{RevealFactor}_{\text{MFAK}}(U^*, U'^*, \ell^*)$ nor $\text{RevealFactorV}_{\text{MFAK}}(U'^*, U^*, \ell^*)$ occurs. The instance involving U^*, U'^* in $\mathcal{S}_{\text{PAK-Z+}}$ is fresh if and only if the corresponding instance in \mathcal{M} is fresh in the ℓ^* th factor. Thus, if event E_1 occurs and event GuessCS occurs, then, whenever \mathcal{A} wins against \mathcal{M} , \mathcal{A}^* wins against $\mathcal{S}_{\text{PAK-Z+}}$, except with probability at most $\Pr(\text{Dist}_1|\text{GuessCS})$. Therefore,

$$\Pr\left(\text{Succ}_{\mathcal{M}}^{\text{ake-fl}}(t, q_{\text{se}}, q_{\text{ex}}, q_{\text{ro}})|E_1, \text{GuessCS}\right) \leq \Pr\left(\text{Succ}_{\text{PAK-Z+}}^{\text{ake}}(t', q_{\text{se}}, q_{\text{ex}}, q'_{\text{ro}})\right),$$

where $q'_{\text{ro}} \leq n(q_{\text{ro}} + z + 6q_{\text{ex}} + 4q_{\text{se}})$, $t' \leq t + n(q_{\text{ro}} + 1)t_{\text{exp}} + q_{\text{ex}}(3t_{\text{exp}} + |I_a|t_{\text{sig}}) + q_{\text{se}}(2nt_{\text{exp}} + |I_a|t_{\text{sig}})$, and $z = \min\{q_{\text{se}} + q_{\text{ex}}, |\text{Clients}| \cdot |\text{Servers}|\}$. Moreover,

$$\left| \Pr\left(\text{Succ}_{\text{MFAK}}^{\text{ake-fl}}(t, q_{\text{se}}, q_{\text{ex}}, q_{\text{ro}})|E_1, \text{GuessCS}\right) - \Pr\left(\text{Succ}_{\mathcal{M}}^{\text{ake-fl}}(t, q_{\text{se}}, q_{\text{ex}}, q_{\text{ro}})|E_1, \text{GuessCS}\right) \right| \leq \Pr(\text{Dist}_1|\text{GuessCS}).$$

Combining these two expressions yields the following result:

Lemma A.1 *Let $U^* \in \text{Clients}$, $U'^* \in \text{Servers}$, and suppose that neither $\text{RevealFactor}_{\text{MFAK}}(U^*, U'^*, \ell^*)$ nor $\text{RevealFactorV}_{\text{MFAK}}(U'^*, U^*, \ell^*)$ occurs (which is event E_1). Then*

$$\Pr\left(\text{Succ}_{\text{MFAK}}^{\text{ake-fl}}(t, q_{\text{se}}, q_{\text{ex}}, q_{\text{ro}})|E_1, \text{GuessCS}\right) \leq 4\Pr\left(\text{Succ}_{\text{PAK-Z+}}^{\text{ake}}(t', q_{\text{se}}, q_{\text{ex}}, q'_{\text{ro}})\right),$$

where $q'_{\text{ro}} \leq n(q_{\text{ro}} + z + 6q_{\text{ex}} + 4q_{\text{se}})$, $t' \leq t + n(q_{\text{ro}} + 1)t_{\text{exp}} + q_{\text{ex}}(3t_{\text{exp}} + |I_a|t_{\text{sig}}) + q_{\text{se}}(2nt_{\text{exp}} + |I_a|t_{\text{sig}})$, and $z = \min\{q_{\text{se}} + q_{\text{ex}}, |\text{Clients}| \cdot |\text{Servers}|\}$, and a similar bound exists for $\text{Adv}_{\text{MFAK}}^{\text{s2c-fl}}$.

A.3 Remaining cases

The remaining cases are as follows:

2. Asymmetric factor uncompromised, $U^* \in \text{Servers}$: no $\text{RevealFactor}_{\text{MFAK}}(U'^*, U^*, \ell^*)$ query.
3. Symmetric factor uncompromised, $U^* \in \text{Clients}$: no $\text{RevealFactor}_{\text{MFAK}}(U^*, U'^*, \ell^*)$ query.

4. Symmetric factor uncompromised, $U^* \in \text{Servers}$: no $\text{RevealFactor}_{\text{MFAK}}(U'^*, U^*, \ell^*)$ query.

The proofs for each of these cases proceed in an analogous manner. For case 2, the modifier simulates an MFAK system to the adversary using an underlying PAK-Z+ system and assuring that the underlying system remains fresh. For cases 3 and 4, the modifier simulates an MFAK system to the adversary using an underlying PAK system and assuring that it remains fresh.

The details for these three cases appear in the full version of the paper (Stebila et al. 2009).

A.4 Overall result

By combining cases 1 and 2, we can obtain a result for instances that are fresh in the ℓ^* th factor when that factor is asymmetric, and by combining cases 3 and 4 we can obtain a result for instances that are fresh in the ℓ^* th factor when that factor is symmetric.

For the ake-fl advantage for an asymmetric factor, we have

$$\Pr\left(\text{Succ}_{\text{MFAK}}^{\text{ake-fl}}(t, q_{\text{se}}, q_{\text{ex}}, q_{\text{ro}})\right) \leq |\text{Clients}| \cdot |\text{Servers}| \cdot 8\Pr\left(\text{Succ}_{\text{PAK-Z+}}(t', q_{\text{se}}, q_{\text{ex}}, q'_{\text{ro}})\right),$$

where $t' \leq t + n(q_{\text{ro}} + 1)t_{\text{exp}} + q_{\text{ex}}(3t_{\text{exp}} + |I_a|t_{\text{sig}}) + q_{\text{se}}(3t_{\text{exp}} + |I_a|t_{\text{sig}})$, $q'_{\text{ro}} \leq n(q_{\text{ro}} + z + 6q_{\text{ex}} + 5q_{\text{se}})$, and $z = \max\{q_{\text{se}} + q_{\text{ex}}, |\text{Clients}| \cdot |\text{Servers}|\}$.

For the ake-fl advantage for a symmetric factor, we have

$$\Pr\left(\text{Succ}_{\text{MFAK}}^{\text{ake-fl}}(t, q_{\text{se}}, q_{\text{ex}}, q_{\text{ro}})\right) \leq |\text{Clients}| \cdot |\text{Servers}| \cdot 2\Pr\left(\text{Succ}_{\text{PAK}}(t'', q_{\text{se}}, q_{\text{ex}}, q''_{\text{ro}})\right),$$

where $q''_{\text{ro}} \leq n(2q_{\text{ro}} + 1 + 4z + 6q_{\text{ex}} + 5q_{\text{se}})$, $t'' \leq t + z|I_a|t_{\text{Gen}} + (q_{\text{ro}} + 1)t_{\text{exp}} + q_{\text{ex}}(3t_{\text{exp}} + |I_a|t_{\text{sig}}) + q_{\text{se}}(3t_{\text{exp}} + |I_a|t_{\text{sig}})$, and $z = \max\{q_{\text{se}} + q_{\text{ex}}, |\text{Clients}| \cdot |\text{Servers}|\}$.

In each case, a similar bound applies for $\text{Adv}_{\text{MFAK}}^{\text{ma-fl}}$. Substituting the security statements for PAK (MacKenzie 2002, Thm. 6.9) and PAK-Z+ (Gentry et al. 2005, Thm. 5.1) and simplifying the expressions, we obtain the following theorem describing the security of MFAK:

Theorem A.2 *Let G be a finite cyclic group generated by g and let S be a signature scheme with security parameter κ . Let \mathcal{A} be an adversary that runs in time t and makes at most q_{se} and q_{ex} queries of type Send and Execute , respectively, and at most q_{ro} queries to the random oracle. Let $b_{\text{co}} = 1$ if \mathcal{A} makes a $\text{RevealFactorV}(\cdot, \cdot, \ell)$ query to a server, and 0 otherwise. Then MFAK is a secure multi-factor password-authenticated key exchange protocol, with*

$$\text{Adv}_{\text{MFAK}}^{\text{ake-fl}}(\mathcal{A}) \leq \begin{cases} \frac{16\delta((1-b_{\text{co}})q_{\text{se}} + b_{\text{co}}q_{\text{ro}})}{|\text{Passwords}^\ell|} + \epsilon, & \text{if the } \ell\text{th factor is symmetric,} \\ \frac{4\delta q_{\text{se}}}{|\text{Passwords}^\ell|} + \epsilon, & \text{if the } \ell\text{th factor is asymmetric,} \end{cases}$$

where $\epsilon = 8q_{\text{se}}\text{Adv}_{G,g}^{\text{cdh}}(t', q'_{\text{ro}}) + 6q_{\text{se}}\text{Succ}_{S,\kappa}^{\text{eu-cma}}(t', q_{\text{se}}) + \frac{5(q_{\text{se}} + q_{\text{ex}})(q_{\text{ro}} + q_{\text{se}} + q_{\text{ex}})}{|G|}$ and $\delta = |\text{Clients}| \cdot |\text{Servers}|$, for $t' = t + (z|I_a| + 8(q'_{\text{ro}})^2 + |I_a|q_{\text{se}} + |I_a|q_{\text{ex}})t_{\text{exp}}$, $q'_{\text{ro}} = n(2q_{\text{ro}} + 4z + 6q_{\text{ex}} + 5q_{\text{se}})$, and $z = \max\{q_{\text{se}} + q_{\text{ex}}, |\text{Clients}| \cdot |\text{Servers}|\}$; a similar bound exists for $\text{Adv}_{\text{MFAK}}^{\text{ma-fl}}(\mathcal{A})$.

An Analysis of the RC4 Family of Stream Ciphers against Algebraic Attacks

Kenneth Koon-Ho Wong¹Gary Carter¹Ed Dawson¹

¹Information Security Institute
Queensland University of Technology
Brisbane, Australia

Email: {kk.wong,g.carter,e.dawson}@qut.edu.au

Abstract

To date, most applications of algebraic analysis and attacks on stream ciphers are on those based on linear feedback shift registers (LFSRs). In this paper, we extend algebraic analysis to non-LFSR based stream ciphers. Specifically, we perform an algebraic analysis on the RC4 family of stream ciphers, an example of stream ciphers based on dynamic tables, and investigate its implications to potential algebraic attacks on the cipher. This is, to our knowledge, the first paper that evaluates the security of RC4 against algebraic attacks through providing a full set of equations that describe the complex word manipulations in the system. For an arbitrary word size, we derive algebraic representations for the three main operations used in RC4, namely state extraction, word addition and state permutation. Equations relating the internal states and keystream of RC4 are then obtained from each component of the cipher based on these algebraic representations, and analysed in terms of their contributions to the security of RC4 against algebraic attacks. Interestingly, it is shown that each of the three main operations contained in the components has its own unique algebraic properties, and when their respective equations are combined, the resulting system becomes infeasible to solve. This results in a high level of security being achieved by RC4 against algebraic attacks. On the other hand, the removal of an operation from the cipher could compromise this security. Experiments on reduced versions of RC4 have been performed, which confirms the validity of our algebraic analysis and the conclusion that the full RC4 stream cipher seems to be immune to algebraic attacks at present.

1 Introduction

Algebraic attacks on stream ciphers, introduced by Courtois & Meier (2003) and Courtois (2004), are attacks in which the keystream is used to construct a system of multivariate polynomial equations with the keys or initial states of the stream ciphers as variables. Solving the system of equations amounts to recovering the keys or initial states. This method of attack was initially applied to block ciphers and public key cryptosystems (Courtois 2001, Courtois & Pieprzyk 2002). Algebraic analysis has been demonstrated at times to be a very useful tool for stream ciphers based on linear feedback shift registers (LFSRs). Several well known LFSR-based stream ciphers have fallen to

algebraic attacks (Al-Hinai et al. 2006, Armknecht & Krause 2003, Cho & Pieprzyk 2004, Courtois 2004, 2003, Courtois & Meier 2003, Wong et al. 2006). It is therefore appropriate to extend algebraic analysis to other well-known ciphers that are not based on LFSRs, in order to evaluate the possibility of successful algebraic attacks on them. This is the primary aim of this paper.

In this paper, we perform an algebraic analysis of the RC4 family of stream ciphers (Schneier 1996), which is a word-based stream cipher based on dynamic tables. We show how valid algebraic relations among the internal states of the cipher are obtained, in order to form a full system of equations describing the cipher. We then investigate the equations and evaluate the resistance of RC4 to algebraic attacks. To the best of our knowledge, this is the first paper on a full algebraic analysis of RC4. The types and number of equations generated from the cipher are discussed, and can be used as a guide for the level of security of RC4 against algebraic attacks, both at present and for future reference, as solution methods for large equation systems may become more efficient over time. The methods of analysis and results presented here could also be extended to RC4 variants, such as RC4A (Paul & Preneel 2004) and VMPC (Zoltak 2004).

To date, RC4 remains a widely used stream cipher in network and wireless applications, as well as in many commercial products. It is a word-based stream cipher, whose simple and elegant design by Rivest in 1987 had been kept secret until 1994. After the specification of the RC4 was revealed, the cipher became the target for cryptanalysis. The first published cryptanalysis of RC4 was by Golić (1997), followed by a number of interesting ones (Knudsen et al. 1998, Fluhrer & McGrew 2000, Mantin & Shamir 2001, Paul & Preneel 2003, 2004). Weaknesses identified in the RC4 cipher have motivated the proposal of several strengthened versions of RC4, such as RC4A (Paul & Preneel 2004). Other researchers were inspired by the design of the cipher and proposed stream ciphers based on the design of RC4 such as the 32 and 64-bit RC4 (Gong et al. 2005) and VMPC (Zoltak 2004). However, distinguishing attacks have since been shown to be effective on both the original and strengthened proposals of RC4 and on new RC4 variants (Maximov 2005, Tsunoo et al. 2005). Cryptanalysis of RC4 remains an active topic with recent developments in improved state and key recovery attacks (Biham & Carmeli 2008, Maximov & Khovratovich 2008, Basu et al. 2009).

In order to provide an algebraic analysis of the RC4 stream cipher, we first show how algebraic relationships can be obtained for the operations within the cipher. The three main operations used in RC4, namely word addition, state extraction and state permutation will be analysed. Algebraic representations

Copyright ©2010, Australian Computer Society, Inc. This paper appeared at the Australasian Information Security Conference (AISC2010), Brisbane, Australia. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 103, Colin Boyd and Willy Susilo, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

for each of these three operations will be derived, and some of their properties will be discussed. Then, we convert these operations into valid expressions relating the internal states and keystream, and construct a system of polynomial equations from them. The solution of the system would amount to the recovery of the initial states. We analyse how each of the operations contribute to the number of equations generated, their respective degrees and form. We arrive at an observation that these three main operations contribute uniquely to the system of equations derived from the RC4 stream cipher, which give a high level of resistance to algebraic attacks only when combined into one cipher.

The paper is organised as follows. Section 2 provides a description of RC4. In section 3, we show how algebraic relations for the operations involved in RC4 can be obtained. In section 4, we construct equations that relate the initial states to the keystream for RC4. Section 5 provides a summary and analysis of the results, which are then used to determine the security of RC4 against algebraic attacks. Section 6 gives an account on actual attempts of algebraic attacks on the cipher using the methods presented. Section 7 concludes the paper.

2 Description of RC4

The RC4 family of stream ciphers is a word-based stream cipher, which has a very large internal state space compared to the key size. For a word size of n bits, it consists of a permutation table of 2^n words and two pointers i, j of one word each. The total internal state space of RC4 is therefore of size $\log_2(2^n!(2^n)^2)$ bits. For the common implementation with $n = 8$, this is approximately 1700 bits. Two algorithms govern the RC4 stream cipher, namely the key scheduling algorithm (KSA) and the pseudo-random generation algorithm (PRGA). In the KSA, a secret key k is used to load and mix the internal states S_i of the register S , resulting in S having some permutation of the 2^n possible n -bit words. The PRGA then proceeds to generate keystream using the states obtained from the KSA. The KSA and PRGA for RC4 are shown in Figure 1. The operations described in the pseudocode are wordwise and the keystream output is denoted by z .

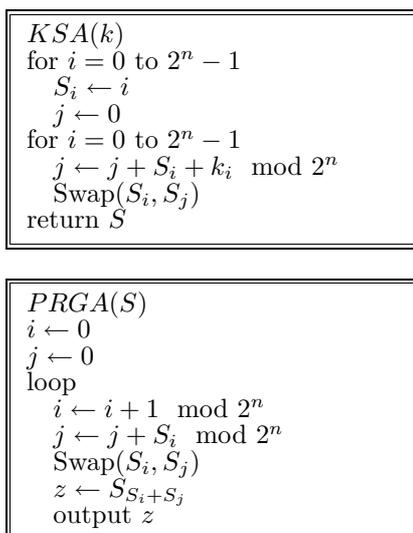


Figure 1: The KSA of RC4 (top), The PRNG of RC4 (bottom).

During the KSA, the identity permutation $(0, 1, \dots, 2^n - 1)$ is loaded into the register S . The

secret key k is then used to initialize S to a random permutation by shuffling the words in S according to the KSA. Once the KSA is complete, the cipher is ready for keystream generation. The PRGA is used to produce pseudo-random keystream words derived from the permutations in S . Each iteration of the PRGA loop produces one output word z , which constitutes n bits of keystream. In this paper, we consider the cipher from the start of the PRGA, to arrive at an initial state recovery algebraic analysis and attack, where the initial state is the permutation in S at that time.

3 Algebraic Analysis of RC4

The RC4 stream cipher of word size n uses one register S of length $2^n - 1$ with an n -bit word representing each state of S . Commonly, RC4 is used with $n = 8$. However, we will present an algebraic analysis that is applicable for arbitrary n . Before key initialisation, the states of S are set such that

$$S = (0, 1, \dots, 2^n - 1).$$

The cipher then initialises according to the KSA, which depends on the key k used. After the KSA, the register S arrives at its initial state S^0 , such that

$$S^0 = (x_0, x_1, \dots, x_{2^n-1}),$$

where x_i are n -bit words represented as elements in $\mathbb{Z}/2^n\mathbb{Z}$. Throughout this paper, we utilise the canonical isomorphism between the residue class ring $\mathbb{Z}/2^n\mathbb{Z}$ representing integers modulo 2^n and the product ring \mathbb{F}_2^n representing the bit strings of those integers, so that all equations describing RC4 are generated as polynomials with coefficients in \mathbb{F}_2 , with the word variables in $\mathbb{Z}/2^n\mathbb{Z}$ also split into bit variables in \mathbb{F}_2 . From here onwards, for $u \in \mathbb{Z}/2^n\mathbb{Z}$ and $0 \leq b \leq n-1$, let $u_{(b)} \in \mathbb{F}_2$ be the b -th least significant bit (LSB) of $u \in \mathbb{Z}/2^n\mathbb{Z}$, and $\mathbf{u} = (u_{(0)}, u_{(1)}, \dots, u_{(n-1)}) \in \mathbb{F}_2^n$ be the binary vector representing u . Additionally, to denote the b -th least significant bit of a state k of register S , we use the notation $S_{k,(b)}$. While the ring $\mathbb{Z}/2^n\mathbb{Z}$ or the extension field \mathbb{F}_{2^n} are possible candidates for this algebraic analysis, we have chosen not to use them due to their associated cumbersome representations and manipulations, compared to the much simpler arithmetic in \mathbb{F}_2 . In addition, there exists some technical difficulties for using these structures, as will be explained in the successive sections. We now derive the algebraic expressions of operations involved in RC4.

3.1 State Extraction

The value of one state S_i in the register S at position i is at times needed. As i is considered unknown, it is not possible to simply represent it as S_i in an algebraic analysis. Instead, we must derive an algebraic expression that extracts the correct word in S for any possible value of i . This state extraction operation is algebraically equivalent to evaluating the piecewise expression

$$S_i = \begin{cases} S_0, & i = 0 \\ S_1, & i = 1 \\ \vdots & \\ S_{2^n-1}, & i = 2^n - 1, \end{cases}$$

where S and i can be unknown. This expression can be split into n independent expressions, one for each

bit $S_{i,(b)}$ of S_i , where $0 \leq b \leq 1$. Each of these expressions is dependent on all bits $i_{(0)}, i_{(1)}, \dots, i_{(n-1)}$ of i , such that

$$S_{i,(b)} = \begin{cases} S_{0,(b)}, & \mathbf{i} = (0, 0, \dots, 0) \\ S_{1,(b)}, & \mathbf{i} = (0, 0, \dots, 1) \\ \vdots & \\ S_{2^n-1,(b)}, & \mathbf{i} = (1, 1, \dots, 1) \end{cases}, \quad 0 \leq b \leq n-1.$$

Since the variables are in \mathbb{F}_2 , the piecewise expression for each bit can then be arranged analogously as a boolean expression and written as

$$S_{i,(b)} = \sum_{u=0}^{2^n-1} \left(S_{u,(b)} \prod_{b=0}^{n-1} (i_{(b)} + u_{(b)} + 1) \right).$$

Here, the expression in the brackets is one when $i = u$ and zero otherwise, resulting in the correct bits of S_i being evaluated. If we instead use the residue class ring $\mathbb{Z}/2^n\mathbb{Z}$ to describe state extraction operation, it is not possible to convert the piecewise expression into a single algebraic expression, since we are not able to find a function that gives a nonzero value for the correct index i and give zeros for the other indices. Therefore, this operation has prevented us from using word-based algebraic analysis in $\mathbb{Z}/2^n\mathbb{Z}$. The above expression is ordered by S_u with polynomials in $i_{(b)}$ as coefficients. This can be rewritten to order by degrees of monomials in $i_{(b)}$ with S_k as coefficients as

$$S_{i,(b)} = \sum_{e=0}^{2^n-1} \left(\prod_{f=0}^{n-1} i_{(f)}^{e_{(f)}} \left(\sum_{k=1}^{2^n-1} S_{k,(b)} \left(\prod_{g=0}^{n-1} e_{(g)} (i_{(g)} + 1) + 1 \right) \right) \right).$$

Consider a bit position $S_{i,(b)}$ throughout the entire register. Due to the fact that S is a permutation of its initial states, the values of that position must contain an equal number of zeros and ones, which means that

$$\sum_{k=0}^{2^n-1} S_{k,(b)} = 0, \quad 0 \leq b \leq n-1.$$

Therefore, the expressions for state extraction can be reduced to

$$S_{i,(b)} = \sum_{e=0}^{2^n-2} \left(\prod_{f=0}^{n-1} i_{(f)}^{e_{(f)}} \left(\sum_{k=1}^{2^n-1} S_{k,(b)} \left(\prod_{g=0}^{n-1} e_{(g)} (i_{(g)} + 1) + 1 \right) \right) \right).$$

This removes the degree $n+1$ terms in the expression, and we are left with expressions of maximum degree n for the state extraction of S_i . For example, with $n = 2$ the expression is

$$\begin{aligned} S_{i,(b)} &= i_{(0)}i_{(1)}S_{0,(b)} + i_{(0)}i_{(1)}S_{1,(b)} \\ &\quad + i_{(0)}i_{(1)}S_{2,(b)} + i_{(0)}i_{(1)}S_{3,(b)} \\ &\quad + i_{(0)}S_{0,(b)} + i_{(0)}S_{3,(b)} \\ &\quad + i_{(1)}S_{1,(b)} + i_{(1)}S_{3,(b)} + S_{3,(b)} \\ &= i_{(0)}S_{0,(b)} + i_{(0)}S_{3,(b)} \\ &\quad + i_{(1)}S_{1,(b)} + i_{(1)}S_{3,(b)} + S_{3,(b)}, \end{aligned}$$

since $S_0 + S_1 + S_2 + S_3 = 0$. The expression for $S_{i,(b)}$ is therefore of degree 2.

3.2 Word Addition

The addition operation in RC4 is defined as word addition modulo 2^n , which we denote as $+_{2^n}$, as opposed to $+$, which is understood as addition modulo 2 throughout this paper, unless otherwise indicated. To obtain the equivalent operations using bit values in \mathbb{F}_2 , we use the additive group isomorphism between $\mathbb{Z}/2^n\mathbb{Z}$ and \mathbb{F}_2^n induced by addition on the binary digits of integers modulo 2^n . Let $u, v, w \in \mathbb{Z}/2^n\mathbb{Z}$ such that

$$u +_{2^n} v = w.$$

The equivalent addition over the binary digits in \mathbb{F}_2^n is defined as

$$\begin{aligned} \mathbf{u} + \mathbf{v} &= (u_{(0)}, u_{(1)}, \dots, u_{(n-1)}) \\ &\quad + (v_{(0)}, v_{(1)}, \dots, v_{(n-1)}) \\ &= (w_{(0)}, w_{(1)}, \dots, w_{(n-1)}) \\ &= \mathbf{w}, \end{aligned}$$

where $w_{(b)}$ satisfies

$$w_{(b)} = \sum_{k=0}^{b-1} \left(u_{(k)}v_{(k)} \prod_{l=k+1}^{b-1} (u_{(l)} + v_{(l)}) \right) + u_{(b)} + v_{(b)}, \quad 0 \leq b \leq n-1.$$

With this definition, we obtain have the additive group isomorphism

$$w_{(b)} = (u +_{2^n} v)_{(b)} = u_{(b)} + v_{(b)}.$$

This amounts to degree $n+1$ expressions in the bit variables for addition. It can be seen that these expressions are independent of n . If we were to obtain the algebraic expressions for word addition in the extension field \mathbb{F}_{2^n} , we would have to extract the individual carry bits using, for example, trace maps. This procedure is quite complex and will most likely yield high degree equations. Therefore, we have decided against using \mathbb{F}_{2^n} for this algebraic analysis. The first few expressions in increasing bit significance are as follows.

$$\begin{aligned} w_{(0)} &= u_{(0)} + v_{(0)}, \\ w_{(1)} &= u_{(0)}v_{(0)} + u_{(1)} + v_{(1)}, \\ w_{(2)} &= u_{(1)}u_{(0)}v_{(0)} + v_{(1)}u_{(0)}v_{(0)} \\ &\quad + u_{(1)}v_{(1)} + u_{(2)} + v_{(2)}, \\ w_{(3)} &= u_{(1)}u_{(2)}u_{(0)}v_{(0)} + u_{(1)}v_{(2)}u_{(0)}v_{(0)}, \\ &\quad + u_{(2)}v_{(1)}u_{(0)}v_{(0)} + v_{(1)}v_{(2)}u_{(0)}v_{(0)}, \\ &\quad + u_{(1)}u_{(2)}v_{(1)} + u_{(1)}v_{(1)}v_{(2)}, \\ &\quad + u_{(2)}v_{(2)} + u_{(3)} + v_{(3)}. \end{aligned}$$

3.3 State Permutation

Swapping states i, j in S can be algebraically described as the action of a permutation matrix \mathbf{M} on S . Given i, j , the entries $m_{r,s}$ of \mathbf{M} are constructed as follows.

- A diagonal entry $m_{r,r}$ is set if $i = j$ or both $i \neq r$ and $j \neq r$,
- An off-diagonal entry $m_{r,s}$ is set if $\{i, j\} = \{r, s\}$.

Using the above rules, the appropriate boolean function used by each entry $m_{r,s}$ of \mathbf{M} in the bits $i_{(b)}, j_{(b)}$

could be created, in a similar way as the state extraction operation. The diagonal entries can be expressed as

$$\begin{aligned}
 m_{r,r} = & \prod_{b=0}^{n-1} (i_{(b)} + j_{(b)} + 1) + \left(1 + \prod_{b=0}^{n-1} (i_{(b)} + r_{(b)} + 1) \right) \\
 & \times \left(1 + \prod_{b=0}^{n-1} (j_{(b)} + r_{(b)} + 1) \right) \\
 & + \prod_{b=0}^{n-1} (i_{(b)} + j_{(b)} + 1) \left(1 + \prod_{b=0}^{n-1} (i_{(b)} + r_{(b)} + 1) \right) \\
 & \times \left(1 + \prod_{b=0}^{n-1} (j_{(b)} + r_{(b)} + 1) \right).
 \end{aligned}$$

The off-diagonal entries can be expressed as

$$\begin{aligned}
 m_{r,s} = & \prod_{b=0}^{n-1} (i_{(b)} + r_{(b)} + 1) \prod_{b=0}^{n-1} (j_{(b)} + s_{(b)} + 1) \\
 & + \prod_{b=0}^{n-1} (i_{(b)} + s_{(b)} + 1) \prod_{b=0}^{n-1} (j_{(b)} + r_{(b)} + 1) = m_{s,r}.
 \end{aligned}$$

The resulting matrix \mathbf{M} is always symmetric. For example, the permutation matrix with $n = 2$ is

$$\mathbf{M} = \begin{pmatrix} m_{0,0} & m_{0,1} & m_{0,2} & m_{0,3} \\ m_{0,1} & m_{1,1} & m_{1,2} & m_{1,3} \\ m_{0,2} & m_{1,2} & m_{2,2} & m_{2,3} \\ m_{0,3} & m_{1,3} & m_{2,2} & m_{3,3} \end{pmatrix},$$

with entries

$$\begin{aligned}
 m_{0,0} = & i_{(0)}i_{(1)} + j_{(0)}j_{(1)} \\
 & + i_{(0)} + i_{(1)} + j_{(0)} + j_{(1)} + 1, \\
 m_{0,1} = & i_{(0)}i_{(1)}j_{(1)} + i_{(1)}j_{(0)}j_{(1)} + i_{(0)}i_{(1)} + i_{(0)}j_{(1)} \\
 & + i_{(1)}j_{(0)} + j_{(0)}j_{(1)} + i_{(0)} + j_{(0)}, \\
 m_{1,1} = & i_{(0)}i_{(1)} + j_{(0)}j_{(1)} + i_{(0)} + j_{(0)} + 1, \\
 m_{0,2} = & i_{(0)}i_{(1)}j_{(0)} + i_{(0)}j_{(0)}j_{(1)} + i_{(0)}i_{(1)} + i_{(0)}j_{(1)} \\
 & + i_{(1)}j_{(0)} + j_{(0)}j_{(1)} + i_{(1)} + j_{(1)}, \\
 m_{1,2} = & i_{(0)}i_{(1)}j_{(0)} + i_{(0)}i_{(1)}j_{(1)} + i_{(0)}j_{(0)}j_{(1)} \\
 & + i_{(1)}j_{(0)}j_{(1)} + i_{(0)}j_{(1)} + i_{(1)}j_{(0)}, \\
 m_{2,2} = & i_{(0)}i_{(1)} + j_{(0)}j_{(1)} + i_{(1)} + j_{(1)} + 1, \\
 m_{0,3} = & i_{(0)}i_{(1)}j_{(0)} + i_{(0)}i_{(1)}j_{(1)} + i_{(0)}j_{(0)}j_{(1)} \\
 & + i_{(1)}j_{(0)}j_{(1)} + i_{(0)}i_{(1)} + j_{(0)}j_{(1)}, \\
 m_{1,3} = & i_{(0)}i_{(1)}j_{(0)} + i_{(0)}j_{(0)}j_{(1)}, \\
 m_{2,3} = & i_{(0)}i_{(1)}j_{(1)} + i_{(1)}j_{(0)}j_{(1)}, \\
 m_{3,3} = & i_{(0)}i_{(1)} + j_{(0)}j_{(1)} + 1.
 \end{aligned}$$

The entries of \mathbf{M} have maximum degree $n + 1$. In the following section, we will show how these algebraic representations can be used to describe RC4 in an algebraic attack.

4 Equation Generation

In this section, we present techniques of equation generation for RC4. By introducing variables at each step of the algorithm, we can keep the equations generated to be of relatively low degrees, which could reduce solution time of the final system of equations. Where possible, we also show low degree multiples of the

equations generated, which could be used to simplify the system further (Courtois 2003). Let S^t, i^t, j^t be the values of S, i, j respectively at the end of clock t , where $t \geq 0$. We then have S^0, i^0, j^0 representing the initial states of S, i, j respectively. The relations among these internal states of RC4 and the keystream can then be expressed as follows.

$$\begin{aligned}
 i^t = & i^{t-1} +_{2^n} 1 && \text{(pointer increment),} \\
 j^t = & j^{t-1} +_{2^n} S_i^{t-1} && \text{(pointer addition),} \\
 S^t = & \mathbf{M}S^{t-1} && \text{(state permutation),} \\
 z^t = & S_{S_i^t +_{2^n} S_j^t} && \text{(keystream generation).}
 \end{aligned}$$

Each operation shown above will be algebraically analysed below.

4.1 Pointer Increment

In the first step, i is incremented by one. This addition is represented by

$$\begin{aligned}
 i_{(0)}^t = & i_{(0)}^{t-1} + 1, \\
 i_{(b)}^t = & i_{(b)}^{t-1} + \prod_{k=0}^{b-1} i_{(k)}^{t-1}, \quad 1 \leq b \leq n-1.
 \end{aligned}$$

Since it is known that $i^0 = 0$, the values of i^t are actually known for all $t \geq 0$. Therefore, no equations are needed to describe this step.

4.2 Pointer Addition

The contents of S_i are then extracted, which gives

$$S_{i,(b)}^t = \sum_{k=0}^{2^n-1} \left(S_{k,(b)}^t \prod_{l=0}^{n-1} (i_{(l)}^t + k_{(l)} + 1) \right), \quad 0 \leq b \leq n-1.$$

From the analysis in section 3.1, this can be expressed as

$$S_{i,(b)}^t = \sum_{e=0}^{2^n-1} \left(\prod_{f=0}^{n-1} i_{(f)}^{e_{(f)}} \left(\sum_{k=1}^{2^n-1} S_{i,(k)}^t \left(\prod_{g=0}^{n-1} e_{(g)}(i_{(g)} + 1) + 1 \right) \right) \right).$$

The addition for j is then given as follows.

$$\begin{aligned}
 j_{(0)}^t = & j_{(0)}^{t-1} + S_{i,(0)}^t, \\
 j_{(b)}^t = & \sum_{k=0}^{b-1} \left(j_{(k)}^{t-1} S_{i,(k)}^t \prod_{l=k+1}^{b-1} (j_{(l)}^{t-1} + S_{i,(l)}^t) \right) \\
 & + j_{(b)}^{t-1} + S_{i,(b)}^t, \quad 1 \leq b \leq n-1.
 \end{aligned}$$

This gives n equations of maximum degree n with n variables representing $j_{(0)}, j_{(1)}, \dots, j_{(n-1)}$ introduced at each clock. It is possible to move all terms to the left hand side and multiply the resulting expression by $(j_{(b-1)}^{t-1} + 1)(S_{i,(b-1)}^{t-1} + 1)$ to obtain

$$(j_{(b-1)}^{t-1} + 1)(S_{i,(b-1)}^{t-1} + 1)(j_{(b)}^{t-1} + S_{i,(b)}^t + j_{(b)}^t) = 0.$$

This would yield equations of maximum degree 3 for the this word addition operation.

4.3 State Permutation

The new pointers i^t, j^t are then used for state permutation in register S . Similar to the derivation before, the diagonal entries of the permutation matrix \mathbf{M} are given by

$$m_{r,r}^t = \prod_{b=0}^{n-1} (i_{(b)}^t + j_{(b)}^t + 1) \left(1 + \prod_{b=0}^{n-1} (i_{(b)}^t + r_{(b)}^t + 1) \right) \\ \times \left(1 + \prod_{b=0}^{n-1} (j_{(b)}^t + r_{(b)}^t + 1) \right) \\ + \prod_{b=0}^{n-1} (i_{(b)}^t + j_{(b)}^t + 1) \left(1 + \prod_{b=0}^{n-1} (i_{(b)}^t + r_{(b)}^t + 1) \right) \\ \times \left(1 + \prod_{b=0}^{n-1} (j_{(b)}^t + r_{(b)}^t + 1) \right).$$

The off-diagonal entries are given by

$$m_{r,s}^t = \prod_{b=0}^{n-1} (i_{(b)}^t + r_{(b)}^t + 1) \prod_{b=0}^{n-1} (j_{(b)}^t + s_{(b)}^t + 1) \\ + \prod_{b=0}^{n-1} (i_{(b)}^t + s_{(b)}^t + 1) \prod_{b=0}^{n-1} (j_{(b)}^t + r_{(b)}^t + 1) = m_{s,r}^t.$$

It can be observed that multiplying each entry $m_{r,s}$ of the permutation matrix \mathbf{M} by

$$\sigma_{r,s}^t = \left(\sum_{b=0}^{n-1} i_{(b)}^t + \sum_{b=0}^{n-1} r_{(b)}^t \right) \left(\sum_{b=0}^{n-1} j_{(b)}^t + \sum_{b=0}^{n-1} s_{(b)}^t \right).$$

gives a low degree multiple of the original expression of the entry, which is of maximum degree 3. In order to incorporate σ into our equations, we can relabel and multiply each entry of \mathbf{M} to obtain the degree 3 expressions $\sigma m_{r,s}$. The number of equations introduced as a result would be $2^{n-1}(2^n - 1)$, since \mathbf{M} is symmetric. An additional $2^n - 1$ linear expressions are required for the row sums of the matrix i.e. the new states of register S . This method would be quite uneconomical for an algebraic attack. Alternatively, let

$$\mathbf{M} = \begin{pmatrix} m_{0,0} & m_{0,1} & \cdots & m_{0,2^n-1} \\ m_{0,1} & m_{1,1} & \cdots & m_{1,2^n-1} \\ \vdots & \vdots & \ddots & \vdots \\ m_{0,2^n-1} & m_{1,2^n-1} & \cdots & m_{2^n-1,2^n-1} \end{pmatrix}, \\ \mathbf{S}^t = \begin{pmatrix} S_{0,(0)}^t & S_{0,(1)}^t & \cdots & S_{0,(n-1)}^t \\ S_{1,(0)}^t & S_{1,(1)}^t & \cdots & S_{1,(n-1)}^t \\ \vdots & \vdots & \ddots & \vdots \\ S_{2^n-1,(0)}^t & S_{2^n-1,(1)}^t & \cdots & S_{2^n-1,(n-1)}^t \end{pmatrix}.$$

The permutation action can then be described as the multiplication

$$\mathbf{S}^t = \mathbf{M}\mathbf{S}^{t-1}.$$

Hence, we have

$$S_{i,(b)}^t = \sum_{k=0}^{2^n-1} m_{b,k} S_{k,(b)}^{t-1}, \quad 0 \leq i \leq 2^n - 1,$$

where the $m_{u,v}$ are the matrix entries of \mathbf{M} . An examination into the equations generated reveals that the equations can be simplified. Since i^t is known, the

off-diagonal ones can only appear at rows i^t and column i^t in \mathbf{M} , which means that the other off-diagonal entries of \mathbf{M} are known to be zero. In particular, for each bit $0 \leq b \leq n - 1$, we have

$$S_{r,(b)}^t = \begin{cases} \sum_{k=0}^{2^n-1} m_{b,k} S_{k,(b)}^{t-1}, & r = i \\ m_{b,i} S_{i,(b)}^{t-1} + m_{b,r} S_{r,(b)}^{t-1}, & r \neq i. \end{cases}$$

These equations are of degree $n + 1$. When $r = i$, the expression $S_{r,(b)}^t$ is of the form

$$S_{r,(b)}^t = \prod_{b=0}^n j_{(b)}^t \sum_{k=0}^{2^n-1} S_{k,(b)}^t + a = a,$$

where a is of degree $n - 1$. As discussed in Section 3.1, the first term is zero, and we equations are then of degree $n - 1$. Overall, the state permutation operation results in $n2^n$ equations of maximum degree $n + 1$ with $n2^n$ variables representing values in the permutation matrices \mathbf{M} introduced at each clock. When $r \neq i$, it is possible to move all terms to the left hand side and obtain the equation

$$(S_{r,(b)}^t + m_{b,i} S_{i,(b)}^{t-1} + m_{b,r} S_{r,(b)}^{t-1})(S_{i,(b)}^{t-1} + S_{r,(b)}^{t-1} + 1) = 0.$$

This gives equations of degree 3. This alternative approach avoids relabelling of matrix entries, at the cost of having more high degree equations in the system.

4.4 Keystream Generation

Finally, state extraction is used twice to obtain a keystream word at each clock. Let r^t be the index of the state from which the keystream output is to be taken. Then,

$$r_{(b)}^t = S_{i,(b)}^t + S_{j,(b)}^t = \sum_{k=0}^{2^n-1} \left(S_{k,(b)} \prod_{b=0}^{n-1} (i_{(b)}^t + k_{(b)}) \right) \\ + \sum_{k=0}^{2^n-1} \left(S_{k,(b)} \prod_{b=0}^{n-1} (j_{(b)}^t + k_{(b)}) \right).$$

The keystream z^t is then given by extracting state r^t of register S . Thus,

$$z_{(b)}^t = S_{r^t,(b)} = \sum_{k=0}^{2^n-1} \left(S_{k,(b)} \prod_{b=0}^{n-1} (r_{(b)}^t + u_{(b)}) \right).$$

This amounts to $2n$ equations of degree n with n variables representing $r_{(0)}^t, r_{(1)}^t, \dots, r_{(n-1)}^t$ introduced at each clock, since z^t is assumed to be known.

4.5 Additional Equations

As discussed in Section 3.1, each bit position of the register S must sum to zero, that is,

$$\sum_{k=0}^{2^n-1} S_{k,(b)} = 0, \quad 0 \leq b \leq n - 1.$$

This provides n additional linear equations at each clock with no extra variables introduced.

Operation	e	v	d_1	d_2
Pointer Increment for i	0	0	0	0
Pointer Addition for j	n	n	3	n
State Permutation	$n2^n$	$n2^n$	3	$n + 1$
Keystream Generation	$2n$	$2n$	n	n
Additional Equations	n	0	1	1

Table 1: Summary of Equations Generated for RC4. For each clock, e is the number of equations generated, v is the number of variables introduced to the system, and d_1, d_2 are the maximum degrees of the equations with and without introducing low degree multiples, respectively.

5 Discussion

Based on the results from the previous sections, the number of equations generated at each clock for each operation is summarised in Table 1. From these results, we present an analysis of the RC4 cipher against algebraic attacks.

5.1 RC4 as an Algebraic Cipher

From the cipher description, one would normally expect the high nonlinearity of RC4 to arise from the state permutation operation. However, if low degree multiples are taken into account, it has been shown from the above analysis that the high nonlinearity is caused by state extraction, since there seems to be no low degree equations that can describe the operation in terms of the internal states. The state permutation, on the other hand, makes a primary contribution to the number of equations generated from the cipher. This is because it is the only operation that affects every state of the register, rather than just certain words or bits in the cipher. Not apparent from Table 1 is the important role of word addition with its effect of the carry bits. This operation relates all bits in each word of the internal states, and yields a system of equations that cannot be separated into smaller ones. If word addition is not present, the system could be split into n independent ones for each bit position, which can be solved independently. This can dramatically reduce the time complexity of an algebraic attack. It is quite interesting to see that each of the three main operations involved in the RC4 stream cipher has its own role in providing the overall security of the cipher when realised from an algebraic point of view, particularly since algebraic attacks had not yet appeared in their current form at the time when RC4 was designed. Together, the three operations in RC4 yield a strong algebraic system, and forms the basis of the resistance of the cipher against algebraic attacks.

5.2 Implications for RC4-Like Ciphers

Similar observations would be expected to arise if the same method of algebraic analysis is used on RC4 variants such as RC4A (Paul & Preneel 2004) and VMPC (Zoltak 2004), due to the similarities of their components. We also note that the algebraic properties of the three operations discussed above could form a sound set of design criteria for potential ciphers of this type. Specifically, in order to provide resistance to algebraic attacks, the cipher should contain components whose operations consist of some that translate to a large number of equations, some to equations of high degree, and some to equations that would make the whole system inseparable. An important point to note from the algebraic analysis of RC4 is that these algebraic properties need not come

from the same component. This is useful because components that satisfy more properties may contain operations that are more complicated and hence less efficient. Security against algebraic attacks need not be sacrificed for efficiency if careful tradeoffs are made between the number of components and their algebraic properties.

5.3 Algebraic Attacks on RC4

From the equation analysis, we obtain $n2^n + 3n$ equations in $n2^n + 3n$ variables at each clock of the cipher. With a register size of 2^n words, there are $n2^n$ additional initial state variables, but there are also $n(2^n + 1)$ additional equations. Since each clock gives n bits of output, we require keystream from at least 2^n clocks before a unique solution could be obtained. In fact, if no low degree multiples are used, we only require this amount of clocks to generate an overdefined system with a unique solution. In total, there would be $2^n(n2^n + 3n)$ equations in $2^n(n2^n + 3n + n(2^n + 1))$ variables. For the common 8-bit RC4 cipher, this gives a system of 534536 equations of maximum degree 9 in 532480 variables. These equations are very sparse, as each variable is only related to those at the immediately preceding, current, and immediately succeeding clocks. If low degree multiples are used, the numbers of equations and variables may rise if dependencies are found among the equations. More information and experimental results on solving equations without low degree multiples will be presented in Section 6.

The purpose of this paper is not to propose an algebraic attack, but to consider the impact of applying algebraic analysis techniques to non-algebraically oriented stream ciphers. As such, we do not provide a measure of the absolute or relative effectiveness of an algebraic attack against the RC4 stream cipher and its variants. Therefore, no comparisons are drawn against existing attacks, and we do not claim any advantages or disadvantages of this method over any existing ones. Sound complexity analyses on algebraic attacks are often difficult to reach due to their reliance on algorithms for solving large sparse multivariate systems of equations of varying forms, which in turn belongs to an area whose theory is yet to be fully developed and documented. Nevertheless, recent progress suggests that by implementing specialised routines to target equations generated from particular ciphers, one can improve the efficiency of equation solution greatly. These include the use of Gröbner basis (Courtois & Patarin 2003) and the Boolean Satisfiability (SAT) (Courtois & Bard 2007) algorithms. As the research on algebraic attacks progresses it is quite reasonable to believe that equation solution techniques will continue to improve in the foreseeable future. Therefore, it is important to discuss methods of generating systems of equations to describe ciphers, so that the feasibility of solution to these system and in turn the security of these respective ciphers can be constantly monitored into the future.

6 Experiments

Actual equation generation and solution attempts were made to verify the validity of the analysis and the feasibility of a successful attack on RC4. Table 2 shows the number of equations and variables that would be generated for the cipher with different word sizes. Our experiments were carried out using Magma 2.14 (Bosma et al. 1997) running on one 64-bit 1.6GHz Itanium 2 processor core on an SGI Altix 4700 supercomputer with 198 GB of shared memory.

n	Number of Variables	Number of Equations	Maximum Degree
2	64	74	3
3	288	315	4
4	1280	1348	5
5	5760	5925	6
6	26112	26502	7
7	118272	119175	8
8	532480	534536	9

Table 2: Summary of Equations Generated for RC4

The full sets of equations for $2 \leq n \leq 3$ have been successfully generated without low degree multiples, and their structures are in agreement with the analysis presented in Section 4. Using its Gröbner bases package in \mathbb{F}_2 , equations for $n = 2$ could be efficiently solved using 4 bits of keystream generated from a randomly chosen initial state. After more than 200 solution trials with keystream generated from random initial states, all of them returned a unique solution between 5.0 and 5.3 seconds. However, a few solution trials have been run with equations generated for $n = 3$, and were not successful after 48 hours of computation time each. Further investigation would be required to determine if it is infeasible to compute a solution using Gröbner basis techniques, or that the long computation time is caused by software restrictions. Nevertheless, based on the results of the experiment, we are quite confident in concluding that the full version of RC4 with $n = 8$ is most likely immune from algebraic attacks. This is based on the fact that methods for solving polynomial equations, such as Gröbner bases techniques, have time complexity exponential in the maximum degree of the equations (Becker & Weispfenning 1993). From our analysis in Section 4 and Table 2, it can be observed that the maximum degree of equations of RC4 rises linearly with the word size, so the time complexity would become infeasibly large very quickly.

7 Conclusion

This paper presents the first algebraic analysis of a non-LFSR based stream cipher, the RC4 family of stream ciphers. A method was shown for obtaining relationships between the internal states and the keystream of the word-based stream cipher. The state extraction, word addition, and state permutation operations were represented in terms of algebraic relations. These were used to form systems of equations describing the full keystream generation stage of the cipher. From these equations, we observed that having state extractions yields a system of high degree, having word addition makes the equation system inseparable, and state permutation is the main source of equations. Together, these operations constitute a strong systems of equations, in the sense that it would be infeasible to solve using currently known techniques. However, if any of these components are compromised, the strength of the system and in turn the security of RC4 against algebraic attacks would likely be reduced. It is interesting to arrive at this observation, given that the design of RC4 predates the introduction of algebraic attacks. Finally, our experimental results with reduced versions of RC4 suggest that the full RC4 is most likely immune from algebraic attacks at present. Further investigation into the cryptographic properties of RC4 is warranted for potential improvements on this first attempt at an algebraic analysis of the cipher. The findings of the algebraic properties of word-based operations in this paper could also be used as a reference for the design

of future ciphers that make use of similar components.

8 Acknowledgements

The first author wishes to thank Sultan Al-Hinai and Lynn Batten for the initial discussions on this work, and the High Performance Computing and Research Support at the Queensland University of Technology, Brisbane, Australia, for the assistance with the hardware and software used in our experiments.

References

- Al-Hinai, S., Batten, L., Colbert, B. & Wong, K. K. (2006), Algebraic attacks on clock-controlled stream ciphers, in L. M. Batten & R. Safavi-Naini, eds, '11th Australasian Conference on Information Security and Privacy — ACISP 2006', Vol. 4058 of *Lecture Notes in Computer Science*, Springer, pp. 1–16.
- Armknecht, F. & Krause, M. (2003), Algebraic attacks on combiners with memory, in D. Boneh, ed., 'Advances in Cryptology - Crypto 2003', Vol. 2729 of *Lecture Notes in Computer Science*, Springer, pp. 162–175.
- Basu, R., Maitra, S., Paul, G. & Talukdar, T. (2009), On some sequences of the secret pseudo-random index j in RC4 key scheduling, in 'Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, 18th International Symposium (AAECC-18)', Vol. 5527 of *Lecture Notes in Computer Science*, Springer, pp. 137–148.
- Becker, T. & Weispfenning, V. (1993), *Gröbner bases: A Computational Approach to Commutative Algebra*, Springer, New York, USA.
- Biham, E. & Carmeli, Y. (2008), Efficient reconstruction of RC4 keys from internal states, in 'Fast Software Encryption', Vol. 5086 of *Lecture Notes in Computer Science*, Springer, pp. 270–288.
- Bosma, W., Cannon, J. & Playoust, C. (1997), 'The MAGMA algebra system. I. The user language.', *Journal of Symbolic Computation* **24**(3-4), 235–265.
- Cho, J. Y. & Pieprzyk, J. (2004), Algebraic attacks on SOBER-t32 and SOBER-t16 without stuttering, in B. Roy & W. Meier, eds, 'Fast Software Encryption', Vol. 3017 of *Lecture Notes in Computer Science*, Springer, pp. 49–64.
- Courtois, N. (2001), The security of hidden field equations (HFE), in D. Naccache, ed., 'Topics in Cryptology - CT-RSA 2001', Vol. 2020 of *Lecture Notes in Computer Science*, Springer, pp. 266–281.
- Courtois, N. (2003), Fast algebraic attacks on stream ciphers with linear feedback, in D. Boneh, ed., 'Advances in Cryptology - Crypto 2003', Vol. 2729 of *Lecture Notes in Computer Science*, Springer, pp. 176–194.
- Courtois, N. (2004), Algebraic attacks on combiners with memory and several outputs, in C. Park & S. Chee, eds, 'Information Security and Cryptology - ICISC 2004', Vol. 3506 of *Lecture Notes in Computer Science*, Springer.
- Courtois, N. & Bard, G. V. (2007), Algebraic cryptanalysis of the Data Encryption Standard, in 'Cryptography and Coding, 11th IMA International Conference', Vol. 4887 of *Lecture Notes in Computer Science*, pp. 152–169.

- Courtois, N. & Meier, W. (2003), Algebraic attacks on stream cipher with linear feedback, in E. Biham, ed., 'Advances in Cryptology - Eurocrypt 2003', Vol. 2656 of *Lecture Notes in Computer Science*, Springer.
- Courtois, N. & Patarin, J. (2003), About the XL algorithm over $GF(2)$, in 'Topics in Cryptology - CT-RSA 2003', Vol. 2612 of *Lecture Notes in Computer Science*, Springer, pp. 141–157.
- Courtois, N. & Pieprzyk, J. (2002), Cryptanalysis of block ciphers with overdefined systems of equations, in Y. Zheng, ed., 'Advances in Cryptology - Asiacrypt 2001', Vol. 2501 of *Lecture Notes in Computer Science*, Springer, pp. 267–287.
- Fluhrer, S. R. & McGrew, D. A. (2000), Statistical analysis of the alleged RC4 keystream generator, in B. Schneier, ed., 'Fast Software Encryption', Vol. 1978 of *Lecture Notes in Computer Science*, Springer, New York, USA, pp. 19–30.
- Golić, J. D. (1997), Linear statistical weakness of alleged RC4 keystream generator, in W. Fumy, ed., 'Advances in Cryptology - Eurocrypt '97', Vol. 1233 of *Lecture Notes in Computer Science*, Springer, pp. 226–238.
- Gong, G., Gupta, K. C., Hell, M. & Nawaz, Y. (2005), Towards a general RC4-like keystream generator, in D. Feng, D. Lin & M. Yung, eds, 'Information Security and Cryptology - CISC 2005', Vol. 3822 of *Lecture Notes in Computer Science*, Springer, pp. 162–174.
- Knudsen, L., Meier, W., Preneel, B., Rijmen, V. & Verdoolaege, S. (1998), Analysis methods for (alleged) RC4, in Ohta, ed., 'Advances in Cryptology - Asiacrypt '98', Vol. 1514, Springer, pp. 327–341.
- Mantin, I. & Shamir, A. (2001), A practical attack on broadcast RC4, in M. Matsui, ed., 'Fast Software Encryption', Vol. 2355 of *Lecture Notes in Computer Science*, Springer, pp. 152–164.
- Maximov, A. (2005), Two linear distinguishing attacks on VMPC and RC4A and weakness of RC4 family of stream ciphers, in H. Gilbert & H. Handschuh, eds, 'Fast Software Encryption', Vol. 3557 of *Lecture Notes in Computer Science*, Springer, Paris, France, pp. 342–359.
- Maximov, A. & Khovratovich, D. (2008), New state recovery attack on RC4, in 'Advances in Cryptology - Crypto 2008', Vol. 5157 of *Lecture Notes in Computer Science*, Springer, pp. 297–316.
- Paul, S. & Preneel, B. (2003), Analysis of non-fortuitous predicative states of the RC4 keystream generator, in T. Johansson & S. Maitra, eds, 'Progress in Cryptology - Indocrypt 2003', Vol. 2904 of *Lecture Notes in Computer Science*, Springer, pp. 52–67.
- Paul, S. & Preneel, B. (2004), A new weakness in the RC4 keystream generator and an approach to improve the security of the cipher, in B. Roy & W. Meier, eds, 'Fast Software Encryption', Vol. 3017 of *Lecture Notes in Computer Science*, Springer, pp. 245–259.
- Schneier, B. (1996), *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd edn, John Wiley and Sons, New York, USA.
- Tsunoo, Y., Saito, T., Kubo, H., Shigeri, M., Suzaki, T. & Kawabata, T. (2005), The most efficient distinguishing attack on VMPC and RC4A, in 'Symmetric Key Encryption Workshop - SKEW 2005'.
- Wong, K. K., Colbert, B., Batten, L. & Al-Hinai, S. (2006), Algebraic attacks on clock-controlled cascade ciphers, in R. Barua & T. Lange, eds, 'Progress in Cryptology - Indocrypt 2006', Vol. 4329 of *Lecture Notes in Computer Science*, Springer, pp. 32–47.
- Zoltak, B. (2004), VMPC one-way function and stream cipher, in B. Roy & W. Meier, eds, 'Fast Software Encryption', Vol. 3017 of *Lecture Notes in Computer Science*, Springer, pp. 210–225.

Certificateless Key Agreement in the Standard Model *

Georg Lippold

Juan González Nieto

Information Security Institute, Queensland University of Technology,
 PO Box 2434, Brisbane, Queensland 4001, Australia
 Email: {g.lippold,j.gonzaleznieto}@qut.edu.au

Abstract

We show how to construct a certificateless key agreement protocol from the certificateless key encapsulation mechanism introduced by Lippold et al. (2009a) in ICISC 2009 using the Boyd et al. (2008) protocol from ACISP 2008. We introduce the Canetti-Krawczyk (CK) model for certificateless cryptography, give security notions for Type I and Type II adversaries in the CK model, and highlight the differences to the existing e^2 CK model discussed by Lippold et al. (2009b). The resulting CK model is more relaxed thus giving more power to the adversary than the original CK model.

Keywords: key agreement, key exchange, key encapsulation mechanism, certificateless, standard model, Diffie-Hellman

1 Introduction

CERTIFICATELESS ENCRYPTION introduced by Al-Riyami & Paterson (2003) is a variant of identity based encryption that limits the key escrow capabilities of the key generation centre (KGC), which are inherent in identity based encryption Boneh & Franklin (2003). Dent (2008) published a survey of more than twenty certificateless encryption schemes that focuses on the different security models and the efficiency of the respective schemes. In certificateless cryptography schemes, there are three secrets per party:

1. The key issued by the key generation centre (Dent (2008) calls it “partial private key”). We assume in the following that this key is ID-based, although it does not necessarily have to be ID-based.
2. The user generated private key x_{ID} (Dent calls it “secret value”).
3. The ephemeral value chosen randomly for each protocol run.

KEY AGREEMENT SCHEMES provide an efficient means for two parties to communicate over an adversarial controlled channel. An overview of almost twenty identity based key agreement protocols has been compiled by Chen et al. (2007); they also provide security proofs for two of the surveyed protocols. Many ID-based schemes guarantee full privacy

for both parties as long as the key generation centre (KGC) does not learn any of the ephemeral secrets used in computing the session key. But as Krawczyk (2005) points out, the leakage of ephemeral keys should not be neglected as they are usually pre-computed and not stored in secure memory. In the context of identity based key agreement protocols, this means that as soon as the ephemeral key of either party leaks, a malicious KGC is able to compute the session key.

AN OVERVIEW OF CURRENT CERTIFICATELESS KEY AGREEMENT SCHEMES has been compiled by Swanson (2008). Certificateless key agreement schemes attempt to provide full privacy even if the ephemeral secrets of the parties leak to the key generation centre or if the key generation centre actively interferes with the messages that are exchanged (e.g. does a man-in-the-middle attack). The first certificateless key agreement scheme with a proof of security was recently published by Lippold et al. (2009b) in the random oracle model (ROM). Lippold et al. (2009b) describe why it is hard to construct and prove certificateless key agreement schemes. The scheme they propose is computationally very expensive and in the random oracle model. They leave the construction of an efficient protocol and the construction of a standard secure protocol as open questions. In this paper, we would like to answer these two open problems. In ACISP 2008, Boyd et al. (2008) showed how to construct identity based (ID) and public key based (PK) authenticated key agreement (AKE) from key encapsulation mechanisms (KEM) secure in the respective model. In this paper, we extend the model to certificateless key encapsulation mechanisms (CL-KEM) and show how to construct an efficient CL-AKE secure in the standard model from the CL-KEM scheme in the standard model recently published by Lippold et al. (2009a).

THE SECURITY MODEL in this paper is a new development of the e^2 CK security model defined by Lippold et al. (2009b) and the Canetti & Krawczyk (2001) security model. We extend the Canetti and Krawczyk (CK) model in Section 3 on page 3 to allow partial corruption of the long term secrets. We provide a meaningful merge between the e^2 CK and the CK models for certificateless encryption that is strong enough to realistically model an adversary against the protocol and is at the same time flexible enough to allow a simple construction of a certificateless key agreement protocol in the standard model. The model is more relaxed due to the fact that we also allow partial corruption of parties. A certificateless key agreement protocol in this model is secure even if one party is fully corrupted and one party is partially corrupted. Additionally, we define the notion of weak Type I and weak Type II CK adversaries for certificateless key agreement. Type I adversaries model an outsider adversary that does not know the

*Research funded by the Australian Research Council through Discovery Project DP0666065
 Copyright ©2010, Australian Computer Society, Inc. This paper appeared at the Australasian Information Security Conference (AISC2010), Brisbane, Australia. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 105, Colin Boyd and Willy Susilo, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

master secret key of the KGC; Type II adversaries model insider adversaries that possess the master secret key of the KGC. These definitions correspond to the weak Type I and weak Type II adversary definitions for certificateless encryption given by Dent (2008).

THE MAIN CONTRIBUTIONS of this paper are:

- First efficient provably secure protocol for certificateless key exchange in the standard model.
- New enhanced security model for certificateless key exchange based on the well-known Canetti-Krawczyk model, giving more power to the adversary.
- New definition of weak Type I and weak Type II certificateless adversaries in the Canetti-Krawczyk model.

2 Definitions

Def. 1 Min-entropy Gennaro et al. (2004)

Let χ be a probability distribution over A . The min-entropy of χ is the value

$$\text{min-ent}(\chi) = \min_{x \in A: \Pr_{\chi}[x] \neq 0} (-\log_2(\Pr_{\chi}[x])) \quad (1)$$

If χ has min-entropy t , then for all $x \in A : \Pr_{\chi}[x] \leq 2^{-t}$.

Def. 2 Strong randomness extractor Nisan & Zuckerman (1996)

A family of efficiently computable hash functions $\mathcal{H} = \{h_{\kappa} : \{0, 1\}^n \rightarrow \{0, 1\}^k | \kappa \in \{0, 1\}^d\}$ is called a strong (m, ϵ) -randomness extractor, if for any random variable X over $\{0, 1\}^n$ that has min-entropy at least m , if κ is chosen uniformly at random from $\{0, 1\}^d$, and R is chosen uniformly at random from $\{0, 1\}^k$, the two distributions $\langle \kappa, h_{\kappa}(X) \rangle$ and $\langle \kappa, R \rangle$ have statistical distance ϵ , that is

$$\frac{1}{2} \sum_{x \in \{0, 1\}^k} |\Pr[h_{\kappa}(X) = x] - \Pr[R = x]| = \epsilon \quad (2)$$

Def. 3 Pseudorandom Function Family (PRF)

Let $\mathcal{F} = \{f_s\}_{s \in S}$ be a family of functions for security parameter $k \in \mathbb{N}$ and with seed $s \in S = S(k)$. Let C be an adversary that is given oracle access to either f_s for $s \xleftarrow{\$} K$ or a truly random function with the same domain and range as the functions in \mathcal{F} . \mathcal{F} is said to be pseudorandom if C 's advantage in distinguishing whether it has access to a random member of \mathcal{F} or a truly random function is negligible in k , for all polynomial-time adversaries C . That is,

$$\text{Adv}_{\mathcal{F}, C}^{p\text{-rand}}(k) = \left| \Pr[C^{F_s(\cdot)}(1^k) = 1] - \Pr[C^{Rand(\cdot)}(k) = 1] \right| \quad (3)$$

is negligible in k .

Def. 4 Decisional Diffie-Hellman (DDH)

Let F be a cyclic group of order p' generated by an element f . Consider the set $F^3 = F \times F \times F$ and the following two probability distributions over it:

$$\mathcal{R}_F = \{(f^a, f^b, f^c) : (a, b, c) \xleftarrow{\$} \mathbb{Z}_{p'}\} \quad (4)$$

and

$$\mathcal{DH}_F = \{(f^a, f^b, f^{ab}) : (a, b) \xleftarrow{\$} \mathbb{Z}_{p'}\} \quad (5)$$

We say the Decisional Diffie-Hellman Assumption holds over $F = \langle f \rangle$ if the two distributions \mathcal{R}_F and \mathcal{DH}_F are indistinguishable by all polynomial-time adversaries \mathcal{D} . More precisely, for $k = |p'|$

$$\text{Adv}_{F, \mathcal{D}}^{ddh}(k) = \left| \Pr[\mathcal{D}(1^k, \rho) = 1 | \rho \xleftarrow{\$} \mathcal{DH}_F] - \Pr[\mathcal{D}(1^k, \rho) = 1 | \rho \xleftarrow{\$} \mathcal{R}_F] \right| \quad (6)$$

Def. 5 CL-KEM

A certificateless key encapsulation mechanism (CL-KEM) $\mathcal{E} = (\text{Setup}, \text{KeyDer}, \text{UserKeyGen}, \text{Enc}, \text{Dec})$ consists of five polynomial-time algorithms:

Setup $(mpk, msk) \xleftarrow{\$} \text{Setup}(1^k)$ Given the security parameter $k \in \mathbb{N}$, the Setup algorithm returns a master public key mpk , and a master secret key msk .

KeyDer $d_{ID} \xleftarrow{\$} \text{KeyDer}(mpk, msk, ID)$ generates an ID-based private key d_{ID} from the master secret key msk for identity ID .

UserKeyGen $(x_{ID}, \beta_{ID}) \xleftarrow{\$} \text{UserKeyGen}(mpk, ID)$ generates a user secret key x_{ID} and a user certificateless public key β_{ID} from the master public key and the identity.

Enc $(C, K) \xleftarrow{\$} \text{Enc}(mpk, ID, \beta_{ID})$ generates the key K , and the certificateless key encapsulation C of K .

Dec $K \xleftarrow{\$} \text{Dec}(d_{ID}, x_{ID}, C)$ extracts the key K from the certificateless encapsulation C using the ID-based private key d_{ID} and the user secret key x_{ID} .

This definition of a CL-KEM is based on the definition given in Lippold et al. (2009a).

Def. 6 CL-KEM Security

In a CL-KEM environment, the adversary has access to the following oracles:

Reveal master key: The adversary is given access to the master secret key.

Reveal ID-based key(ID): The adversary extracts the ID-based private key of party ID .

Get user public key(ID): The adversary obtains the certificateless public key for ID . If the certificateless key for the identity has not yet been generated, it is generated with the user key gen algorithm.

Replace public key(ID, pk): Party ID 's certificateless public key is replaced with pk chosen by the adversary. All communication (encryption, encapsulation) for Party ID will use the new public key.

Reveal secret value(ID): The adversary extracts the secret value x_{ID} that corresponds to the certificateless public key for party ID . If the adversary issued a replace public key query for ID before, \perp is returned.

Decapsulate(ID, C): The adversary learns the decapsulation of C under ID or \perp if C is invalid or if the adversary replaced the public key of ID .

Decapsulate(ID, C, x): The adversary learns the decapsulation of C under ID using the secret value x . The special symbol \perp will be returned if C is invalid.

Get challenge key encapsulation(ID^*): *The adversary requests a challenge key encapsulation and thus marks the transition from $\mathbf{Oracles}_1$ to $\mathbf{Oracles}_2$ in Experiment 7. The simulator returns a challenge key encapsulation as described in Experiment 7.*

The security of a CL-KEM scheme $\mathcal{E} = (\text{Setup}, \text{KeyDer}, \text{UserKeyGen}, \text{Enc}, \text{Dec})$ is defined by the following experiment:

Exp. Chall. $\xleftarrow{\$} \text{CL-KEM}_{MM}^{cl-kem-cca}(k)$:

$$\begin{aligned} (mpk, msk) &\xleftarrow{\$} \text{CL-KEM Setup}(k) \\ (ID^*, state) &\xleftarrow{\$} \mathcal{M}^{\mathbf{Oracles}_1}(find, mpk) \\ K_0^* &\xleftarrow{\$} \mathcal{K}; (C^*, K_1^*) \xleftarrow{\$} \text{CL-KEM Enc}(pk, ID^*) \\ \gamma &\xleftarrow{\$} \{0, 1\}; K^* = K_\gamma^* \\ \gamma' &\xleftarrow{\$} \mathcal{M}^{\mathbf{Oracles}_2}(guess, K^*, C^*, state) \\ \text{Return } \gamma &== \gamma' \end{aligned} \quad (7)$$

The advantage an adversary \mathcal{M} has against a CL-KEM scheme is therefore expressed by

$$\text{Adv}_{\mathcal{M}}^{\text{CL-KEM}}(k) = \left| \Pr \left[\text{Exp. Chall.}_{\text{CL-KEM}_{MM}}^{cl-kem-cca}(k) \right] - 1/2 \right|$$

For a Type I adversary \mathcal{M} , $\mathbf{Oracles}_1$ and $\mathbf{Oracles}_2$ mean access to all oracles listed above with the following limitations:

1. No reveal master key queries.
2. C^* must not be submitted to a decapsulate oracle under ID^* .
3. Not both (reveal secret value OR replace public key) AND reveal ID-based key oracles may be asked for ID^* .

For a Type 2 adversary \mathcal{M} , $\mathbf{Oracles}_1$ and $\mathbf{Oracles}_2$ are subject to the following limitations:

1. $\mathbf{Oracles}_1$ and $\mathbf{Oracles}_2$ now includes reveal master key as allowed query,
2. C^* must not be submitted to a decapsulate oracle under ID^* .
3. reveal secret value must never be asked for ID^* ,
4. $\mathbf{Oracles}_1$ must not include replace public key for ID^* .

This definition stems from (Lippold et al. 2009a, Section 3.3)

Def. 7 Weak Perfect Forward Secrecy (wPFS)
A key-exchange protocol provides weak PFS (wPFS) if an attacker \mathcal{M} cannot distinguish from random a key of any session for which the session and its matching session are clean¹ even if \mathcal{M} has learned the private keys of both peers to the session (Krawczyk 2005, Definition 22)

Def. 8 Key Compromise Impersonation (KCI)
We say that a KE-attacker \mathcal{M} that has learned the private key of party \hat{A} succeeds in a Key-compromise impersonation (KCI) attack against \hat{A} if \mathcal{M} is able to distinguish from random the session key of a complete session at \hat{A} for which the session peer is uncorrupted and the session and its matching session (if it exists) are clean (Krawczyk 2005, Definition 20).

¹Roughly speaking clean is the same as fresh in Definition 9.

3 Formal definition of the security model

We want to use the Protocol by Boyd et al. (2008) to construct a certificateless key agreement in the standard model. Current existing security definitions for key agreement protocols were defined by Swanson (2008) in the e²CK model and later improved by Lippold et al. (2009b). However, the protocol by Boyd et al. (2008) is proven in the Canetti-Krawczyk model. We continue to list the two security models and then discuss the major differences.

3.1 The e²CK model

Lippold et al. (2009b) strengthened the e²CK security model for certificateless key exchange that was introduced by Swanson (2008). We recapitulate the model here briefly.

Let $\mathcal{U} = \{U_1, \dots, U_n\}$ be a set of parties. The protocol may be run between any two of these parties. For each party there exists an identity based public key that can be derived from its identifier. There is a key generation centre that issues identity based private keys to the parties through a secure channel. Additionally, the parties generate their own secret values and corresponding certificateless public keys.

The adversary is in control of the network over which protocol messages are exchanged. $\Pi_{i,j}^t$ represents the t^{th} protocol session which runs at party i with intended partner party j . Additionally, the adversary is allowed to replace certificateless public keys that are used to compute the session key. The adversary does not have to disclose the private key matching the replaced certificateless public key to the respective party.

A session $\Pi_{i,j}^t$ enters an *accepted state* when it computes a session key $SK_{i,j}^t$. Note that a session may terminate without ever entering into an accepted state. The information of whether a session has terminated with acceptance or without acceptance is assumed to be public. The session $\Pi_{i,j}^t$ is assigned a partner ID $pid = (ID_i, ID_j)$. The session ID sid of $\Pi_{i,j}^t$ at party i is the transcript of the messages exchanged with party j during the session. Two sessions $\Pi_{i,j}^t$ and $\Pi_{j,i}^u$ are considered matching if they have the same pid (and sid).

The game runs in two phases. During the first phase of the game, the adversary \mathcal{M} is allowed to issue the following queries in any order:

Send($\Pi_{i,j}^t, x$): If the session $\Pi_{i,j}^t$ does not exist, it will be created as initiator at party i if $x = \lambda$, or as a responder at party j otherwise. If the participating parties have not been initiated before, the respective private and public keys are created. Upon receiving the message x , the protocol is executed. After party i has sent and received the last set of messages specified by the protocol, it outputs a decision indicating accepting or rejecting the session. In the case of one-round protocols, party i behaves as follows:

- $x = \lambda$: Party i generates an ephemeral value and responds with an outgoing message only.
- $x \neq \lambda$: If party i is a responder, it generates an ephemeral value for the session and responds with an outgoing message m and a decision indicating acceptance or rejection of the session. If party i as an initiator, it responds with a decision indicating accepting or rejecting the session.

In this work, we require $i \neq j$, i.e. a party will not run a session with itself.

Reveal master key The adversary is given access to the master secret key.

Session key reveal($\Pi_{i,j}^t$): If the session has not accepted, it returns \perp , otherwise it reveals the accepted session key.

Reveal ID-based secret(i): Party i responds with its ID-based private key, e.g. $sH_1(ID_i)$.

Reveal secret value(i): Party i responds with its secret value x_i that corresponds to its certificateless public key. If i has been asked the *replace public key* query before, it responds with \perp .

Replace public key(i, pk): Party i 's certificateless public key is replaced with pk chosen by the adversary. Party i will use the new public key for all communication and computation.

Reveal ephemeral key($\Pi_{i,j}^t$): Party i responds with the ephemeral secret used in session $\Pi_{i,j}^t$.

We can group the key reveal queries into three types: the *reveal master key* and *reveal ID-based secret* queries try to undermine the security of the ID-based part of the scheme, the *reveal secret value* and *replace public key* queries try to undermine the security of the public key based part of the scheme, and the *reveal ephemeral key* query tries to undermine the security of one particular session.

We define the state *fully corrupt* as a session that was asked all three types of reveal queries: the *reveal master key* or *reveal ID-based secret*, the *reveal secret value* or the *replace public key*, and the *reveal ephemeral key* query.

Once the adversary \mathcal{M} decides that the first phase is over, it starts the second phase by choosing a *fresh session* $\Pi_{i,j}^t$ and issuing a *Test*($\Pi_{i,j}^t$) query, where the *fresh session* and *test query* are defined as follows:

Def. 9 Fresh session

A session $\Pi_{i,j}^t$ is *fresh* if (1) $\Pi_{i,j}^t$ has accepted; (2) $\Pi_{i,j}^t$ is unopened (not being issued the session key reveal query); (3) the session state at neither party participating in this session is fully corrupted; (4) there is no opened session $\Pi_{j,i}^u$ which has a matching conversation to $\Pi_{i,j}^t$.

Test($\Pi_{i,j}^t$) The input session $\Pi_{i,j}^t$ must be fresh. A bit $b \in \{0, 1\}$ is randomly chosen. If $b = 0$, the adversary is given the session key, otherwise it randomly samples a session key from the distribution of valid session keys and returns it to the adversary.

After the *test*($\Pi_{i,j}^t$) query has been issued, the adversary can continue querying except that the test session $\Pi_{i,j}^t$ should remain *fresh*. We emphasize here that partial corruption is allowed as this is a benefit of our security model. Additionally, *replace public key* queries may be issued to *any* party after the test session has been completed.

At the end of the game, the adversary outputs a guess \hat{b} for b . If $\hat{b} = b$, we say that the adversary wins. The adversary's advantage in winning the game is defined as

$$Adv^{\mathcal{M}}(k) = \left| \Pr[\mathcal{M} \text{ wins}] - \frac{1}{2} \right|$$

3.2 The Canetti-Krawczyk model

We give a slightly shortened version of the CK model used by Boyd et al. (2008) and refer the reader to the paper for details. In the Canetti-Krawczyk (CK) model, a protocol π is modeled as a collection of n programs running at different parties P_1, \dots, P_n . Each program is an interactive probabilistic polynomial-time (PPT) machine. A *session* is defined as an invocation of π at party P_i , and every party may have multiple sessions running concurrently. The communication network is controlled by the adversary \mathcal{M} , who is also a PPT machine. The adversary controls the message flow between the parties by activating a party P_i , which may be done in two ways:

1. An *establish session* (P_i, P_j, s) request where P_j is another party with whom the session is to be established, and s is the session ID string which uniquely identifies a session between the participants.
2. By means of an *incoming message* m with a specified sender P_j .

A *matching session* is defined by having two session (P_i, P_j, s) and (P'_i, P'_j, s') for that $P_i = P'_i$, $P_j = P'_j$ and $s = s'$. s is defined by the concatenation of the messages exchanged by the respective parties. \mathcal{M} can ask any of the following queries:

corrupt(P_i) \mathcal{M} learns the long term key of P_i .

session-key(P_i, P_j, s) \mathcal{M} learns the session key of an accepted session for (P_i, P_j, s) at party P_i .

session-state(P_i, P_j, s) Returns the internal state information of party P_i with respect to session s with party P_j but does not include the long term key of P_j .

session-expiration(P_i, P_j, s) Erases the session key form for session s with P_j from the internal memory of P_i .

test-session(P_i, P_j, s) The challenger \mathcal{B} selects a random bit b . If $b = 1$, then the correct session key is returned. Otherwise, a randomly chosen key from the probability distribution of the key space of the protocol is returned. This query may only be asked to a session that is not *exposed*, where an *exposed* session is defined as a session that has been asked either

- a *session-state* or *session-key* reveal query to this session or the matching session, or
- a *corrupt* query to either partner before the session expires at that partner.

3.3 Comparison of the two models

Although the *send* query in the e²CK model roughly corresponds to the *activation* in the CK model, there are some subtle differences in the security models that we try to sort out in the following.

The CK model used by Boyd et al. (2008) does not consider partial corruption of the long term secret, as Boyd et al. only consider cases where a party has only one long term secret. On the contrary, in the e²CK model used by Lippold et al. (2009b), each party has two long-term secrets which may be corrupted "individually".

Furthermore, Boyd et al. (2008) do not allow the adversary only to extract the randomness used in a specific run of the protocol, i.e. they do not allow *ephemeral key reveal* queries.

On the other hand, Boyd et al. allow *state reveal* queries in addition to the *session-key reveal* query that both models allow, but both queries are not allowed for the *test session*.

For the following, we propose the following meaningful mapping from one model to the other:

- Instead of using *ephemeral key reveal*, we allow *session state reveal* queries. We note that *session state reveal* queries must not target the test session, so they are of limited use to the adversary. Our goal in this paper is to extend the KEM-KEM-AKE construction by Boyd et al. (2008) to the certificateless case. As the proofs for KEM schemes do not allow the adversary to extract the randomness used during the key encapsulation (which would be the equivalent to an ephemeral key), ephemeral key reveal queries cannot be allowed in a protocol that uses a KEM as a building block for any other scheme. KEM schemes would be trivial to break if any adversary was allowed to recover the randomness used in the challenge query.
- However, we allow the adversary to corrupt the long term secrets of a party “individually”. On the one hand this allows for Type II certificateless adversaries that correspond to a key generation centre for the ID-based scheme as described in Section 3.4, on the other hand we give more power to the adversary as the protocol must still be secure even if one party is fully corrupted and the other party is partially corrupted.

3.4 Adversaries against weakly CK-secure certificateless key agreement schemes

Both Swanson (2008) and Lippold et al. (2009b) give only definitions for adversaries in the e²CK model. We give the first security definitions for an adversary in the Canetti-Krawczyk (CK) model against a weakly secure certificateless key agreement protocol. The Type I adversary models an outsider adversary that may corrupt parties but may not learn the master secret key of the key generation centre (KGC). The Type II adversary reflects a malicious but honest KGC.

Def. 10 Weak Type I CK-secure key agreement scheme

A certificateless key agreement scheme is Weak Type I CK-secure if every probabilistic, polynomial-time adversary \mathcal{M} has negligible advantage in winning the game described in Section 3 on page 3 subject to the following constraints.

- \mathcal{M} may corrupt at most two long-term secrets of one party involved in the test session, and one long-term secret of the other party involved in the test session.
- \mathcal{M} is allowed to replace public keys of any party; however, this counts as the corruption of one secret.
- \mathcal{M} may not reveal the secret value of any identity for which it has replaced the certificateless public key.
- \mathcal{M} is not allowed to ask session key reveal queries for session keys computed by identities where \mathcal{M} replaced the identity’s public key.
- \mathcal{M} is allowed to replace public keys of any party after the test query has been issued.

- \mathcal{M} is not allowed to ask session state reveal queries for sessions at identities where \mathcal{M} replaced the identity’s public key.
- \mathcal{M} is not allowed to ask session state reveal queries for the test session or the matching session to the test session.
- \mathcal{M} is not allowed to ask ephemeral key reveal queries.

Def. 11 Weak Type II CK-secure key agreement scheme

A certificateless key agreement scheme is Weak Type II CK-secure if every probabilistic, polynomial-time adversary \mathcal{M} has negligible advantage in winning the game described in Section 3 on page 3 subject to the following constraints.

- \mathcal{M} is given the master secret key s at the start of the game.
- \mathcal{M} may corrupt at most one user secret key of the parties participating in the test session.
- \mathcal{M} is allowed to replace the certificateless public key of any party; however, this counts as the corruption of a user secret key.
- \mathcal{M} may not reveal the secret value of any identity for which it has replaced the certificateless public key.
- \mathcal{M} is not allowed to ask session key reveal queries for session keys computed by identities where the identity’s public key was replaced.
- \mathcal{M} is allowed to replace public keys of any party after the test query has been issued.
- \mathcal{M} is not allowed to ask session state reveal queries for sessions at identities where \mathcal{M} replaced the identity’s public key.
- \mathcal{M} is not allowed to ask session state reveal queries for the test session or the matching session to the test session.
- \mathcal{M} is not allowed to ask ephemeral key reveal queries.

4 The Boyd et al. Protocol with a CL-KEM

We recall the generic AKE protocol constructions from KEM schemes by Boyd et al. (2008) from ACISP’08. The first scheme does not offer weak perfect forward secrecy (wPFS) as described in Definition 7 on page 3 but offers key compromise impersonation (KCI) resistance as described in Definition 8 on page 3. It is shown in Table 1 on the following page. The second scheme adds an additional Diffie-Hellman to the first protocol and then achieves both weak perfect forward secrecy and KCI resistance. The protocol is shown in Table 2 on the next page. Boyd et al. (2008) prove these protocols secure even if the long term secret of one party is known to the adversary.

We use these protocols to construct a weakly CK-secure certificateless key agreement scheme in the standard model, by replacing the KEM scheme in the Boyd et al. (2008) construction with the certificateless KEM (CL-KEM) scheme recently proposed by Lippold et al. (2009a). Constructing a certificateless key agreement scheme in the standard model is an open problem considered by Lippold et al. (2009b).

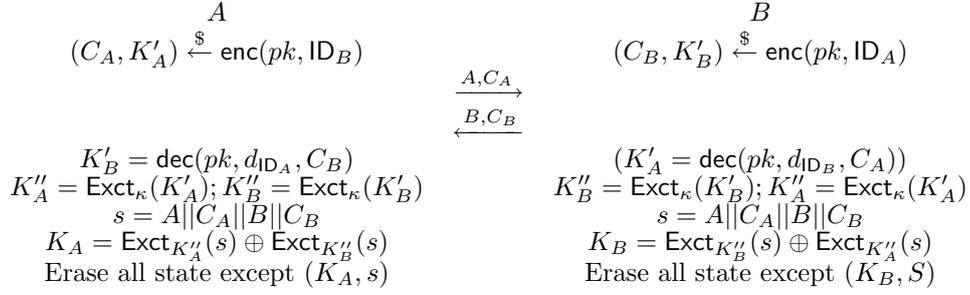


Table 1: Boyd et al. (2008) Protocol 1

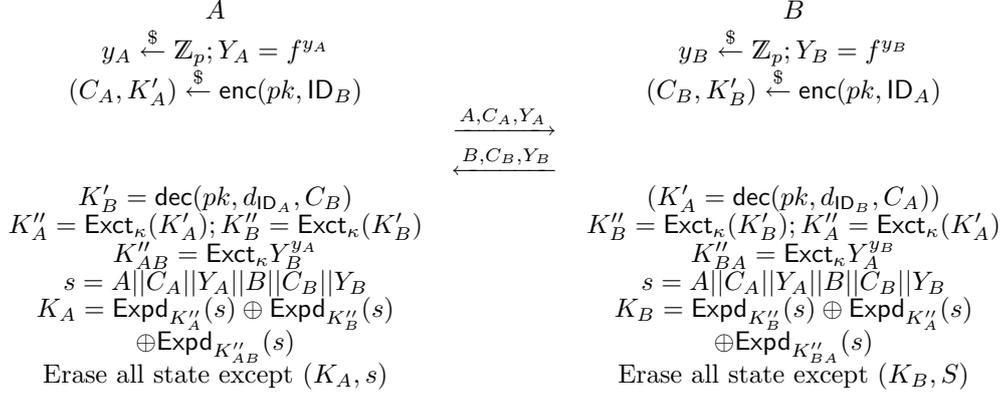


Table 2: Boyd et al. (2008) Protocol 2

In Table 1 and Table 2, the following notations are used:

- $\{\text{Expd}_K(\cdot)\}_{K \in \mathbb{U}_1} : \{0, 1\}^\sigma \rightarrow \mathbb{U}_2$ is a pseudorandom function family as in Definition 3 on page 2.
- $\text{Exct}_{\kappa}(\cdot) : \mathbb{K} \rightarrow \mathbb{U}_1$ is a randomly chosen strong (m, ϵ) -randomness extractor as in Definition 2 on page 2 for appropriate m and ϵ .
- n_{orac} is the total number of sessions / oracles created by the adversary against the protocol.
- $\frac{1}{p}$ is the maximum probability that $C_1 = C_2$ where $(C_1, K_1) \xleftarrow{\$} \text{enc}(pk, \text{ID})$ and $(C_2, K_2) \xleftarrow{\$} \text{enc}(pk, \text{ID})$ for any identity ID.

5 Proving the derived protocol secure

Lippold et al. (2009b) point out that a “natural” combination of an ID-based key agreement protocol (ID-AKE) and a public key based key agreement protocol (PK-AKE) does not give full security in the CL-AKE setting. In the e²CK model described by Lippold et al. (2009b) there are three secrets per party of which the adversary may corrupt any two to gain an advantage in breaking the CL-AKE protocol. The adversary may choose the secrets in its favour: by corrupting the ID-based secrets at party one on the PK secret at the other party, and the ephemeral secrets at both parties, both protocols are broken although each party still holds one uncompromised secret. We investigate if a CL-AKE from a CL-KEM plugged into the KEM-KEM construction by Boyd et al. (2008) provides reasonable security.

Obviously, as Protocol 1 from Boyd et al. (2008) does not achieve weak forward secrecy, the security of the protocol relies on the long-term keys of the respective parties. Thus, this type of protocol will not give resistance if all four long term secrets of both parties

are compromised. We now turn to the security proof for the protocol in the reconsidered security model explained in Section 3.3 on page 4 and Section 3.4 on the previous page.

We follow the proof given in Boyd et al. (2008) closely and will only list the changes that are needed to prove the scheme secure against a certificateless adversary. Recall that the weak certificateless adversary may not ask session key or session state reveal queries for sessions at a party where the adversary replaced that party’s certificateless public key nor may these queries be asked for the *test* session.

We give some intuition why the proof for the Boyd et al. (2008) protocol does still hold in the reconsidered security model from section 3.3 on page 4 and section 3.4 on the previous page. In a CL-KEM setting, there are only two secrets per party considered, as a KEM is a “receive only” protocol. The ephemeral secret used to construct the message is never disclosed to the adversary in a KEM protocol. The security of the KEM holds in the certificateless case if a party has at least one uncompromised secret, i.e. an uncompromised ID-based key or an uncompromised user secret key. Consider the CL-KEM-KEM-AKE setting, where the *test* session runs between party A with ID-based private key d_A and user secret key x_A and party B with ID-based private key d_B and user secret key x_B . There are now essentially three cases to distinguish:

1. A weak Type I CK-adversary \mathcal{M} that corrupts both long-term secrets d_A and x_A of party A and one long-term secret of party B. In this case the security of the KEM at party B guarantees the security of the protocol.
2. A weak Type I CK-adversary \mathcal{M} that corrupts only one long-term secret of party A but both long-term secrets d_B, x_B of party B. In this case the security of the KEM at party A guarantees the security of the protocol.

3. The case where the CK-adversary corrupts all long-term keys d_A, x_A, d_B, x_B of the respective parties. In this case Protocol 1 is broken, but Protocol 2 is still secure due to the additional Diffie-Hellman.

For Protocol 1 we propose the following theorem similar to Boyd et al. (2008):

Theorem 1 *Let \mathcal{B} be any adversary against Protocol 1. Then the advantage of \mathcal{B} against the SK-security (with partial WFS and KCI resistance) of Protocol 1 is:*

$$\text{Adv}_{\mathcal{B}}^{\text{sk}}(k) \leq \frac{n_{\text{orac}}^2}{b} + 2n_{\text{orac}} \left(\text{Adv}_{\mathcal{E},A}^{\text{CL-KEM-CCA}}(k) + \epsilon + \text{Adv}_{\mathcal{F},C}^{\text{p-rand}}(k) \right)$$

The proof in Appendix A on the following page for Protocol 1 works for the cases 1 and 2 above and is very similar to the proof in Boyd et al. (2008). All games in the proof were left as in the original paper, only Game 3 was modified. In the certificateless setting, besides fully corrupting party A in case one, the adversary is also allowed to partially corrupt party B. In case two similarly the adversary is allowed to fully corrupt party B and partially corrupt party A. However, in both cases the CL-KEM scheme for the partially corrupted party (B in case 1 and A in case 2) is still secure, as by definition CL-KEM schemes tolerate partial corruption. Boyd et al. (2008) use any successful adversary against the KEM-KEM-AKE construction as an adversary against the KEM scheme in Game 3 of their proof. This proof technique carries through to the certificateless case in our modified security model as all oracle queries that the adversary may ask can still be answered by the challenger as described in Game 3 of Boyd et al. (2008).

For Protocol 2 on the previous page we have the following theorem in analogy to Boyd et al. (2008):

Theorem 2 *Let \mathcal{B} be any adversary against Protocol 2. Then the advantage of \mathcal{B} against the SK-security (with WFS and KCI resistance) of Protocol 2 is:*

$$\begin{aligned} \text{Adv}_{\mathcal{B}}^{\text{sk}}(k) \leq & \max \left(2n_{\text{orac}}^2 \text{Adv}_{F,D}^{\text{ddh}}(k) + 2\epsilon \right. \\ & + 2\text{Adv}_{\mathcal{F},C}^{\text{p-rand}}(k), \\ & \left. \frac{n_{\text{orac}}^2}{p} + 2n_{\text{orac}} \left(\text{Adv}_{\mathcal{E},A}^{\text{CL-KEM-CCA}}(k) \right. \right. \\ & \left. \left. + \epsilon + \text{Adv}_{\mathcal{F},C}^{\text{p-rand}}(k) \right) \right). \end{aligned}$$

As Boyd et al. (2008) do not alter Game 3 in their proof of Protocol 2 with respect to the proof of Protocol 1, the proof carries through to the CL-KEM case as well. We included their proof in Appendix B.

6 Conclusion

We give the first construction of an efficient certificateless key agreement scheme proven secure in the standard model. We use the existing KEM-KEM construction by Boyd et al. (2008) from ACISP 2008 to construct our scheme. We review existing security notions for certificateless key exchange and propose the new notion of Canetti-Krawczyk security for certificateless key agreement. We show that the KEM-KEM construction by Boyd et al. (2008) holds also for certificateless KEM schemes in the new certificateless Canetti-Krawczyk security model for certificateless key agreement.

References

- Al-Riyami, S. S. & Paterson, K. G. (2003), Certificateless Public Key Cryptography, in C.-S. Lai, ed., ‘ASIACRYPT’, Vol. 2894 of *Lecture Notes in Computer Science*, Springer, pp. 452–473. Online available at <http://eprint.iacr.org/2003/126.pdf>. 1
- Boneh, D. & Franklin, M. (2003), ‘Identity based encryption from the Weil pairing’, *SIAM Journal of Computing* **32**(3), 586–615. Online available at <http://crypto.stanford.edu/~dabo/papers/bfibe.pdf>. 1
- Boyd, C., Cliff, Y., González Nieto, J. M. & Paterson, K. G. (2008), Efficient one-round key exchange in the standard model, in Y. Mu, W. Susilo & J. Seberry, eds, ‘ACISP’, Vol. 5107 of *Lecture Notes in Computer Science*, Springer, pp. 69–83. 1, 3, 4, 5, 6, 7
- Canetti, R. & Krawczyk, H. (2001), Analysis of key-exchange protocols and their use for building secure channels, in B. Pfitzmann, ed., ‘EUROCRYPT’, Vol. 2045 of *Lecture Notes in Computer Science*, Springer, pp. 453–474. 1
- Chen, L., Cheng, Z. & Smart, N. P. (2007), ‘Identity-based key agreement protocols from pairings’, *Int. J. Inf. Sec.* **6**(4), 213–241. 1
- Dent, A. W. (2006), ‘A note on game-hopping proofs’, Cryptology ePrint Archive, Report 2006/260. <http://eprint.iacr.org/2006/260>. 8, 11
- Dent, A. W. (2008), ‘A survey of certificateless encryption schemes and security models’, *International Journal of Information Security* **7**(5), 349–377. URL: <http://dx.doi.org/10.1007/s10207-008-0055-0> 1, 2
- Gennaro, R., Krawczyk, H. & Rabin, T. (2004), Secure hashed diffie-hellman over non-ddh groups, in C. Cachin & J. Camenisch, eds, ‘EUROCRYPT’, Vol. 3027 of *Lecture Notes in Computer Science*, Springer, pp. 361–381. 2
- Krawczyk, H. (2005), ‘HMQV: A High-Performance Secure Diffie-Hellman Protocol’, Cryptology ePrint Archive, Report 2005/176. <http://eprint.iacr.org/2005/176>. 1, 3
- Lippold, G., Boyd, C. & González Nieto, J. M. (2009a), ‘Efficient Certificateless KEM in the Standard Model’, Cryptology ePrint Archive, Report 2009/451. <http://eprint.iacr.org/2009/451>, ICISC 2009 accepted paper. 1, 2, 3, 5
- Lippold, G., Boyd, C. & González Nieto, J. M. (2009b), Strongly Secure Certificateless Key Agreement, in H. Shacham & B. Waters, eds, ‘Pairing’, Vol. 5671 of *Lecture Notes in Computer Science*, Springer, pp. 206–230. 1, 3, 4, 5, 6
- Nisan, N. & Zuckerman, D. (1996), ‘Randomness is linear in space’, *J. Comput. Syst. Sci.* **52**(1), 43–52. 2
- Swanson, C. M. (2008), ‘Security in Key Agreement: Two-Party Certificateless Schemes’, http://uwspace.uwaterloo.ca/bitstream/10012/4156/1/Swanson_Colleen.pdf. Master Thesis, University of Waterloo. Download 2009-01-29. 1, 3, 5

A Proof of Theorem 1

We now proceed to prove Theorem 1. The proof has two parts; the first part proves the security of Protocol 1 proves the security of Protocol 2 when the partner to the test session is not corrupted. The second part proves the security of Protocol 1 when the partner to the test session is corrupted (in this case, we require the test session to have a matching session by the time \mathcal{B} finishes). Remember that we are only considering partial forward secrecy, and therefore \mathcal{B} does not corrupt both the owner of the test session and the corresponding partner.

Throughout the proof, we call each session to be activated at a party an oracle. We denote the oracles with which \mathcal{B} interacts Π_X^i where X is the name of a party and i is the number of the oracle. We number the oracles such that Π_X^i is the i^{th} oracle created by \mathcal{B} out of all oracles created by \mathcal{B} (i.e. if Π_X^i and Π_Y^j are two oracles, then $i = j$ implies $X = Y$). Also, for any party, X , the identity of that party is denoted e_X . We consider the following series of games with \mathcal{B} .

A.1 Case 1: Partner to the test-session is not corrupted

In this case, the partner to the test session is either not corrupted or only partially corrupted, and the owner of the test session may be fully corrupted, either prior to the session (as in a KCI attack), or after the session expires (as in a forward secrecy attack). This part of the proof uses the following series of games with \mathcal{B} .

Game 0. This game is the same as a real interaction with the protocol. A random bit b is chosen, and when $b = 0$, the real key is returned in answer to the test session query, otherwise a random key from \mathbb{U}_2 is returned.

Game 1. This game is the same as the previous one, except that if two different sessions output exactly the same message and have the same intended partner, the protocol halts.

Game 2. This game is the same as the previous one, except that before the adversary begins, a random value $m \xleftarrow{\$} \{1, 2, \dots, n_{\text{orac}}\}$ is chosen. We call the m^{th} oracle to be activated the target oracle. If the target oracle is not the test oracle, the protocol halts and \mathcal{B} fails and outputs a random bit. We denote the input message to the target oracle with C , the corresponding output message with C^* , the target oracle's owner with T and the target oracle's intended partner with T^* . Note that there may not be a matching test session activated at T^* .

Game 3. In this game, a random value K'^* is chosen. Whenever C^* is used as input to an oracle owned by T^* , the calculation of the key is modified so that K'^* is used in place of $\text{dec}(pk, d_{T^*}, C^*)$; the message output by this oracle is calculated as usual. Similarly, K'^* is used instead of K'_T in the calculation of the session key by T .

The rest of the Game 3 is the same as Game 2. If $b = 1$, a random key from \mathbb{U}_2 is returned. Otherwise, K'^* is used in the computation of the test session as described above.

Game 4. This game is the same as the previous one, except that a random value $K''^* \xleftarrow{\$} \mathbb{U}_1$ is chosen and the use of $\text{Exct}(K'^*)$ is replaced with K''^* .

Game 5. This game is the same as the previous one, except that whenever the value $\text{Expd}_{K''^*}(s')$ for any s' would be used in generating keys, a random value from \mathbb{U}_2 is used instead (the same random value is used for the same value of s' ; a different random value is chosen for different s').

A.2 Analysis of Games 0 to 2:

Let p_{sameMsg} be the probability of two or more sessions outputting the same message. We have

$$1 - p_{\text{sameMsg}} > 1 - \frac{n_{\text{orac}}^2}{b}.$$

Then

$$|\Pr[\sigma_0] - \Pr[\sigma_1]| < \frac{n_{\text{orac}}^2}{2b} \quad (8)$$

when $\frac{1}{2} > \frac{n_{\text{orac}} - 1}{b} > 0$

This can be used to bound τ_0 as follows:

$$\tau_0 = |2\Pr[\sigma_0] - 1| \quad (9)$$

$$\leq 2 \left(|\Pr[\sigma_0] - \Pr[\sigma_1]| + \left| \Pr[\sigma_1] - \frac{1}{2} \right| \right) \quad (10)$$

$$\leq \frac{n_{\text{orac}}^2}{b} + \tau_1 \quad (11)$$

In Game 2, the probability of the protocol halting due to an incorrect choice of m is $1 - \frac{1}{n_{\text{orac}}}$. Whether or not an abortion would occur in this game could be detected in the previous game if it also chose m in the same way. Therefore, we may use Dent's gamehopping technique as described in Dent (2006) to find:

$$\tau_2 = \frac{1}{n_{\text{orac}}} \tau_1 \Rightarrow n_{\text{orac}} \tau_2 = \tau_1 \quad (12)$$

and (11) gives

$$\tau_0 \leq \frac{n_{\text{orac}}^2}{b} + n_{\text{orac}} \tau_2 \quad (13)$$

A.3 Analysis of Game 3:

We now construct adversary \mathcal{A} against the security of the CL-KEM, using \mathcal{B} . \mathcal{A} is constructed such that when it receives the real key for the CL-KEM scheme, the view of \mathcal{B} is the same as in Game 2, but if \mathcal{A} receives a random CL-KEM key, the view of \mathcal{B} is the same as in Game 3. Then, by the security of the CL-KEM scheme, we can claim these games are indistinguishable.

To begin, \mathcal{A} is given the master public key pk . \mathcal{A} passes this value as well as the description of $\text{Exct}(\cdot)$, its key κ and $\{\text{Expd}_K(\cdot)\}_{K \in \mathbb{U}_1}$ to \mathcal{B} . Recall that \mathcal{A} has access to the corresponding oracles $\mathcal{O}_{\text{KeyDer}}(\cdot)$ and $\mathcal{O}_{\text{dec}}(\cdot, \cdot)$.

\mathcal{A} runs as described in Game 2, except that when the target session is activated, \mathcal{A} outputs e_{T^*} as the identity on which it wants to be tested. \mathcal{A} receives a ciphertext C^* for T^* and key K'^* , which may be the decryption of C^* or may be a random CL-KEM key, each with equal probability. \mathcal{A} then uses C^* as the output of the target session, modifies the calculation of keys so that K'^* is used in place of $\text{dec}(pk, \text{KeyDer}(pk, \alpha, e_{T^*}), C^*)$, and uses K'^* instead of K'_T to find the answer to the test session query when $b = 0$.

All legitimate queries made by \mathcal{B} can still be answered by \mathcal{A} using its oracles in as follows.

- A corrupt query on some identity e_X may be answered with $\mathcal{O}_{\text{KeyDer}}(e_X)$.
- A partial corrupt query can be made by on the partner to the test session, T^* . This query can be answered with $\mathcal{O}_{\text{KeyDer}}(T^*)$ for either the certificateless secret value or the ID-based private key but not both, as these queries are allowed in a certificateless KEM.
- \mathcal{A} must maintain the session state of each oracle so that it may be returned in answer to session state reveal queries (session state reveal is not allowed on the test session or its matching session).
- Any message C_X to any party (including T^*) with identity e_X may be decrypted using $\mathcal{O}_{\text{dec}}(e_X, C_X)$ to generate keys for reveal session key queries and the test query. Any party other than T^* can be decrypted with the private key that \mathcal{A} generated for that party. A message M to T^* may be decrypted using \mathcal{O}_{dec} .

When \mathcal{B} halts and outputs its bit b' , \mathcal{A} halts and outputs $1 - b'$. The probability that \mathcal{A} is correct is $\Pr[\sigma_2]$ when K'^* is the real key for the CL-KEM message, and $1 - \Pr[\sigma_3]$ when K'^* is not the key for the CL-KEM message. We can then find that:

$$\text{Adv}_{\mathcal{E}, \mathcal{A}}^{\text{CL-KEM-CCA}}(k) = \quad (14)$$

$$\left| 2 \left(\frac{1}{2} (\Pr[\sigma_2] + 1 - \Pr[\sigma_3]) \right) - 1 \right| \quad (15)$$

$$= \Pr[\sigma_2] - \Pr[\sigma_3] \quad (16)$$

$$\tau_2 = |2\Pr[\sigma_2] - 1| \quad (17)$$

$$\leq |2\Pr[\sigma_2] - 2\Pr[\sigma_3]| + |2\Pr[\sigma_3] - 1| \quad (18)$$

$$\tau_2 \leq 2\text{Adv}_{\mathcal{E}, \mathcal{A}}^{\text{CL-KEM-CCA}}(k) + \tau_3 \quad (19)$$

A.4 Analysis of Game 4

We now consider an adversary, \mathcal{D} , against the security of the randomness extraction function. This adversary runs a copy of \mathcal{B} and interacts with \mathcal{B} in such a manner that it is the same as when \mathcal{B} interacts with either Game 3 or 4. \mathcal{D} receives a key κ for the randomness extraction function and a value R_1 such that either $R_1 = \text{Exct}(X)$ for some $X \xleftarrow{\$} \mathcal{K}$ or $R_1 \xleftarrow{\$} \mathbb{U}_1$. \mathcal{D} sets κ to be the public parameter used to key the randomness extraction function, and chooses the other public parameters according to the protocol. \mathcal{D} runs as described for Game 4, except that \mathcal{D} uses R_1 in place of K'^* . When \mathcal{B} outputs its guess of the bit b , \mathcal{D} outputs that $R_1 = \text{Exct}(X)$ for some X if \mathcal{B} is correct, and \mathcal{D} outputs that $R_1 \xleftarrow{\$} \mathbb{U}_1$ otherwise. The probability that \mathcal{D} is correct is $\frac{1}{2} (\Pr[\sigma_3] + 1 - \Pr[\sigma_4])$. By the security of the randomness extraction function, we have:

$$\epsilon \geq |2\Pr[\mathcal{D} \text{ correct}] - 1| \quad (20)$$

$$= |\Pr[\sigma_3] - \Pr[\sigma_4]| \quad (21)$$

$$\tau_3 = |2\Pr[\sigma_3] - 1| \quad (22)$$

$$\leq |2\Pr[\sigma_3] - 2\Pr[\sigma_4]| + |2\Pr[\sigma_4] - 1| \quad (23)$$

$$\leq 2\epsilon + \tau_4 \quad (24)$$

A.5 Analysis of Game 5

Now, we consider another adversary, \mathcal{D}' , this time against the randomness expansion (or pseudorandom) function family $\{\text{Expd}_K(\cdot)\}_{K \in \mathbb{U}_1}$. We define \mathcal{D}' to run a copy of \mathcal{B} , and to interact with \mathcal{B} in such a

manner that it is the same as when \mathcal{B} interacts with with either Game 4 or 5. \mathcal{D}' receives the definition of the function family $\{\text{Expd}_K(\cdot)\}_{K \in \mathbb{U}_1}$, and an oracle $\mathcal{O}(\cdot)$ which is either $\text{Expd}_K(\cdot)$ for some value of K unknown to \mathcal{D}' or a truly random function. \mathcal{D}' runs a copy of the protocol for \mathcal{B} in the same way as described for Game 4, except that whenever the value $\text{Expd}_{K'^*}(s')$ for any s' would be used in generating keys, \mathcal{D}' uses the value $\mathcal{O}(s')$ instead. When \mathcal{B} outputs its guess of the bit b , \mathcal{D}' outputs that its oracle is a member of the given function family if \mathcal{B} is correct, and \mathcal{D}' outputs that its oracle is a truly random function otherwise. The probability that \mathcal{D}' is correct is $\frac{1}{2} (\Pr[\sigma_4] + 1 - \Pr[\sigma_5])$. By the security of the randomness expansion function we have:

$$\text{Adv}_{\mathcal{F}, \mathcal{C}}^{\text{p-rand}}(k) \geq |2\Pr[\mathcal{D}' \text{ correct}] - 1| \quad (25)$$

$$= |\Pr[\sigma_4] - \Pr[\sigma_5]| \quad (26)$$

$$\tau_4 = |2\Pr[\sigma_4] - 1| \quad (27)$$

$$\leq |2\Pr[\sigma_4] - 2\Pr[\sigma_5]| + |2\Pr[\sigma_5] - 1| \quad (28)$$

$$\leq 2\text{Adv}_{\mathcal{F}, \mathcal{C}}^{\text{p-rand}}(k) + \tau_5 \quad (29)$$

In Game 5, let us denote the key returned in the test session query with $R_1 \oplus \text{Expd}_{K'^*}(s')$ when $b = 0$, and R_2 when $b = 1$, where R_1 and R_2 are chosen uniformly at random from \mathbb{U}_2 . Now, R_2 is chosen independently of all other values in the protocol, so \mathcal{B} can gain no information about R_2 directly; \mathcal{B} can only gain information about R_2 by determining whether $b = 0$ or $b = 1$. Furthermore, when $b = 0$, unless \mathcal{B} can gain some information about R_1 , the response to the test session query also looks random and is therefore indistinguishable from the case when $b = 1$.

To gain information about R_1 from a source other than the test session query response, \mathcal{B} must obtain the key of a session that has also used R_1 in the generation of its key. Now, if R_1 is used in the generation of a session's key, then that session must have had the same session identifier, and hence exchanged the same messages as the test session. Therefore, the session is either owned by T^* with intended partner T and received C^* as part of its input or is owned by T with intended partner T^* and had C^* as part of its output. However, such a session owned by T^* will match the test session and so not be subject to reveal key queries. Hence, \mathcal{B} can gain no information about R_1 , and so \mathcal{B} can gain no information about b in Game 5, and therefore

$$\tau_5 = 0$$

Combining the results in equations (13), (19), (24) and (29) we conclude:

$$\tau_0 \leq \frac{n_{\text{orac}}^2}{b} + 2n_{\text{orac}} \left(\text{Adv}_{\mathcal{E}, \mathcal{A}}^{\text{CL-KEM-CCA}}(k) + \epsilon + \text{Adv}_{\mathcal{F}, \mathcal{C}}^{\text{p-rand}}(k) \right) \quad (30)$$

A.6 Case 2: Partner to the test session is corrupted

Recall that we only consider partial weak forward secrecy for Protocol 1. Consequently we require that

1. the owner of the test session may only be partially corrupted by the adversary,
2. the adversary be passive in the protocol corresponding to the test session; that is there exists

a matching session to the test session at the intended partner by the time that the test session query is issued by the adversary; and

3. the partner to the test session be fully corrupted only after the matching session expires.

For this part of the proof we set up the following series of 6 games with \mathcal{B} . Game 0 and 1 are the same as in Case 1. Game 2 and 3 are analogous to Game 2 and 3 in Case 1 except that now our target oracle is the partner to the test session, and it is its input to the session key that is substituted by a random value. Game 4 and 5, which are used to prove the security of the session key derivation mechanism via randomness extraction and expansion also remain essentially the same.

Game 0. This game is the same as a real interaction with the protocol. A random bit b is chosen, and when $b = 0$, the real key is returned in answer to the test session query, otherwise a random key from \mathbb{U}_2 is returned.

Game 1. This game is the same as the previous one, except that if two different sessions output exactly the same message and have the same intended partner, the protocol halts.

Game 2. This game is the same as the previous one, except that before the adversary begins, a random value $m \stackrel{\$}{\leftarrow} \{1, 2, \dots, n_{\text{orac}}\}$ is chosen. We call the m^{th} oracle to be activated the target oracle. If the target oracle is not the *partner* to the test oracle, the protocol halts and \mathcal{B} fails and outputs a random bit. We denote the input message to the target oracle with C^* , the corresponding output message with C , the target oracle's owner with T^* and the target oracle's intended partner with T .

Game 3. In this game, a random value $K' \stackrel{\$}{\leftarrow} \mathcal{K}$ is chosen. Further a bit $c \in_R \{0, 1\}$ is chosen as a guess as to whether \mathcal{B} fully corrupts T or T^* . Whenever C is used as input to an oracle owned by T , the calculation of the key is modified so that K' is used in place of $\text{dec}(pk, d_T, C)$; the message output by this oracle is calculated as usual. Similarly, K' is used instead of K'_T in the calculation of the session key by T^* .

The rest of the Game 3 is the same as Game 2. If $b = 1$, a random key from \mathbb{U}_2 is returned. Otherwise, K' is used in the computation of the test session as described above.

Game 4. This game is the same as the previous one, except that a random value $K'' \stackrel{\$}{\leftarrow} \mathbb{U}_1$ is chosen and the use of $\text{Exct}(K')$ is replaced with K'' .

Game 5. This game is the same as the previous one, except that whenever the value $\text{Expd}_{K''}(s')$ for any s' would be used in generating keys, a random value from \mathbb{U}_2 is used instead (the same random value is used for the same value of s' ; a different random value is chosen for different s').

In the analysis that follows, we denote with σ'_i the event that the adversary is successful in Game i and with τ'_i the corresponding advantage.

A.7 Analysis of Games 0 to 2:

This analysis is the same as the that of Games 0 to 2 in Case 1. Thus,

$$\tau'_0 \leq \frac{n_{\text{orac}}^2}{b} + n_{\text{orac}}\tau'_2 \quad (31)$$

A.8 Analysis of Game 3:

This analysis is very similar to that of Game 3 in Case 1. We construct adversary \mathcal{A} against the security of the CL-KEM, using \mathcal{B} . \mathcal{A} is given the master public key pk and passes this value as well as the description of $\text{Exct}(\cdot)$, its key κ and $\{\text{Expd}_K(\cdot)\}_{K \in \mathbb{U}_1}$ to \mathcal{B} . \mathcal{A} runs as described in Game 2, except that when the target oracle is activated, \mathcal{A} outputs e_T as the identity on which it wants to be tested. \mathcal{A} receives a ciphertext C for T and key K' , which may be the decryption of C or may be a random CL-KEM key, each with equal probability. \mathcal{A} then uses C as the output of the target oracle (and hence input to the test session). \mathcal{A} modifies the calculation of keys so that K' is used in place of $\text{dec}(pk, \text{KeyDer}(pk, \alpha, e_T), C)$, and uses K' instead of K'_T to find the answer to the test session query when $b = 0$.

All legitimate queries made by \mathcal{B} can still be answered by \mathcal{A} using its oracles as follows.

- A corrupt query on some identity e_X may be answered with $\mathcal{O}_{\text{KeyDer}}(e_X)$.
- A corrupt query targeting either the certificate-less secret value or the identity based private key (but not both) for the owner of the test session can also be answered with $\mathcal{O}_{\text{KeyDer}}(T)$. (Recall that the owner of the test session T must not be fully corrupted).
- \mathcal{A} must maintain the session state of each oracle so that it may be returned in answer to session state reveal queries (session state reveal is not allowed on the test session or its matching session).
- Any message C_X to any party (including T) with identity e_X may be decrypted using $\mathcal{O}_{\text{dec}}(e_X, C_X)$ to generate keys for reveal session key queries and the test query.

When \mathcal{B} halts and outputs its bit b' , \mathcal{A} halts and outputs $1 - b'$. As in Case 1, the probability that \mathcal{A} is correct is $\Pr[\sigma'_2]$ when K' is the real key for the CL-KEM message, and $1 - \Pr[\sigma'_3]$ when K' is not the key for the CL-KEM message. Hence,

$$\tau'_2 \leq 2\text{Adv}_{\mathcal{E}, \mathcal{A}}^{\text{CL-KEM-CCA}}(k) + \tau'_3 \quad (32)$$

A.9 Analysis of Game 4

This is the same as in Case 1. Thus,

$$\tau'_3 \leq 2\epsilon + \tau'_4 \quad (33)$$

A.10 Analysis of Game 5

This is the same as in Case 1. Thus,

$$\tau'_4 \leq 2\text{Adv}_{\mathcal{F}, \mathcal{C}}^{\text{p-rand}}(k) + \tau'_5 \quad (34)$$

A.11 Combining Results

Again using the same reasoning as in Case 1, we conclude that $\tau'_5 = 0$, and therefore combining equations (31), (32), (33) and (34) we have:

$$\tau'_0 \leq \frac{n_{\text{orac}}^2}{b} + 2n_{\text{orac}} \left(\text{Adv}_{\mathcal{E}, \mathcal{A}}^{\text{CL-KEM-CCA}}(k) + \epsilon + \text{Adv}_{\mathcal{F}, \mathcal{C}}^{\text{p-rand}}(k) \right) \quad (35)$$

This is the same advantage as in Case 1 and hence Theorem 1 follows.

B Proof of Theorem 2:

The security difference between Protocol 1 and 2 is that the latter provides full WFS, i.e. in addition to the adversarial capabilities considered in the proof of Theorem 1, we now allow the adversary to corrupt both parties to the test session. It is natural then to consider the proof of Theorem 2 in two parts: the first part where the adversary does not corrupt both parties to the test session, and the second part where it does. Then, the first part is essentially identical to the proof of Theorem 1. The only difference is that in the analysis of Game 3 (in both Case 1 and 2) \mathcal{A} needs to simulate the extra Diffie-Hellman values, which \mathcal{A} can easily do for all sessions, including the test session. (Note that \mathcal{A} will always choose at least one of Y_A or Y_B , hence it can always compute K''_{AB}).

We deal now with the second part of the proof, where the adversary corrupts the two partners to the test session. Note however that the adversary is restricted to being passive during the protocol run corresponding to the test session – a consequence of only being able to achieve weak forward secrecy in one round. As we will see below this allows us to inject a challenge Decisional Diffie-Hellman triplet into the test session.

The second part of the proof allows any party to be corrupted. It considers the following four games with \mathcal{B} .

Game 0. This game is the same as a real interaction with the protocol. A random bit b is chosen, and when $b = 0$, the real key is returned in answer to the test session query, otherwise a random key from \mathbb{U}_2 is returned.

Game 1. This game is the same as the previous one, except that before the adversary begins, random values $j, j^* \stackrel{\$}{\leftarrow} \{1, 2, \dots, n_{\text{orac}}\}$ are chosen. Let T and T^* be the owners of the j^{th} and $j^{*\text{th}}$ sessions respectively. If the j^{th} session at T is not the test session or if the output of the $j^{*\text{th}}$ session at T^* is not used as input to the test session, then the protocol halts and \mathcal{B} fails and outputs a random bit. Furthermore, the session key of the test session is calculated as usual except that $\text{Exct}(h)$ for $h \in_R \langle f \rangle$ replaces K''_{TT^*} . The test session's matching session has its key set to the same value (i.e. a random test session key is always returned, no matter what the value of b).

Game 2. This game is the same as the previous one except that $\text{Exct}(h)$ is replaced with $K'' \in_R \mathbb{U}_1$

Game 3. This game is the same as the previous one, except that whenever the value $\text{Expd}_{K''}(s')$ for any s' would be used in generating keys, a random value from \mathbb{U}_2 is used instead (the same random value is used for the same value of s' ; a different random value is chosen for different s').

B.1 Analysis Game 1

We now construct adversary \mathcal{A} against the DDH problem, using \mathcal{B} . \mathcal{A} is constructed such that provided \mathcal{A} does not have to abort the protocol as specified in Game 1 then if \mathcal{A} 's input is from \mathcal{DH}_F , the view of \mathcal{B} is the same as in Game 0, but if \mathcal{A} 's input is from \mathcal{R}_F , the view of \mathcal{B} is the same as in Game 1. Then, by Assumption 4, we can claim these games are indistinguishable.

To begin, \mathcal{A} generates all the protocol parameters and passes the public parameters to \mathcal{B} .

Let (f^a, f^b, h) be \mathcal{A} 's challenge DDH inputs (with h either f^c or f^{ab}). When the j^{th} and the $j^{*\text{th}}$ sessions

are activated, \mathcal{A} uses its inputs f^a and f^b instead of the values Y_T and Y_{T^*} when it generates the outputs of these sessions. Apart from this change, all session inputs and outputs are generated according to the protocol specification.

When the test session query is made, \mathcal{A} uses $\text{Exct}(h)$ in place of K''_{TT^*} when calculating the real test session key.

\mathcal{A} is able to answer all other queries correctly since it knows all of the system parameters and all of the session states. \mathcal{A} outputs 1 if \mathcal{B} is correct, 0 otherwise.

The probability that \mathcal{A} will not have to abort the protocol as described in Game 1 is $\frac{1}{n_{\text{orac}}^2}$. Hence, by the game hopping technique from Dent (2006) and using a similar logic to that shown in (26) to (29) we have that:

$$\tau_0 \leq 2n_{\text{orac}}^2 \text{Adv}_{F, \mathcal{D}}^{\text{ddh}}(k) + \tau_1 \quad (36)$$

B.2 Analysis Game 2

The analysis of Game 2 is the same as that of Game 4 of Case 1. Thus,

$$\tau_1 \leq 2\epsilon + \tau_2 \quad (37)$$

B.3 Analysis Game 3

The analysis of Game 3 is the same as that of Game 5 of Case 1 and 2. Thus,

$$\tau_2 \leq 2\text{Adv}_{\mathcal{F}, \mathcal{C}}^{\text{p-rand}}(k) + \tau_3 \quad (38)$$

Since the test session key is masked with a random value, and that value is independent of all other sessions, \mathcal{B} has no advantage in Game 3 ($\tau_3 = 0$). The value is independent because only the test session has the test session's session id.

B.4 Combining Results:

Let E be the event that the test session has a matching session by the time \mathcal{B} finishes, and let σ be the event that \mathcal{B} guesses b correctly in Protocol 2. Then we have:

$$\begin{aligned} \text{Adv}_{\mathcal{B}}^{\text{sk}}(k) &= |2\Pr[\sigma] - 1| \\ \Pr[\sigma] &= \Pr[\sigma|E]\Pr[E] + \Pr[\sigma|\neg E]\Pr[\neg E] \\ &= \Pr[\sigma|\neg E] + \Pr[E](\Pr[\sigma|E] - \Pr[\sigma|\neg E]). \end{aligned}$$

Therefore,

$$\begin{aligned} \min(\Pr[\sigma|\neg E], \Pr[\sigma|E]) &\leq \Pr[\sigma] \\ &\leq \max(\Pr[\sigma|\neg E], \Pr[\sigma|E]) \end{aligned}$$

and

$$\text{Adv}_{\mathcal{B}}^{\text{sk}}(k) \leq \max\left(\text{Adv}_{\mathcal{B}}^{\text{sk}}(k)|E, \text{Adv}_{\mathcal{B}}^{\text{sk}}(k)|\neg E\right)$$

and so we can combine (30), and (35)-(38) to find:

$$\begin{aligned} \text{Adv}_{\mathcal{B}}^{\text{sk}}(k) &\leq \\ &\max\left(2n_{\text{orac}}^2 \text{Adv}_{F, \mathcal{D}}^{\text{ddh}}(k) + 2\epsilon + 2\text{Adv}_{\mathcal{F}, \mathcal{C}}^{\text{p-rand}}(k), \right. \\ &\quad \left. \frac{n_{\text{orac}}^2}{b} + 2n_{\text{orac}} \left(\text{Adv}_{\mathcal{E}, \mathcal{A}}^{\text{CL-KEM-CCA}}(k) + \epsilon + \right. \right. \\ &\quad \left. \left. \text{Adv}_{\mathcal{F}, \mathcal{C}}^{\text{p-rand}}(k) \right) \right). \end{aligned}$$

Combinatorial Multicollision Attacks on Generalized Iterated Hash Functions

Kimmo Halunen¹

Juha Kortelainen²

Tuomas Kortelainen³

¹ Oulu University Secure Programming Group
Department of Electrical and Information Engineering
University of Oulu
Email: khalunen@ee.oulu.fi

² Department of Information Processing Science
University of Oulu
Email: juha.kortelainen@oulu.fi

³ Mathematics Division
Department of Electrical and Information Engineering
University of Oulu
Email: tukorte@paju.oulu.fi

Abstract

We develop a word combinatorial approach to multicollisions in generalized iterated hash functions. The work rests on the notable discoveries of A. Joux and on generalizations provided by M. Nandi and D. Stinson as well as J. Hoch and A. Shamir. New results and improvements to some previously published ones are produced. We also wish to unify the diverse notations and bring the results together by applying concepts of combinatorics on words. A multicollision attack method informally described by Hoch and Shamir is presented as a statistical procedure and analyzed in detail.

Keywords: iterated hash functions, multicollisions, word combinatorics

1 Introduction

Iterated hash functions have been the most successful method for constructing fast and secure hash functions. The underlying principle proposed by Merkle and Damgård (Damgård 1989, Merkle 1990) is quite simple and easy to implement. However, most of the modern hash functions built on this foundation have been found flawed (Wang & Yu 2005, Wang et al. 2005, Klima 2005, Stevens 2006, Dobbertin 1998). Many of these flaws come from the weaknesses in the underlying compression functions. In recent years more rigorous theoretical study has also found some weaknesses in the iterative structure itself.

One of the most remarkable results on the iterative structure was Joux's method concerning multicollisions in iterated hash functions (Joux 2004), which has been used to disprove some of the assumptions on hash function security. Furthermore, these achievements concerning multicollisions have been generalized by Nandi and Stinson (Nandi & Stinson 2007) and later by Hoch and Shamir (Hoch & Shamir 2006). These results show that Joux's method can be applied against a more general class of iterated hash functions.

Copyright ©2010, Australian Computer Society, Inc. This paper appeared at the Twenty-Ninth Australasian Information Security Conference (AISC2010), Brisbane, Australia. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 105, Colin Boyd and Willy Susilo, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

In this paper, we show one way to formulate the theoretical results concerning hash functions and especially multicollisions for iterated hash functions in a well established mathematical system. The notations and basic theory of (word) combinatorics and algebra are extensively applied. In (Hoch & Shamir 2006) a method to construct multicollisions on Iterated Concatenated and Expanded (ICE) Hash Functions is introduced. The description of the method is informal and difficult, if not impossible, to understand in detail. We wish to give a rigorous mathematical treatment to this method and point out certain deficiencies in the original version of it. A fairly detailed complexity analysis of the respective attack construction is provided.

The paper is organised in the following way. The second section introduces the basic definitions of word combinatorics and partial orders. The third section shows how iterated hash functions and multicollisions can be depicted in this theoretical setting; the earlier work committed in the field is reviewed and the structure of the attack schema on generalized iterated hash functions described. In the fourth section we prove the combinatorial results needed for the construction of a tractable multicollision attack on so called bounded generalized iterated hash functions. The fifth section contains the exact exposition of the multicollision attack. In the final sections, we discuss our results, draw some conclusions from our research and give possible future research proposals.

2 Basics on words, languages and partial orders

We encourage the reader to skip over the first section and refer to the basic concepts only as the need arises.

2.1 Words and languages

Let $\mathbb{N} = \{0, 1, 2, \dots\}$ be the set of all nonnegative integers and $\mathbb{N}_+ = \mathbb{N} \setminus \{0\}$. For each $l \in \mathbb{N}_+$, denote by \mathbb{N}_l the set of l first positive integers: $\mathbb{N}_l = \{1, 2, \dots, l\}$. For each finite set S , let $|S|$ be the cardinality of S , i.e. the number of elements in S .

An *alphabet* is any finite nonempty set of abstract symbols called *letters*. Let A be an alphabet. A *word* (over A) is any finite sequence of symbols in A . Thus, assuming that w is a word over A , we can write $w = x_1x_2 \cdots x_n$, where $n \in \mathbb{N}$ and $x_i \in A$ for

$i = 1, 2, \dots, n$. Above n is the *length* $|w|$ of w . Notice that n may be equal to zero; then w is the *empty word*, denoted by ϵ , that contains no letters. By $|w|_a$ we mean the number of occurrences of the letter a in w . Define the *alphabet* of w by $\text{alph}(w) = \{a \in A \mid |w|_a > 0\}$. Obviously, $\text{alph}(\epsilon) = \emptyset$. Let A^* (A^+ , resp.) be the set of all words (nonempty words, resp.) over A . By A^n , $n \in \mathbb{N}$, we mean the set of all words of length n over A . The *product of the words* u and v in A^* is the word uv obtained by writing u and v after one another. Clearly product defines a binary operation \cdot in A^* : $u \cdot v = uv$ for all $u, v \in A^*$. In algebraic terms (A^* , \cdot) is a *free monoid* and (A^+ , \cdot) is a *free semigroup*. The word u is a *subword* of w if there exist $m \in \mathbb{N}$ and $x_0, u_1, x_1, \dots, u_m, x_m \in A^*$ such that $w = x_0 u_1 x_1 \dots u_m x_m$ where $u = u_1 u_2 \dots u_m$. A subword u of w is a *factor* of w if $w = x_0 u x_1$ for some $x_0, x_1 \in A^*$. A *permutation* of an alphabet A is any word u in A^+ such that $|u|_a = 1$ for each $a \in A$.

Let A and B be alphabets. A mapping $h: A^* \rightarrow B^*$ is a (*monoid*) *morphism* if $h(uv) = h(u)h(v)$ for each $u, v \in A^*$. If $B \subseteq A$, then the *projection morphism* from A^* into B^* , denoted by π_B^A (or π_B , when A is understood), is defined by $\pi_B^A(b) = b$ for each $b \in B$ and $\pi_B^A(a) = \epsilon$ for each $a \in A \setminus B$.

A *language* (over the alphabet A) is any set of words L (such that $L \subseteq A^*$). Let L and T be languages. The *product of L and T* is the language $L \cdot T = \{uv \mid u \in L, v \in T\}$. We write often LT instead $L \cdot T$. The product is, by induction, easily generalized to concern more than two languages: $L_1 L_2 \dots L_{n+1} = (L_1 \cdot L_2 \dots L_n) \cdot L_{n+1}$ for all $n \in \mathbb{N}_+$ and languages L_1, L_2, \dots, L_{n+1} . Let the *n th power* of L be the language $L^n = L_1 L_2 \dots L_n$, where $L_i = L$ for $i = 1, 2, \dots, n$, $n \in \mathbb{N}_+$. Denote $L^+ = \cup_{i=1}^{\infty} L^i$. In the case of a singleton language $\{w\}$ (consisting of the one word w only), we write w^+ instead of $\{w\}^+$.

2.2 Partial orders

A binary relation R of the set X is a *partial order* (in X) if it is irreflexive ($\forall x \in X : (x, x) \notin R$), antisymmetric ($\forall x, y \in X : (x, y) \in R \Rightarrow (y, x) \notin R$) and transitive ($\forall x, y, z \in X : (x, y) \in R \wedge (y, z) \in R \Rightarrow (x, z) \in R$).

Let \prec be a partial order in X . Call (X, \prec) a *partially ordered set*. The elements $x, y \in X$ are *incomparable* (in (X, \prec)) if neither $x \prec y$ nor $y \prec x$ holds. The nonempty finite sequence x_1, x_2, \dots, x_n of elements of X is a *chain* of (X, \prec) if $x_i \prec x_{i+1}$ for $i \in \{1, 2, \dots, n-1\}$. Above $n \in \mathbb{N}_+$ is the *length* of the chain $x_1 \prec x_2 \dots \prec x_n$. For each chain c of (X, \prec) , let $|c|$ be the length of c . An (indexed) set of chains $\{c_i\}_{i \in I}$ is a *chain decomposition* of (X, \prec) , if $\{C_i\}_{i \in I}$ is a partition of X , where $C_i = \{x \in X \mid x \text{ occurs in the chain } c_i\}$. Obviously, $\{(x)\}_{x \in X}$ is a (trivial) chain decomposition of (X, \prec) .

Consider now a finite partially ordered set (X, \prec) , i.e., a partially ordered set such that X is finite. The *maximum number of incomparable elements* of (X, \prec) is the cardinality of the largest set $Y \subseteq X$ such that the elements of Y are pairwise incomparable. The *minimum chain decomposition size* of (X, \prec) is the smallest number $m \in \mathbb{N}_+$ such that there exist chains c_1, c_2, \dots, c_m of (X, \prec) for which $\{c_i\}_{i=1}^m$ is a chain decomposition of (X, \prec) . Finally, let *maximum chain length* of (X, \prec) be the greatest number $m \in \mathbb{N}_+$ such that there exists a chain of length m in (X, \prec) .

An important connection between the two concepts defined above is stated in a famous theorem of Dilworth (Dilworth 1950).

Theorem 1 (Dilworth's Theorem). *Let (X, \prec) be a finite, partially ordered set. Then the maximum*

number of incomparable elements of (X, \prec) is equal to the minimum chain decomposition size of (X, \prec) .

Let now α be a nonempty word. Define the binary relation \prec_α of $\text{alph}(\alpha)$ as follows. For each $a, b \in \text{alph}(\alpha)$, let $a \prec_\alpha b$ hold if and only if $a \neq b$ and each occurrence of a in α happens before each occurrence of b in α . Certainly if $a \prec_\alpha b$, then there exist words α_1 and α_2 such that $\alpha = \alpha_1 \alpha_2$ and $|\alpha_1|_b = |\alpha_2|_a = 0$. Obviously, \prec_α is irreflexive, antisymmetric and transitive, so $(\text{alph}(\alpha), \prec_\alpha)$ is a partially ordered set. Call the elements of a nonempty set $A \subseteq \text{alph}(\alpha)$ *independent* (with respect to \prec_α) if they form a chain in $(\text{alph}(\alpha), \prec_\alpha)$.

3 Hash functions and collisions

In this section we give basic definitions of hash functions and multicollisions using a fresh and rigorous notation. The principles of (iterative) hash functions were, however, presented already in (Damgård 1989) and advanced ideas on multicollisions appear in (Joux 2004), (Nandi & Stinson 2007), and (Hoch & Shamir 2006). We finish the section by a general (stepwise) description of the studied attack method.

3.1 Fundamental concepts

By *block representation* of a message, we mean the division and padding of the message into blocks of equal size. We may certainly without loss of generality assume that all our messages are written in the binary alphabet $\{0, 1\}$ and given in a block representation form.

Definition 1. A hash function of length n (where $n \in \mathbb{N}_+$) is a mapping $f: \{0, 1\}^* \rightarrow \{0, 1\}^n$.

An ideal hash function $f: \{0, 1\}^* \rightarrow \{0, 1\}^n$ is a *variable input length random oracle* (VIL-RO for short): for each $x \in \{0, 1\}^*$, the value $f(x) \in \{0, 1\}^n$ is chosen uniformly at random.

Let $k \in \mathbb{N}_+$. A *k -collision* in the hash function $f: \{0, 1\}^* \rightarrow \{0, 1\}^n$ is a set $A \subseteq \{0, 1\}^*$ such that $|A| = k$ and $f(x) = f(y)$ for all $x, y \in A$. A 2-collision (on f) is also called a collision.

A *k -collision attack* (algorithm) on a hash function f can loosely be characterized to be a probabilistic process (based on the birthday problem) that finds a k -collision on f with some nonnegligible probability. The *complexity* of the attack can be measured, for instance, with respect to the number of messages the hash values of which have to be determined to carry out the attack successfully.

According to the (generalized) *birthday paradox*, an k -collision can be found (with probability approx. 0.5) by hashing $(k!)^{\frac{1}{k}} 2^{\frac{n(k-1)}{k}}$ messages (Suzuki et al. 2008). In the case $k = 2$ this gives $\sqrt{2} \cdot 2^{\frac{n}{2}}$ hashings. Intuitively most of us would expect the number to be around 2^{n-1} .

Definition 2. A compression function (of block size m and length n) is a mapping $f: \{0, 1\}^n \times \{0, 1\}^m \rightarrow \{0, 1\}^n$ where $m, n \in \mathbb{N}_+$, $m > n$.

Again, an ideal compression function $f: \{0, 1\}^n \times \{0, 1\}^m \rightarrow \{0, 1\}^n$ is a *fixed input length random oracle* (FIL-RO for short): for each $h \in \{0, 1\}^n$ and $y \in \{0, 1\}^m$, the value $f(h, y) \in \{0, 1\}^n$ is chosen uniformly at random.

Throughout the paper m and n are arbitrary but fixed positive integers such that $m > n$ and $f: \{0, 1\}^n \times \{0, 1\}^m \rightarrow \{0, 1\}^n$ is a given compression function (which is not necessarily a FIL-RO).

Define the function $f^+ : \{0, 1\}^n \times (\{0, 1\}^m)^+ \rightarrow \{0, 1\}^n$ inductively as follows. Let $h \in \{0, 1\}^n$, $y_1 \in \{0, 1\}^m$, and $y_2 \in (\{0, 1\}^m)^+$. Then $f^+(h, y_1) = f(h, y_1)$ and $f^+(h, y_1 y_2) = f^+(f(h, y_1), y_2)$.

Let $l \in \mathbb{N}_+$ and $\alpha \in \mathbb{N}_l^+$. Write α in the form $\alpha = a_1 a_2 \cdots a_s$, where $s \in \mathbb{N}_+$ and $a_i \in \mathbb{N}_+$ for $i = 1, 2, \dots, s$. For each $u \in \{0, 1\}^{ml}$ such that $u = u_1 u_2 \cdots u_l$ where $u_j \in \{0, 1\}^m$ for $j = 1, 2, \dots, l$, define

$$u(\alpha) = u_{a_1} u_{a_2} \cdots u_{a_s}.$$

Note that $u(\alpha)$ above is constructed of the blocks of u by writing them in the order determined by the word α .

Define now the *iterated compression function* $f_\alpha : \{0, 1\}^n \times \{0, 1\}^{ml} \rightarrow \{0, 1\}^n$ (based on α and f) by $f_\alpha(h, u) = f^+(h, u(\alpha))$. Thus, given $h_0 \in \{0, 1\}^n$ and $u = u_1 u_2 \cdots u_l$, where $u_j \in \{0, 1\}^m$ for $j = 1, 2, \dots, l$, and setting $h_i = f(h_{i-1}, u_{a_i})$ for $i = 1, 2, \dots, s$, the equality $f_\alpha(h, u_1 u_2 \cdots u_l) = h_s$ obviously holds. In the presentation $u = u_1 u_2 \cdots u_l$, call u_j the j th block of u , $j = 1, 2, \dots, l$.

Given $k \in \mathbb{N}_+$ and $h_0 \in \{0, 1\}^n$, a k -collision (with initial value h_0) in the iterated compression function f_α is a set $A \subseteq \{0, 1\}^{ml}$ such that $|A| = k$ and for all $u, v \in A$, $|u| = |v|$ and $f_\alpha(h_0, u) = f_\alpha(h_0, v)$. We say that A is *nontrivial* if for all $u, v \in A$ and each $j \in \mathbb{N}_l \setminus \text{alph}(\alpha)$, the j th block of u is equal to the j th block of v .

A k -collision attack on f_α is a probabilistic procedure (often based on the birthday problem) that, by making queries on f and receiving respective responses, finds a nontrivial k -collision on f_α with probability one for any initial value h_0 . We wish to remind that a *query* on f is any pair $(h, x) \in \{0, 1\}^n \times \{0, 1\}^m$ that serves as an input to f ; the respective *response* is the value $f(h, x)$. The *complexity of an k -collision attack on f_α* is the expected number of queries on f required to get a k -collision. Call the attack *tractable* if its complexity is in $O(2^{\frac{n}{2}})$.

Finally, we are ready to characterize a generalized iterated hash function. For each $l \in \mathbb{N}_+$, let $\alpha_l \in \mathbb{N}_l^+$ be such that $\text{alph}(\alpha_l) = \mathbb{N}_l$. Denote $\hat{\alpha} = (\alpha_1, \alpha_2, \dots)$. Define the *generalized iterated hash function* $H_{\hat{\alpha}, f} : \{0, 1\}^n \times (\{0, 1\}^m)^+ \rightarrow \{0, 1\}^n$ (based on $\hat{\alpha}$ and f) as follows: Given the initial value $h_0 \in \{0, 1\}^n$ and the message x the block representation of which consists of l blocks, let $H_{\hat{\alpha}, f}(h_0, x) = f_{\alpha_l}(h_0, x)$.

Given $k \in \mathbb{N}_+$ and $h_0 \in \{0, 1\}^n$, a k -collision in the generalized iterated hash function $H_{\hat{\alpha}, f}$ is a set $A \subseteq (\{0, 1\}^m)^+$ such that $|A| = k$ and for all $u, v \in A$, $|u| = |v|$ and $H_{\hat{\alpha}, f}(h_0, u) = H_{\hat{\alpha}, f}(h_0, v)$. Suppose now that A is a k -collision set in $H_{\hat{\alpha}, f}$ with initial value h_0 . Let $l \in \mathbb{N}_+$ be such that $A \subseteq \{0, 1\}^{ml}$, i.e., the length in blocks of messages in A is l . Then, by definition, for each $u, v \in A$, the equality $f_{\alpha_l}(h_0, u) = f_{\alpha_l}(h_0, v)$ holds. Since the $\text{alph}(\alpha_l) = \mathbb{N}_l$, the set A is a nontrivial k -collision in f_{α_l} with initial value h_0 .

3.2 Earlier work

In (Joux 2004) Joux considers an iterated hash functions $H : (\{0, 1\}^m)^+ \rightarrow \{0, 1\}^n$ based on the compression function f and the equality $H(u) = f^+(IV, u)$ where $IV \in \{0, 1\}^n$ is a fixed *initial value*. He shows that, for each $r \in \mathbb{N}_+$ there exists a probabilistic procedure of complexity $O(r 2^{n/2})$ producing a 2^r -collision in H . The idea of Joux is simple and ingenious: applying the birthday paradox, a sequence of message sets $\{u_{11}, u_{12}\}, \{u_{21}, u_{22}\}, \dots, \{u_{r1}, u_{r2}\}$ such that $u_{i1} \neq u_{i2}$; and $f(h_{i-1}, u_{i1}) = f(h_{i-1}, u_{i2}) = h_i$ for $i = 1, 2, \dots, r$ is generated with

the expected number of $O(r 2^{n/2})$ queries on f . Then $H(h_0, y_1 y_2 \cdots y_r) = H(h_0, z_1 z_2 \cdots z_r)$ for all $y_1, z_1 \in \{u_{11}, u_{12}\}, y_2, z_2 \in \{u_{21}, u_{22}\} \cdots y_r, z_r \in \{u_{r1}, u_{r2}\}$ inducing that $\{u_{11}, u_{21}\} \cdots \{u_{21}, u_{22}\} \cdots \{u_{r1}, u_{r2}\}$ is a 2^r -collision on H .

In (Nandi & Stinson 2007) Nandi and Stinson show that given a compression function $f : \{0, 1\}^n \times \{0, 1\}^m \rightarrow \{0, 1\}^n$, a number $r \in \mathbb{N}_+$, and a word α such that $\text{alph}(\alpha)$ is sufficiently large and $|\alpha|_a \leq 2$ for each $a \in \text{alph}(\alpha)$, there exists a 2^r -collision attack on f_α of complexity $O(r^2 \cdot (\ln r) \cdot (n + \ln(\ln 2r)) \cdot 2^{n/2})$.

In (Hoch & Shamir 2006) Hoch and Shamir generalize the result of Nandi and Stinson by showing the following. Given a compression function $f : \{0, 1\}^n \times \{0, 1\}^m \rightarrow \{0, 1\}^n$, numbers $q, r \in \mathbb{N}_+$, and a word α such that $\text{alph}(\alpha)$ is sufficiently large and $|\alpha|_a \leq q$ for all $a \in \text{alph}(\alpha)$, there exists a 2^r -collision attack on f_α of complexity $O(p(n, r) 2^{\frac{n}{2}})$. Above $p(n, r)$ is a polynomial of n and r . Unfortunately, some of the proofs in (Hoch & Shamir 2006) are only sketches and the presentation contains place to place inaccuracies being thus hard to apprehend. In the following we wish to present the above attack as a well-defined statistical experiment, give rigorous proofs to results verifying that the attack procedure really works, and analyse its complexity in detail.

Multicollisions have been applied in practical attacks usually as a method for generating second preimages for hash values (Klima 2009). Multicollisions have been found for MD4, HAVAL, and Blender (Yu & Wang 2007, Klima 2009). The herding attack proposed by Kelsey & Kohno is also based on multicollisions (Kelsey & Kohno 2006). Moreover, attacks against dithered hash functions and hash functions using linear XOR and additive checksums have been found using multicollisions as a basis (Andreeva et al. 2008, Gauravaram & Kelsey 2008).

Thus, theoretical advances in finding multicollisions have had an impact to practical hash functions. However, there are no practical implementations of the generalized iterated hash functions and thus theoretical advances in this field are not of too great practical impact, but can help in devising more secure hash functions in the future.

3.3 Nested Multicollision Attack Schema (NMCAS)

Below we describe a general (and at this stage still informal) attack procedure that, given $H_{\hat{\alpha}, f}$, $h_0 \in \{0, 1\}^n$, and $r \in \mathbb{N}_+$ creates a 2^r -collision set on in any generalized iterated hash function $H_{\hat{\alpha}, f}$ with initial value h_0 . Throughout the paper we assume that

1. the compression function f is available to our attack procedure: when given a query, f provides the procedure with the respective response; and
2. the sequence $\hat{\alpha} = (\alpha_1, \alpha_2, \dots)$ in $H_{\hat{\alpha}, f}$ is *effectively encoded*, i.e., each word α_i in $\hat{\alpha}$ is directly (that is to say, computationally tractably) at hand as an input to our attack schema.

Procedure Schema NMCAS

Input: A generalized iterated hash function $H_{\hat{\alpha}, f}$, initial value $h_0 \in \{0, 1\}^n$, positive integer r .

Output: A 2^r -collision set on $H_{\hat{\alpha}, f}$.

Step 1: Choose (a large) $l \in \mathbb{N}_+$. Consider the l th element α_l of the sequence $\hat{\alpha}$. Let $\alpha_l = i_1 i_2 \cdots i_s$, where $s \in \mathbb{N}_+$ and $i_j \in \mathbb{N}_l$ for $j = 1, 2, \dots, s$.

Step 2: Fix a (large) set of *active indices* $\text{Act} \subseteq \mathbb{N}_l = \{1, 2, \dots, l\}$.

Step 3: Factorize the word α_l into nonempty strings appropriately, i.e., find $p \in \{1, 2, \dots, s\}$ and $\beta_i \in \mathbb{N}_l^+$ such that $\alpha = \beta_1\beta_2 \dots \beta_p$.

Step 4: Based upon the active indices, create a large multicollision set on f_{β_1} . More exactly, find message block sets M_1, M_2, \dots, M_l satisfying the following properties.

- (i) If $i \in \mathbb{N}_l \setminus Act$, then the set M_i consists of one (constant) message $m_{i1} = iv$.
- (ii) If $i \in Act$, then the set M_i consists of two different messages m_{i1} and m_{i2} .
- (iii) The set $M = M_1M_2 \dots M_l = \{u_1u_2 \dots u_l \mid u_i \in M_i \text{ for } i = 1, 2, \dots, l\}$ is a $2^{|Act|}$ -collision set on f_{β_1} with initial value h_0 .

Step 5: Based on the set $C_1 = M$, find message sets C_2, C_3, \dots, C_p such that

- (iv) $C_p \subseteq C_{p-1} \subseteq \dots \subseteq C_1 = M$.
- (v) For each $j \in \{1, 2, \dots, p\}$ the set C_j is a (large) multicollision set on $f_{\beta_1\beta_2 \dots \beta_j}$ with initial value h_0 .
- (vi) $|C_p| = 2^r$.

Step 6: Output C_p .

It should be clear that if the above procedure is successfully carried out, then

$$H_{\hat{\alpha},f}(h_0, m) = H_{\hat{\alpha},f}(h_0, m')$$

for all $m, m' \in C_p$. It should be noted that \mathcal{NMCAS} can be applied trivially to produce a 2^r -collision set with initial value h_0 for any generalized iterated hash function $H_{\hat{\alpha},f}$. Namely, choosing $l > n + r$ we know that among the messages of length l , a 2^r -collision set exists. Letting then $Act = \mathbb{N}_l$ and $p = 1$, we can, by going in the worst case through all the 2^{n+r} possible message values certainly find the desired multicollision. The complexity of the attack, i.e., the expected number of queries on f is then $((2^r)!)^{\frac{1}{2^r}} 2^{\frac{n(2^r-1)}{2^r}}$ messages (Suzuki et al. 2008).

Given $H_{\hat{\alpha},f}$ and r , does there exist a tractable 2^r -collision attack on $H_{\hat{\alpha},f}$? The problem in its full generality, i.e., with no restrictions on $H_{\hat{\alpha},f}$ seems to be extremely difficult and is certainly still open. We conjecture that the answer is no. Hoch and Shamir showed in (Hoch & Shamir 2006) that if the number of occurrences of different symbols in the words of the sequence $\hat{\alpha}$ is limited by a constant, then a 2^r -collision set on $H_{\hat{\alpha},f}$ with any initial value can be constructed so that the expected number of queries on f is $O(2^{\frac{n}{2}})$.

Call the sequence $\hat{\alpha} = (\alpha_1, \alpha_2 \dots)$ q -bounded, $q \in \mathbb{N}_+$, if $|\alpha_j|_i \leq q$ for each $j \in \mathbb{N}_+$ and $i \in \mathbb{N}_j$. Let $H_{\hat{\alpha},f}$ be a generalized iterated hash function. Given $q \in \mathbb{N}_+$, call the sequence $\hat{\alpha} = (\alpha_1, \alpha_1, \dots)$ (and the respective $H_{\hat{\alpha},f}$ as well) q -bounded if $|\alpha_l|_a \leq q$ for each $l \in \mathbb{N}_+$ and $a \in \mathbb{N}_l$.

Suppose now that $q \in \mathbb{N}_+$ and in $H_{\hat{\alpha},f}$ the sequence $\hat{\alpha}$ is q -bounded. In the following we shall show that the procedure \mathcal{NMCAS} with input $H_{\hat{\alpha},f}$, h_0 , r , can be realized so that a 2^r -collision set in $H_{\hat{\alpha},f}$ with initial value h_0 is created (with probability one) and the expected number of queries on the compression function f is $O(2^{\frac{n}{2}})$.

The reason to the successful construction is the fact that since $\hat{\alpha}$ is q -bounded, unavoidable regularities start to appear in the word α_l of $\hat{\alpha}$ when l is increased. More accurately, choosing l big enough (still so that l depends only polynomially on n), arbitrarily large sets $A \subseteq alph(\alpha_l)$ can be found such that

- (P1) $\alpha_l = \beta_1\beta_2 \dots \beta_p$, where $p \in \mathbb{N}_l$, β_i is a word such that $A \subseteq \beta_i$ and the elements of A

are independent with respect to \prec_{β_i} for $i = 1, 2, \dots, p$; and

- (P2) for any $i \in \{1, 2, \dots, p-1\}$, if $\pi_B(\beta_i) = z_1z_2 \dots z_{n^{p-i}k}$ is a factorization of $\pi_B(\beta_i)$ such that $|alph(z_j)| = n^{i-1}$ for $j = 1, 2, \dots, n^{p-i}k$ and $\pi_B(\alpha_{i+1}) = u_1u_2 \dots u_{n^{p-i-1}k}$ is a factorization of $\pi_B(\alpha_{i+1})$ such that $|alph(u_j)| = n^i$ for $j = 1, 2, \dots, n^{p-i-1}k$, then for each $j_1 \in \{1, 2, \dots, n^{p-i}k\}$, there exists $j_2 \in \{1, 2, \dots, n^{p-i-1}k\}$ such that $alph(z_{j_1}) \subseteq alph(u_{j_2})$.

The property (P1) allows us to construct a $2^{|Act|}$ -collision on f_{β_1} with any initial value h_0 so that the expected number of queries on f is $2.5|\beta_1|2^{\frac{n}{2}}$, i.e., steps 1 to 4 in \mathcal{NMCAS} can be carried out tractably. The property (P2) guarantees that based on the multicollision set guaranteed by (P1), we can roll the attack on and create the multicollision set C_i for $f_{\beta_1\beta_2 \dots \beta_i}$ so that the expected number of queries on f is $|\beta_1\beta_2 \dots \beta_i|2^{\frac{n}{2}}$ for $i = 2, 3, \dots, p$ and the cardinality of C_p is 2^r .

Thus steps 5 and 6 in \mathcal{NMCAS} do not consume too much resources.

We give the necessary combinatorial results for properties (P1) and (P2) in the next section. The construction of the actual attack is postponed to Section 5. For rigorous proofs we refer to the full version of the paper to be published later.

Remark. In many combinatorial problems of words arbitrarily long words over a fixed (finite) alphabet are considered; unavoidable regularities appear as well as in our case and famous results of classical combinatorics like Ramsey's Theorem and Shirshov's Theorem may be applied (for details, see for instance the book of deLuca and Varricchio (DeLuca & Varricchio 1999)).

4 Basic combinatorial results

Let α be a (nonempty) word and A any alphabet. We wish to study how the occurrences in α of any symbol $a \in A$ are positioned in relation to occurrences in α of other symbols of A . Define $(\alpha)_A = \epsilon$ if $\pi_A(\alpha) = \epsilon$ and $(\alpha)_A = a_1a_2 \dots a_s$ if $\pi_A(\alpha) \in a_1^+a_2^+ \dots a_s^+$, where $s \in \mathbb{N}_+$, $a_1, a_2, \dots, a_s \in A$, and $a_i \neq a_{i+1}$ for $i = 1, 2, \dots, s-1$. It is clear that the word α_A exists and is unique.

Remark. Let α be a nonempty word and $A \subseteq alph(\alpha)$ nonempty. Then the following conditions are equivalent.

- (a) the elements of A are independent with respect to \prec_α ;
- (b) there exists an order a_1, a_2, \dots, a_d of all elements of A such that the word $\pi_A(\alpha)$ is in $a_1^+a_2^+ \dots a_d^+$; and
- (c) the word α_A is a permutation of A .

Suppose that $\hat{\alpha} = (\alpha_1, \alpha_2, \dots)$ is q -bounded, $q \in \mathbb{N}_+$, i.e., for each $j \in \mathbb{N}_+$ and $i \in \mathbb{N}_j$, the inequality $|\alpha_j|_i \leq q$ holds. Our first task is to show that the property (P1) (see the previous page) holds.

The result presented in Lemma 1 in (Hoch & Shamir 2006) (Theorem 2 in this paper) can be proved directly without applying graph theory and Hall's matching theorem. Let $A = (a_{ij})_{n \times n}$ be a $n \times n$ -dimensional binary matrix ($n \in \mathbb{N}_+$). We say that A possesses a *pass* if there exist $r, s \in \{1, 2, \dots, n\}$, $r > s$, such that A contains a $r \times (n-s)$ -dimensional submatrix with zero elements only. The pair (r, s) is the *size* of the pass. We show the following

Lemma 1. *The matrix $A = (a_{ij})_{n \times n}$ does not possess a pass if and only if there exists a bijection $\sigma : \{1, 2, \dots, n\} \rightarrow \{1, 2, \dots, n\}$ such that $a_{i\sigma(i)} = 1$ for $i = 1, 2, \dots, n$.*

The previous lemma straightforwardly implies a result concerning equal size partitions of a finite set into equal size sets.

Theorem 2. *Let $k \in \mathbb{N}_+$ and A be a finite nonempty set such that k divides $|A|$. Let furthermore $\{B_i\}_{i=1}^k$ and $\{C_j\}_{j=1}^k$ be partitions of A such that $|B_i| = |C_j|$ for $i, j = 1, 2, \dots, k$. Then for each $x \in \mathbb{N}_+$ such that $|A| \geq k^3 \cdot x$, there exists a bijection $\sigma : \{1, 2, \dots, k\} \rightarrow \{1, 2, \dots, k\}$ for which $|A_i \cap B_{\sigma(i)}| \geq x$ for $i = 1, 2, \dots, k$.*

Remark. The result of the previous theorem is the best possible in the sense that the power 3 of k cannot be reduced to 2. Consider the following example. Let A be a set consisting of $k^2 \cdot x$ elements where $k, x \in \mathbb{N}_+$, $x \geq k^2$. Suppose $r \in \{1, 2, \dots, k-2\}$ and let $\{A_i\}_{i=1}^k$ and $\{B_i\}_{i=1}^k$ be two partitions of A such that $|A_i \cap B_1| = x + k - r$ for $i = 1, 2, \dots, r+1$; $|A_i \cap B_j| = x$ for $i = 1, 2, \dots, r+1$, $j = 2, 3, \dots, r$; $|A_i \cap B_j| = x - 1$ for $i = 1, 2, \dots, r+1$, $j = r+1, r+2, \dots, k$; $|A_{r+2} \cap B_1| = x - (r+1)(k-r)$; $|A_{r+2} \cap B_j| = x$ for $j = 2, 3, \dots, r$; $|A_{r+2} \cap B_j| = x + r + 1$ for $j = r+1, r+2, \dots, k$; and $|A_i \cap B_j| = x$ for $i = r+3, r+4, \dots, k$, $j = 1, 2, \dots, k$. Then $|A| = k^2 \cdot x$ and $|A_i| = |B_i| = k \cdot x$ for $i = 1, 2, \dots, k$. Clearly there does not exist a permutation σ of $\{1, 2, \dots, k\}$ such that $|A_i \cap B_{\sigma(i)}| \geq x$ for $i = 1, 2, \dots, k$.

The following lemma is a nice application of Dilworth's famous theorem.

Lemma 2. *Let m, n and q be positive integers and α a word such that $\text{alph}(\alpha) \geq m \cdot n$. Then either (i) the maximum chain length of $(\text{alph}(\alpha), \prec_\alpha)$ is at least m ; or (ii) the maximum number of pairwise incomparable elements in $(\text{alph}(\alpha), \prec_\alpha)$ is greater than n .*

Remark. It should be noted that the limits given by the previous lemma are sharp in the sense that for each $m, n \in \mathbb{N}_+$, there exists a word α such that $|\text{alph}(\alpha)| = m \cdot n$, the maximum chain length in $(\text{alph}(\alpha), \prec_\alpha)$ is equal to m and the maximum number of pairwise incomparable elements in the set $(\text{alph}(\alpha), \prec_\alpha)$ is equal to n . Clearly the word $(a_1 b_1 c_1 a_1)^2 (a_2 b_2 c_2)^2 (a_3 b_3 c_3)^2 (a_4 b_4 c_4)^2$ is an example of such an α for $|\alpha| = 12$, $m = 4$, $n = 3$.

By applying Theorem 2, Lemma 2 and the pigeonhole principle in a nontrivial manner, one can prove

Theorem 3. *For all positive integers q and m there exist positive integers r_q and s_q with the following property. Let α be a word such that $|\text{alph}(\alpha)| \geq r_q \cdot m^{s_q}$ and $|\alpha|_a \leq q$ for each $a \in \text{alph}(\alpha)$. Then there exists $A \subseteq \text{alph}(\alpha)$ with $|A| \geq m$ and $p \in \{1, 2, \dots, q\}$ as well as words $\alpha_1, \alpha_2, \dots, \alpha_p$ such that $\alpha = \alpha_1 \alpha_2 \dots \alpha_p$ and for all $i \in \{1, 2, \dots, p\}$, the word $(\alpha_i)_A$ is a permutation of A .*

Remark. The parameters r_q and s_q in Theorem 3 grow very fast with respect to q , the parameter restricting the number of occurrences of any symbol in α . For s_q we have the recurrence relations

$$\begin{cases} s_1 = 1 \\ s_{q+1} = s_q^2 + 1, \text{ if } q \in \mathbb{N}_+ \end{cases}$$

We can roughly estimate that $s_q^2 < s_{q+1} < 2s_q^2$ for each $q \in \mathbb{N}$. It is easily seen that s_q is in $\Omega(2^{2^{q-1}})$ and

in $O(2^{2^{q-1}})$. On the other hand

$$\begin{cases} r_1 = 1 \\ r_{q+1} = (q+1)^{s_q} r_q^{s_q+1}, \text{ if } q \in \mathbb{N}_+ \end{cases}$$

Again, with a rough estimate, $(q+1)^{s_q} r_q^{s_q} < r_{q+1} < (q+1) r_q^{2s_q}$ for all $q \in \mathbb{N}$, $q \geq 2$. Again, with a standard consideration we find that r_q is in $\Omega(2^{2^{2^{q-1}-1}})$ and in $O(2^{2^{2^{q-3}}})$. This, among other things, limits the appliance of the lemma substantially. It means that one can exercise the lemma only to those words in which $\text{alph}(\alpha)$ is very large when compared with q .

The result we achieved in our previous theorem is not yet sufficient for our purposes; inside the permutations $(\alpha_1)_A, (\alpha_2)_A, \dots, (\alpha_k)_A$ of A , the symbols have to be appropriately grouped. We need an application of the following lemma.

Lemma 3. *Let $d_0, d_1, d_2, \dots, d_r$, where $r \in \mathbb{N}_+$ be positive integers such that d_i divides d_{i-1} for $i = 1, 2, \dots, r$, A an alphabet of cardinality $|A| = d_0 d_1^2 d_2^2 \dots d_r^2$, and w_1, w_2, \dots, w_{r+1} permutations of A . There then exists a subset B of A of cardinality $|B| = d_0$ such that the following conditions are satisfied.*

- (1) *For any $i \in \{1, 2, \dots, r\}$, if $\pi_B(w_i) = x_1 x_2 \dots x_{d_i}$ is the factorization of $\pi_B(w_i)$ and $\pi_B(w_{i+1}) = y_1 y_2 \dots y_{d_i}$ is the factorization of $\pi_B(w_{i+1})$ into d_i equal length ($= \frac{d_0}{d_i}$) blocks, then for each $j \in \{1, 2, \dots, d_i\}$, there exists $j' \in \{1, 2, \dots, d_i\}$ such that $\text{alph}(x_j) = \text{alph}(y_{j'})$; and*
- (2) *If $w_r = z_1 z_2 \dots z_{d_r}$ and $w_{r+1} = u_1 u_2 \dots u_{d_r}$ are factorizations of w_r and w_{r+1} , respectively, into d_r equal length ($= d_0 d_1^2 d_2^2 \dots d_{r-1}^2 d_r$) blocks, then the words $\pi_B(w_r) = \pi_B(z_1) \pi_B(z_2) \dots \pi_B(z_{d_r})$ and $\pi_B(w_{r+1}) = \pi_B(u_1) \pi_B(u_2) \dots \pi_B(u_{d_r})$ are factorizations of $\pi_B(w_r)$ and $\pi_B(w_{r+1})$, respectively, into d_r equal length ($= \frac{d_0}{d_r}$) blocks.*

The next theorem is of fundamental importance to our further considerations. It combines the results of Theorem 3 and Lemma 3.

Theorem 4. *Let α be a word and $k \geq 2$ and $q \geq 1$ integers such that*

- (1) $|\text{alph}(\alpha)| \geq r_q n^{(q^2-4q+5)s_q} k^{(2q-3)s_q}$; and
- (2) $|\alpha|_a \leq q$ for each $a \in \text{alph}(\alpha)$

where r_q and s_q are as in Theorem 3. There then exists $B \subseteq \text{alph}(\alpha)$, $p \in \{1, 2, \dots, q\}$ and a factorization $\alpha = \alpha_1 \alpha_2 \dots \alpha_p$ for which

- (3) $|B| = n^{p-1} k$;
- (4) $B \subseteq \text{alph}(\alpha_i)$ and the elements of B are independent with respect to \prec_{α_i} for $i = 1, 2, \dots, p$; and
- (5) *For any $i \in \{1, 2, \dots, p-1\}$, if $\pi_B(\alpha_i) = z_1 z_2 \dots z_{n^{p-i} k}$ is the factorization of $\pi_B(\alpha_i)$ into $n^{p-i} k$ equal length ($= n^{i-1}$) blocks and $\pi_B(\alpha_{i+1}) = u_1 u_2 \dots u_{n^{p-i-1} k}$ the factorization of $\pi_B(\alpha_{i+1})$ into $n^{p-i-1} k$ equal length ($= n^i$) blocks, then for each $j_1 \in \{1, 2, \dots, n^{p-i} k\}$, there exists $j_2 \in \{1, 2, \dots, n^{p-i-1} k\}$ such that $\text{alph}(z_{j_1}) \subseteq \text{alph}(u_{j_2})$.*

5 Construction and analysis of the nested multicollision attack

In this section we shall supplement the steps of the Nested Multicollision Attack Schema so that a detailed description and analysis of a probabilistic multicollision attack procedure comes true.

5.1 The attack as a statistical experiment

Suppose that $f : \{0, 1\}^n \times \{0, 1\}^m \rightarrow \{0, 1\}^n$ is a compression function (not necessarily a FIL-RO) and $l \in \mathbb{N}_+$. Let iv be a fixed constant message block in $\{0, 1\}^m$. Assume furthermore that we have fixed a set $Act \subseteq \mathbb{N}_l$ of so called *active indices*. Let $\tau \in \mathbb{N}_l^+$ be a word such that $alph(\tau)$ contains exactly one element, say t , which is an active index.

A *basic birthday attack on f_τ with active index t and initial value h* , denoted by $BBA(f_\tau, t, h)$ is understood to be a statistical (probabilistic) experiment carried out as follows.

- (1) Generate a set $R \subseteq \{0, 1\}^m$ of $2^{\frac{n}{2}}$ random message blocks;
- (2) Let

$$S = \{u_1 u_2 \cdots u_l \mid u_t \in R \text{ and } \forall i \in \mathbb{N}_l \setminus \{t\} : u_i = iv\};$$

- (3) For each $u \in S$, compute the value $f_\tau(h, u)$ to find message blocks $x, y \in R$, $x \neq y$, and the respective collision value h' such that

$$\begin{aligned} f_\tau(h, (iv)^{t-1} x (iv)^{l-t}) &= \\ f_\tau(h, (iv)^{t-1} y (iv)^{l-t}) &= h'. \end{aligned}$$

The probability that $BBA(f_\tau, t, h)$ yields a collision is (approximately) equal to 0.4 (for details, see for instance (Nandi & Stinson 2007, Menezes et al. 1996)). In an (*extended*) *birthday attack* on f_τ with active index t and initial value h , (abbreviated $EBA(f_\tau, t, h)$) one or more basic birthday attacks are carried out one after another so long that a collision is found. Thus in an extended birthday attack a collision is always found with probability equal to one. Assuming that the collision probability in a basic birthday attack is exactly 0.4, the expected number of BBA 's in an EBA is obviously 2.5. Thus the expected number of queries on f in $EBA(f_\tau, t, h)$ is equal to $2.5 |\tau| 2^{\frac{n}{2}}$.

Let now α be a word over the alphabet \mathbb{N}_l and Act be the set of $r \in \mathbb{N}_+$ active indices a_1, a_2, \dots, a_r such that $a_1 \prec_\alpha a_2 \prec_\alpha \cdots \prec_\alpha a_r$. Suppose furthermore that $\alpha = \alpha_1 \alpha_2 \cdots \alpha_r$ is a factorization of α such that for each $i \in \{1, 2, \dots, r\}$, all occurrences of the symbol a_i in α lie in α_i . In our construction (see Lemma 4), a sequence

$$\begin{aligned} EBA(f_{\alpha_1}, h_0, a_1), EBA(f_{\alpha_2}, h_1, a_2), \dots, \\ EBA(f_{\alpha_r}, h_{r-1}, a_r) \end{aligned}$$

of extended birthday attacks is executed. Above h_0 is the initial value and for each $i \in \{1, 2, \dots, r\}$, during the execution of $EBA(f_{\alpha_i}, h_{i-1}, a_i)$, values $h_i \in \{0, 1\}^n$ and distinct message blocks $x_{a_i}, y_{a_i} \in \{0, 1\}^m$ are found such that

$$\begin{aligned} h_i &= f_{\alpha_i}(h_{i-1}, (iv)^{a_i-1} x_{a_i} (iv)^{l-a_i}) = \\ &= f_{\alpha_i}(h_{i-1}, (iv)^{a_i-1} y_{a_i} (iv)^{l-a_i}) \end{aligned}$$

The collision value h_i of $EBA(f_{\alpha_i}, h_{i-1}, a_i)$ serves as the initial value to the attack $EBA(f_{\alpha_{i+1}}, h_i, a_{i+1})$ for $i = 1, 2, \dots, r-1$. We may assume that the EBA 's above are statistically independent, so the expected

number of BBA 's in the sequence is $2.5r$. We may also deduce that the expected number of queries on the total sequence is equal to $2.5 |\alpha_1 \alpha_2 \cdots \alpha_r| 2^{\frac{n}{2}}$. Obviously the set

$$M = \{u_1 u_2 \cdots u_l \mid \forall i \in \{1, 2, \dots, r\} : u_{a_i} \in \{x_{a_i}, y_{a_i}\} \text{ and } \forall i \in \mathbb{N}_l \setminus Act : u_i = iv\}$$

is a 2^r -multicollision on f_α with initial value h_0 . If we above chose $\alpha_i = a_i$ for $i = 1, 2, \dots, r$, we can interpret Joux's 2^r -multicollision attack to be a special case of our construction: certainly the complexity of this attack is $2.5r 2^{\frac{n}{2}}$.

The time is now ripe to augment the three first steps in the procedure \mathcal{NMCAS} . We call the expanded plan of action *Nested Multicollision Attack* (\mathcal{NMCA}).

Procedure \mathcal{NMCA}

Input: A q -bounded generalized iterated hash function $H_{\hat{\alpha}, f}$, initial value $h_0 \in \{0, 1\}^n$, positive integer r .

Output: A 2^r -collision set on $H_{\hat{\alpha}, f}$.

Step 1: Let $l = r_q n^{(q^2-4q+5)s_q} k^{(2q-3)s_q}$. Let $\alpha = \alpha_l$ where α_l is the l th element of the sequence $\hat{\alpha}$. Write α in the form $\alpha = i_1 i_2 \cdots i_s$, where $s \in \mathbb{N}_+$ and $i_j \in \mathbb{N}_l$ for $j = 1, 2, \dots, s$.

Step 2: Let $Act = B$, where $B \subseteq \mathbb{N}_l = \{1, 2, \dots, l\}$ is as in Theorem 4, be the set of active indices.

Step 3: Let p be as in Theorem 4 and $\alpha = \beta_1 \beta_2 \cdots \beta_p$ the factorization of α such that the words $\beta_1, \beta_2, \dots, \beta_p$ have the same properties as the words $\alpha_1, \alpha_2, \dots, \alpha_p$, respectively, in Theorem 4.

5.2 The two phases of the attack

Lemma 4. Let α be a word over the alphabet \mathbb{N}_l , r a positive integer and a_1, a_2, \dots, a_r in $alph(\alpha)$ symbols such that $a_1 \prec_\alpha a_2 \prec_\alpha \cdots \prec_\alpha a_r$. Let furthermore $\alpha = \alpha_1 \alpha_2 \cdots \alpha_r$ be a factorization of α such that for each $i \in \{1, 2, \dots, r\}$, all occurrences of the symbol a_i in α lie in α_i . Given an initial value $h_0 \in \{0, 1\}^n$, we can, with probability equal to one, find message block sets $M_1, M_2, \dots, M_l \subseteq \{0, 1\}^m$ as well as values $h_1, h_2, \dots, h_r \in \{0, 1\}^n$ such that

- (1) $M_b = \{iv\}$ for each $b \in \mathbb{N}_l \setminus A$, where $A = \{a_1, a_2, \dots, a_r\}$;
- (2) $M_{a_i} = \{u_i, u'_i\}$, where $u_i \neq u'_i$ for each $i \in \{1, 2, \dots, r\}$;
- (3) for each $i \in \{1, 2, \dots, r\}$ the set $M = M_1 \cdot M_2 \cdots M_l$ is a 2-collision set on f_{α_i} with initial value h_{i-1} and a 2^i -collision set on $f_{\alpha_1 \alpha_2 \cdots \alpha_i}$ such that $\forall u, u' \in M$:

$$\begin{aligned} h_i &= f_{\alpha_i}(h_{i-1}, u) = f_{\alpha_i}(h_{i-1}, u'); \text{ and} \\ &= f_{\alpha_1 \alpha_2 \cdots \alpha_i}(h_0, u) = f_{\alpha_1 \alpha_2 \cdots \alpha_i}(h_0, u'). \end{aligned}$$

Moreover, the expected number of queries on f needed to carry out the task is $2.5 |\alpha| 2^{\frac{n}{2}}$.

We can now top up the fourth step of \mathcal{NMCAS} .

Step 4 of \mathcal{NMCA} : Let M_1, M_2, \dots, M_l be as in Lemma 4.

Lemma 5. Let α be a word over the alphabet \mathbb{N}_l , d and r positive integers, $A \subseteq alph(\alpha)$ a set of cardinality $|A| = dnr$, and $\alpha = \beta_1 \beta_2 \cdots \beta_{nr} \gamma_1 \gamma_2 \cdots \gamma_r$ a factorization of α with the following properties.

- (1) $A \subseteq alph(\beta) \cap alph(\gamma)$ where $\beta = \beta_1 \beta_2 \cdots \beta_{nr}$ and $\gamma = \gamma_1 \gamma_2 \cdots \gamma_r$;

- (2) $|\text{alph}(\beta_i) \cap A| = d$ for $i = 1, 2, \dots, nr$, and $|\text{alph}(\gamma_j) \cap A| = nd$ for $j = 1, 2, \dots, r$; and
- (3) for each $i \in \{1, 2, \dots, nr\}$ there exists $j \in \{1, 2, \dots, r\}$ such that $\text{alph}(\beta_i) \subseteq \text{alph}(\gamma_j)$.

Let furthermore $u_1, u'_1, u_2, u'_2, \dots, u_{nr}, u'_{nr} \in \{0, 1\}^{ml}$ be messages and h_0, h_1, \dots, h_{nr} in $\{0, 1\}^n$ be values such that for each $i \in \{1, 2, \dots, nr\}$:

- (4) $\forall b \in \mathbb{N}_l \setminus A : u_i(b) = u'_i(b) = iv$; and
- (5) $u_i(\beta_i) \neq u'_i(\beta_i)$ and $h_i = f_{\beta_i}(h_{i-1}, u_i) = f_{\beta_i}(h_{i-1}, u'_i)$.

The set S of all messages $u \in \{0, 1\}^{ml}$ such that for each $b \in \mathbb{N}_l \setminus A : u(b) = iv$ and for each $i \in \{1, 2, \dots, nr\} : u(\beta_i) \in \{u_i(\beta_i), u'_i(\beta_i)\}$ is then well-defined and satisfies for each $i \in \{1, 2, \dots, nr\}$ and $u \in S$ the equality $h_i = f_{\beta_i}(h_{i-1}, u)$. Moreover we can, with probability equal to one, find messages $v_1, v'_1, v_2, v'_2, \dots, v_r, v'_r$ in S and values $h'_0, h'_1, \dots, h'_r, h'_0 = h_{nr}$, such that for each $j \in \{1, 2, \dots, r\}$:

- (6) $v_i(\gamma_j) \neq v'_j(\gamma_j)$ and $h'_j = f_{\gamma_j}(h'_{j-1}, v_j) = f_{\gamma_j}(h'_{j-1}, v'_j)$.

The expected number of queries on f needed to carry out the task is $2.5|\gamma|2^{\frac{n}{2}}$. Finally, the set T of all messages $v \in \{0, 1\}^{ml}$ such that for each $b \in \mathbb{N}_l \setminus A : v(b) = iv$ and for each $j \in \{1, 2, \dots, r\} : v(\gamma_j) \in \{v_j(\gamma_j), v'_j(\gamma_j)\}$ is then a well-defined subset of S and forms a nontrivial 2^r -collision set on f_α with initial value h_0 .

The following theorem combines the results of the two previous lemmata; we verify that Step 5 in \mathcal{NMCAS} can be carried out in a tractable fashion without consuming too much resources.

Theorem 5. Let α be a word over the alphabet \mathbb{N}_l , r and p positive integers, A a subset of the alphabet $\text{alph}(\alpha)$ of cardinality $|A| = n^{p-1}r$, and $\alpha = \alpha_1\alpha_2 \dots \alpha_p$ a factorization of α such that for each $i \in \{1, 2, \dots, p\}$, the elements of A form a chain in the partially ordered set $(\text{alph}(\alpha_i), \prec_{\alpha_i})$ (i.e., the elements of A are independent with respect to \prec_{α_i}). Assume furthermore that for each $i \in \{1, 2, \dots, p\}$, there exists a factorization $\alpha_i = \alpha_{i1}\alpha_{i2} \dots \alpha_{i, n^{p-i}r}$ of the word α_i such that the following conditions are satisfied.

- (1) $|\text{alph}(\alpha_{ij}) \cap A| = n^{i-1}$ for each $i \in \{1, 2, \dots, p\}$ and $j \in \{1, 2, \dots, n^{p-i}r\}$; and
- (2) for each $i \in \{1, 2, \dots, p\}$ and $j \in \{1, 2, \dots, n^{p-i}r\}$ there exists $k \in \{1, 2, \dots, n^{p-i-1}r\}$ such that $\text{alph}(\alpha_{ij}) \cap A$ is a subset of $\text{alph}(\alpha_{i+1,k}) \cap A$

Then, given an initial value $h_0 \in \{0, 1\}^n$ we can, with probability equal to one, find a nontrivial 2^r -collision set on f_α . Moreover, the expected number of queries on f_α needed to carry out the task is $2.5|\alpha|2^{\frac{n}{2}}$.

The fifth step of step of \mathcal{NMCAS} can be completed:

Step 5 of \mathcal{NMCAS} : Let B_1, B_2, \dots, B_p be as in Theorem 4.

Let us recapitulate our results.

Theorem 6. Let m, n and q be positive integers such that $m > n$ and $q \geq 2$, f a compression function of block size m and length n , and $\hat{\alpha} = (\alpha_1, \alpha_2, \dots)$ a q -bounded sequence of words such that $\text{alph}(\alpha_i) = \mathbb{N}_l$ for each $i \in \mathbb{N}_+$. Assume furthermore that $\hat{\alpha}$ is effectively encoded. There then exists a probabilistic algorithm which, given any $r \in \mathbb{N}_+$ constructs a 2^r -collision set on the generalized iterated hash function $H_{\hat{\alpha}, f}$ so that the expected number of queries on f is $2.5r_q n^{(q^2-4q+5)s_q r^{(2q-3)s_q} 2^{\frac{n}{2}}}$, where the parameters r_q and s_q are defined recursively by $r_1 = s_1 = 1$, $r_{i+1} = (i+1)^{s_i} r_i^{s_i+1}$ and $s_{i+1} = s_i^2 + 1$ for $i \in \mathbb{N}_+$.

We wish to remind that in (Hoch & Shamir 2006) an informal proof of the previous theorem was given.

5.3 The case $q = 2$ and some complexity considerations

Suppose now that in the input of the procedure \mathcal{NMCAS} procedure the generalized iterated hash function $H_{\hat{\alpha}, f}$ is such that the sequence $\hat{\alpha} = (\alpha_1, \alpha_2, \dots)$ is 2-bounded. Now assume that $q = 2$. Then, by Theorem 3, the equalities $r_2 = 2^{s_1} r_1^{s_1+1} = 2$ and $s_2 = s_1^2 + 1 = 2$ hold. By Theorem 6, when creating a 2^r -collision ($r \in \mathbb{N}_+$) on $H_{\hat{\alpha}, f}$, the expected number of queries on f is $2.5r_2 n^{(2^2-4\cdot 2+5)s_2 r^{(2\cdot 2-3)s_2} 2^{\frac{n}{2}}} = 5n^2 k^2 2^{\frac{n}{2}}$. In (Nandi & Stinson 2007) with rigorous considerations a somewhat smaller average complexity $O(r^2 \cdot (\ln r) \cdot (n + \ln(\ln 2r)) \cdot 2^{n/2})$ was attained.

Note that in \mathcal{NMCAS} it is possible to take also q as an input parameter. Then the procedure is of course extremely unefficient: due to the recurrence relations $r_1 = s_1 = 1$, and $r_{i+1} = (i+1)^{s_i} r_i^{s_i+1}$, $s_{i+1} = s_i^2 + 1$ for $i \in \mathbb{N}_+$, we have a procedure that is at least triple exponential with respect to q . A natural question arises whether or not in Theorem 3, the length of the word α could be chosen to be considerably smaller. Our opinion is that then a significantly different proof technique is needed. If Lemma 2 and Dilworth's Theorem (and thus the relation between independent elements and incomparable elements in $\text{alph}(\alpha)$) is applied, it is difficult to imagine that a 2^r -collision on $H_{\hat{\alpha}, f}$ could be constructed so that the expected number of queries on f is less than exponential with respect to q .

When q is greater than 2, it is fairly easy to see from Theorems 4 and 6 and Remark 4 that the length of the messages necessary for carrying out our attack becomes impossible for any practical implementation.

5.4 Computational complexity and security issues

It seems that to implement a generalized iterated hash function, a relatively strong computing device (in automata-theoretic sense) is needed. In fact, a two-way deterministic pushdown transducer technique seems to be indispensable regardless of the way the respective compression function is realized as a computer program. This raises the question of efficiency; a two-way deterministic pushdown transducer is a much more complicated machine (and thus much more resource consuming to implement and use) than a finite state transducer which is needed to realize a traditional iterated hash function. Two-way deterministic pushdown automata accept a subfamily of context sensitive languages that contains non-context-free languages whereas finite automata accept only regular languages.

If a generalized iterated hash function is used, the sender has to construct the whole message before he can start to hash it. Similarly, the receiver has to have the complete message available before the sent

hash value can be verified to be correct. Suppose that we wish to avoid this restriction and start to hash the message before it is completed. Then the message blocks occurring at the end of the message are not available when we start hashing; this causes extra restrictions on the sequence $\hat{\alpha}$ and possibly implies that multicollisions are easily found.

We also need an efficient encoding for the sequence $\hat{\alpha} = (\alpha_1, \alpha_2, \dots)$. If $\hat{\alpha}$ is complicated, i.e., $H_{\hat{\alpha}, f}$ is secure, then picking an element α_l from $\hat{\alpha}$ may be resource consuming. On the other hand, if $\hat{\alpha}$ is simple, then efficient multicollision attacks become possible.

6 Conclusion

In this paper we have demonstrated how the analysis of multicollisions in iterated hash functions can be done with the use of word combinatorics. We have also given some new results and settled some inaccuracies in the previous results concerning multicollisions in generalised iterated hash functions. We have also brought these results into a unified and well established theoretical framework, which should make further investigation of the theory of iterated hash functions easier.

The next step in the research could be to investigate the possibilities of generating words, which have desirable properties in the context of multicollisions. We could also categorise words and whole languages with respect to their performance in the iterative structure. Also the even more general types of iterated hash functions presented in (Nandi & Stinson 2007) and (Hoch & Shamir 2006) could be brought into this framework and further analysed.

References

- Andreeva, E., Bouillaguet, C., Fouque, P.-A., Hoch, J. J., Kelsey, J., Shamir, A. & Zimmer, S. (2008), Second preimage attacks on dithered hash functions, in N. P. Smart, ed., 'EUROCRYPT', Vol. 4965 of *Lecture Notes in Computer Science*, Springer, pp. 270–288.
- Damgård, I. B. (1989), A design principle for hash functions, in G. Brassard, ed., 'Advances in Cryptology—CRYPTO '89', Vol. 435 of *Lecture Notes in Computer Science*, Springer-Verlag, 1990, pp. 416–427.
- DeLuca, A. & Varrichio, S. (1999), *Finiteness and Regularity in Semigroups and Formal Languages*, Springer Verlag.
- Dilworth, R. (1950), 'A decomposition theorem for partially ordered sets', *The Annals of Mathematics* **51**, 161–166.
- Dobbertin, H. (1998), 'Cryptanalysis of MD4', *Journal of Cryptology: the journal of the International Association for Cryptologic Research* **11**(4), 253–271.
- Gauravaram, P. & Kelsey, J. (2008), Linear-XOR and additive checksums don't protect damgård-merkle hashes from generic attacks, in T. Malkin, ed., 'CT-RSA', Vol. 4964 of *Lecture Notes in Computer Science*, Springer, pp. 36–51.
- Hoch, J. J. & Shamir, A. (2006), Breaking the ICE - finding multicollisions in iterated concatenated and expanded (ICE) hash functions, in M. J. B. Robshaw, ed., 'FSE', Vol. 4047 of *Lecture Notes in Computer Science*, Springer, pp. 179–194.
- Joux, A. (2004), Multicollisions in iterated hash functions. application to cascaded constructions, in M. K. Franklin, ed., 'CRYPTO', Vol. 3152 of *Lecture Notes in Computer Science*, Springer, pp. 306–316.
- Kelsey, J. & Kohno, T. (2006), Herding hash functions and the nostradamus attack, in S. Vaudenay, ed., 'EUROCRYPT', Vol. 4004 of *Lecture Notes in Computer Science*, Springer, pp. 183–200.
- Klima, V. (2005), 'Finding MD5 collisions on a notebook PC using multi-message modifications', <http://eprint.iacr.org/2005/102.pdf>.
- Klima, V. (2009), 'Huge multicollisions and multi-preimages of hash functions blender-n', *Cryptology ePrint Archive*, Report 2009/006. <http://eprint.iacr.org/>.
- Menezes, A. J., van Oorschot, P. C. & Vanston, S. A., eds (1996), *Handbook of Applied Cryptography*, CRC Press.
- Merkle, R. C. (1990), A certified digital signature, in G. Brassard, ed., 'Advances in Cryptology – CRYPTO '89', Vol. 435 of *Lecture Notes in Computer Science*, International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, Santa Barbara, CA, USA, pp. 218–238.
- Nandi, M. & Stinson, D. R. (2007), 'Multicollision attacks on some generalized sequential hash functions', *IEEE Transactions on Information Theory* **53**(2), 759–767.
- Stevens, M. (2006), 'Fast collision attack on MD5', <http://eprint.iacr.org/2006/104.pdf>.
- Suzuki, K., Tonien, D., Kurosawa, K. & Toyota, K. (2008), 'Birthday paradox for multi-collisions', *IEEE Transactions* **91-A**(1), 39–45.
- Wang, X., Yin, Y. L. & Yu, H. (2005), Finding collisions in the full SHA-1, in V. Shoup, ed., 'CRYPTO', Vol. 3621 of *Lecture Notes in Computer Science*, Springer, pp. 17–36.
- Wang, X. & Yu, H. (2005), How to break MD5 and other hash functions, in R. Cramer, ed., 'EUROCRYPT', Vol. 3494 of *Lecture Notes in Computer Science*, Springer, pp. 19–35.
- Yu, H. & Wang, X. (2007), Multi-collision attack on the compression functions of MD4 and 3-pass HAVAL, in K.-H. Nam & G. Rhee, eds, 'ICISC', Vol. 4817 of *Lecture Notes in Computer Science*, Springer, pp. 206–226.

Author Index

Alfawaz, Salahuddin, 51

Boyd, Colin, iii
Burdon, Mark, 3

Carter, Gary, 73
Chang, Sheueling, 61
Chrobok, Natascha, 25

Dawson, Ed, 3, 35, 73

Fidge, Colin, 15

González Nieto, Juanma, 81

Halunen, Kimmo, 93

Kortelainen, Juha, 93
Kortelainen, Tuomas, 93

Lang, Simon, 43
Lippold, Georg, 81

Mohannak, Kavoos, 51

Nelson, Karen, 51

O'Keefe, Richard, 25

Pieprzyk, Josef, 7

Reid, Jason, 3, 35

Salim, Farzad, 3, 35
Stebila, Douglas, 61
Susilo, Willy, iii

Tang, Maolin, 15
Trotman, Andrew, 25

Udupi, Poornaprajna, 61

Wang, Huaxiong, 7
Wang, Peishun, 7
Williams, Neville, 43
Wong, Kenneth Koon-Ho, 73

Recent Volumes in the CRPIT Series

ISSN 1445-1336

Listed below are some of the latest volumes published in the ACS Series *Conferences in Research and Practice in Information Technology*. The full text of most papers (in either PDF or Postscript format) is available at the series website <http://crpit.com>.

- Volume 84 - Artificial Intelligence and Data Mining 2007**
 Edited by Kok-Leong Ong, Deakin University, Australia, Wenyuan Li, University of Texas at Dallas, USA and Junbin Gao, Charles Sturt University, Australia. December, 2007. 978-1-920682-65-1. Contains the proceedings of the 2nd International Workshop on Integrating AI and Data Mining (AIDM 2007), Gold Coast, Australia. December 2007.
- Volume 85 - Advances in Ontologies 2007**
 Edited by Thomas Meyer, Meraka Institute, South Africa and Abhaya Nayak, Macquarie University, Australia. December, 2007. 978-1-920682-66-8. Contains the proceedings of the 3rd Australasian Ontology Workshop (AOW 2007), Gold Coast, Queensland, Australia.
- Volume 86 - Safety Critical Systems and Software 2007**
 Edited by Tony Cant, Defence Science and Technology Organisation, Australia. December, 2007. 978-1-920682-67-5. Contains the proceedings of the 12th Australian Conference on Safety Critical Systems and Software, August 2007, Adelaide, Australia.
- Volume 87 - Data Mining and Analytics 2008**
 Edited by John F. Roddick, Jiuyong Li, Peter Christen and Paul Kennedy. November, 2008. 978-1-920682-68-2. Contains the proceedings of the 7th Australasian Data Mining Conference (AusDM 2008), Adelaide, Australia. December 2008.
- Volume 88 - Koli Calling 2007**
 Edited by Raymond Lister University of Technology, Sydney and Simon University of Newcastle. November, 2007. 978-1-920682-69-9. Contains the proceedings of the 7th Baltic Sea Conference on Computing Education Research.
- Volume 89 - Australian Video**
 Edited by Heng Tao Shen and Michael Frater. October, 2008. 978-1-920682-70-5. Contains the proceedings of the 1st Australian Video Conference.
- Volume 90 - Advances in Ontologies**
 Edited by Thomas Meyer, Meraka Institute, South Africa and Mehmet Orgun, Macquarie University, Australia. September, 2008. 978-1-920682-71-2. Contains the proceedings of the Knowledge Representation Ontology Workshop (KROW 2008), Sydney, September 2008.
- Volume 91 - Computer Science 2009**
 Edited by Bernard Mans Macquarie University. January, 2009. 978-1-920682-72-9. Contains the proceedings of the Thirty-Second Australasian Computer Science Conference (ACSC2009), Wellington, New Zealand, January 2009.
- Volume 92 - Database Technologies 2009**
 Edited by Xuemin Lin, University of New South Wales and Athman Bouguettaya, CSIRO. January, 2009. 978-1-920682-73-6. Contains the proceedings of the Twentieth Australasian Database Conference (ADC2009), Wellington, New Zealand, January 2009.
- Volume 93 - User Interfaces 2009**
 Edited by Paul Calder Flinders University and Gerald Weber University of Auckland. January, 2009. 978-1-920682-74-3. Contains the proceedings of the Tenth Australasian User Interface Conference (AUIC2009), Wellington, New Zealand, January 2009.
- Volume 94 - Theory of Computing 2009**
 Edited by Prabhhu Manyem, University of Ballarat and Rod Downey, Victoria University of Wellington. January, 2009. 978-1-920682-75-0. Contains the proceedings of the Fifteenth Computing: The Australasian Theory Symposium (CATS2009), Wellington, New Zealand, January 2009.
- Volume 95 - Computing Education 2009**
 Edited by Margaret Hamilton, RMIT University and Tony Clear, Auckland University of Technology. January, 2009. 978-1-920682-76-7. Contains the proceedings of the Eleventh Australasian Computing Education Conference (ACE2009), Wellington, New Zealand, January 2009.
- Volume 96 - Conceptual Modelling 2009**
 Edited by Markus Kirchberg, Institute for Infocomm Research, A*STAR, Singapore and Sebastian Link, Victoria University of Wellington, New Zealand. January, 2009. 978-1-920682-77-4. Contains the proceedings of the Fifth Asia-Pacific Conference on Conceptual Modelling (APCCM2008), Wollongong, NSW, Australia, January 2008.
- Volume 97 - Health Data and Knowledge Management 2009**
 Edited by James R. Warren, University of Auckland. January, 2009. 978-1-920682-78-1. Contains the proceedings of the Third Australasian Workshop on Health Data and Knowledge Management (HDKM 2009), Wellington, New Zealand, January 2009.
- Volume 98 - Information Security 2009**
 Edited by Ljiljana Brankovic, University of Newcastle and Willy Susilo, University of Wollongong. January, 2009. 978-1-920682-79-8. Contains the proceedings of the Australasian Information Security Conference (AISC 2009), Wellington, New Zealand, January 2009.
- Volume 99 - Grid Computing and e-Research 2009**
 Edited by Paul Roe and Wayne Kelly, QUT. January, 2009. 978-1-920682-80-4. Contains the proceedings of the Australasian Workshop on Grid Computing and e-Research (AusGrid 2009), Wellington, New Zealand, January 2009.
- Volume 100 - Safety Critical Systems and Software 2007**
 Edited by Tony Cant, Defence Science and Technology Organisation, Australia. December, 2008. 978-1-920682-81-1. Contains the proceedings of the 13th Australian Conference on Safety Critical Systems and Software, Canberra Australia.
- Volume 101 - Data Mining and Analytics 2009**
 Edited by Paul J. Kennedy, University of Technology, Sydney, Kok-Leong Ong, Deakin University and Peter Christen, The Australian National University. November, 2009. 978-1-920682-82-8. Contains the proceedings of the 8th Australasian Data Mining Conference (AusDM 2009), Melbourne Australia.