Colin Boyd
Juan M. González Nieto (Eds.)

LNCS 3574

Information Security and Privacy

10th Australasian Conference, ACISP 2005
Brisbane, Australia, July 2005
Proceedings

Springer

Lecture Notes in Computer Science 3574

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

Colin Boyd Juan M. González Nieto (Eds.)

Information Security and Privacy

10th Australasian Conference, ACISP 2005
Brisbane, Australia, July 4-6, 2005
Proceedings

Springer

Volume Editors

Colin Boyd
Juan M. González Nieto
Queensland University of Technology
Information Security Institute
GPO Box 2434, Brisbane 4000, Australia
E-mail: c.boyd@qut.edu.au, juanma@isrc.qut.edu.au

Library of Congress Control Number: 2005928379

CR Subject Classification (1998): E.3, K.6.5, D.4.6, C.2, E.4, F.2.1, K.4.1

ISSN 0302-9743
ISBN-10 3-540-26547-3 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-26547-4 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Boller Mediendesign
Printed on acid-free paper SPIN: 11506157 06/3142 5 4 3 2 1 0

Preface

The 2005 Australasian Conference on Information Security and Privacy was the tenth in the annual series that started in 1996. Over the years ACISP has grown from a relatively small conference with a large proportion of papers coming from Australia into a truly international conference with an established reputation. ACISP 2005 was held at Queensland University of Technology in Brisbane, during July 4–6, 2005.

This year there were 185 paper submissions and from these 45 papers were accepted. Accepted papers came from 13 countries, with the largest proportions coming from Australia (12), China (8) and Japan (6). India and Korea both contributed 2 papers and one came from Singapore. There were also 11 papers from European countries and 3 from North America. We would like to extend our sincere thanks to all authors who submitted papers to ACISP 2005.

The contributed papers were supplemented by four invited talks from eminent researchers in information security. The father-and-son team of Prof. and Dr. Bob Blakley (Texas A&M University and IBM) gave a talk entitled "All Sail, No Anchor III," following up on a theme started at their ACISP 2000 invited talk. Adrian McCullagh (Phillips Fox Lawyers and QUT) talked on the benefit and perils of Internet banking. Ted Dunstone (Biometix) enlightened us on multimodal biometric systems. Yvo Desmedt (University College London) elucidated the growing gap between theory and practice in information security.

We were fortunate to have an energetic team of experts who formed the Program Committee. Their names may be found overleaf, and we thank them warmly for their considerable efforts. This team was helped by an even larger number of individuals who reviewed papers in their particular areas of expertise. A list of these names is also provided; we hope it is complete.

We are delighted to acknowledge the generous financial sponsorship of ACISP 2005 by Eracom Technologies and RNSA (a research network funded by the Australian Research Council). The conference was hosted by the Information Security Institute at Queensland University of Technology who provided first-class facilities and material support. The excellent Local Organizing Committee was led by the ACISP 2005 General Chair, Ed Dawson, and included Lauren May, Elizabeth Hansford and Christine Orme. We made use of electronic submission and reviewing software expertly written and supported by Andrew Clark from the Information Security Institute at QUT; this software was invaluable in easing our administrative tasks.

July 2005

Colin Boyd
Juan M. González Nieto

ACISP 2005
10th Australasian Conference on Information Security and Privacy

Sponsored by
Information Security Institute, Queensland University of Technology
ARC Research Network for a Secure Australia (RNSA)
Eracom Technologies Pty. Ltd.

General Chair

Ed Dawson	*Queensland University of Technology, Australia*

Program Chairs

Colin Boyd	*Queensland University of Technology, Australia*
Juan M. González Nieto	*Queensland University of Technology, Australia*

Program Committee

Paul Ashley	*IBM, Australia*
Tuomas Aura	*Microsoft Research, UK*
Feng Bao	*Institute for Infocomm Research, Singapore*
Lynn Batten	*Deakin University, Australia*
Matt Bishop	*University of California at Davis, USA*
Bob Blakley	*Texas A&M University, USA*
Mike Burmester	*Florida State University, USA*
Marc Dacier	*Eurecom, France*
Yvo Desmedt	*University College London, UK*
Josep Domingo	*Universitat Rovira i Virgili, Spain*
Jordi Forné	*Universitat Politècnica de Catalunya, Spain*
Virgil Gligor	*University of Maryland, USA*
Dieter Gollmann	*TU Hamburg-Harburg, Germany*
Peter Gutmann	*University of Auckland, New Zealand*
Bill Hutchinson	*Edith Cowan University, Australia*
Audun Josang	*DSTC, Australia*
Marc Joye	*CIM-PACA, France*
Svein Knapskog	*Norwegian University of Science and Technology, Norway*

Byoungcheon Lee	*Joongbu University, Korea*
Javier López	*University of Málaga, Spain*
Wenbo Mao	*HP Laboratories, UK*
Chris Mitchell	*Royal Holloway, UK*
George Mohay	*QUT, Australia*
Paul Montague	*Motorola, Australia*
SangJae Moon	*Kyungpook National University, Korea*
Winfried Mueller	*University of Klagenfurt, Austria*
Eiji Okamoto	*University of Tsukuba, Japan*
Susan Pancho-Festin	*University of the Philippines, Philippines*
Radia Perlman	*Sun Microsystems, USA*
Josef Pieprzyk	*Macquarie University, Australia*
Bart Preneel	*Katholieke Universiteit Leuven, Belgium*
Pandu Rangan	*Indian Institute of Technology, India*
Anthony Rhodes	*Zayed University, UAE*
Carsten Rudolph	*Fraunhofer SIT, Germany*
Rei Safavi-Naini	*University of Wollongong, Australia*
Pierangela Samarati	*University of Milan, Italy*
Akashi Satoh	*IBM Research, Japan*
Jennifer Seberry	*University of Wollongong, Australia*
Miquel Soriano	*Universitat Politècnica de Catalunya, Spain*
Sridha Sridharan	*QUT, Australia*
Vijay Varadharajan	*Macquarie University, Australia*
Kapali Viswanathan	*SETS, India*
Huaxiong Wang	*Macquarie University, Australia*
Matt Warren	*Deakin University, Australia*
Chuan-Kun Wu	*Australian National University, Australia*
Yuliang Zheng	*University of North Carolina, Charlotte, USA*

External Reviewers

Riza Aditya
Isaac Agudo
Toru Akishita
Stig Andersson
André Årnes
Joonsang Baek
Mark Branagan
Gareth Brisbane
Jordi Castellà-Roca
Vinod Chandran
Liqun Chen
Joe Cho
Mathieu Ciet
Andrew Clark
Scott Contini
Nora Dabbous
Breno de Medeiros
Christophe De Cannière
Alex Dent
Jintai Ding
Hans Dobbertin
Christophe Doche
Jiang Du
Oscar Esparza
Serge Fehr
Clinton Fookes
Steven Galbraith
Praveen Gauravaram
Pierre Girard
Goichiro Hanaoka
Keith Harrison
Yvonne Hitchcock
Michael Hitchens
Zhenjie Huang
Sarath Indrakanti
Kouichi Itoh
Udaya Kiran Tupakula
Lars Knudsen
Joe Lano
HoonJae Lee
Ching Lin
Ling Liu
Subhamoy Maitra
Antoni Martínez-Ballesté
Michael Mason
Anish Mathuria
Bill Millan
Jose A. Montenegro
Sumio Morioka
Yi Mu
Jose Luis Muoz
Gregory Neven
Lan Nguyen
Katsuyuki Okeya
Jose A. Onieva
Kenny Paterson
Josep Pegueroles
Kun Peng
Angela Piper
Fabien Pouget
Geraint Price
Michaël Quisquater
Jason Reid
Christian Ritz
Bruno Robisson
Rodrigo Roman
Chun Ruan
Francesc Sebé
Bouchra Senadji
Leonie Simpson
Nigel Smart
Agusti Solanas
Martijn Stam
Ron Steinfeld
Chris Steketee
Hung-Min Sun
Willy Susilo
Gelareh Taban
Dong To
Guillaume Urvoy-Keller
Tri Van Le
N. Vijayarangan
R. Vijayasarathy
Guilin Wang
Yongge Wang
Duncan S. Wong
Yongdong Wu
Alec Yasinsac
Fangguo Zhang
Janson Zhang
Weiliang Zhao
Yunlei Zhao
Huafei Zhu
Jacob Zimmermann

Table of Contents

Elliptic Curve Cryptography

Mobile Security

Side Channel Attacks

Evaluation and Biometrics

Public Key Cryptosystems

Access Control I

Signatures I

Invited Talk

Access Control II

Threshold Cryptography

Protocols I

Group Signatures

Protocols II

Signatures II

Invited Talk

Credentials

Symmetric Cryptography

All Sail, No Anchor III: Risk Aggregation and Time's Arrow

Bob Blakley[1] and G.R. Blakley[2]

[1] IBM,
Austin, TX 78758, USA
blakley@us.ibm.com
[2] Texas A&M University
College Station, TX 77843-3368, USA
blakley@math.tamu.edu

Abstract. This paper explains why protection mechanisms which distribute even the protected forms of information assets lead to increased risks. It describes a mechanism (called a "lethal secret sharing system") which enables the imposition of "forgetfulness" by an asset owner on the receiver of a protected asset. This forgetfulness, or "lethe", is enforced by allowing the asset owner to give information about a piece of knowledge to the asset receiver in such a way that the receiver can be prevented at a future time from using the knowledge to recover the information.

1 Introduction

All Sail, No Anchor I: Cryptography, Risk, and e-Commerce [1] examined how the passage of time creates risk in electronic information systems.

This paper will extend the discussion to consider how re-use of cryptographic and other information protection artifacts aggregates risks and thereby makes electronic information systems more dangerous as time passes.

When a system is designed so that a cascade of failures, or a single failure with multiple adverse results can lead to large losses, that system is said to aggregate risks. Many critical systems are designed to avoid risk aggregation problems by eliminating "single points of failure".

Risk aggregation is often thought of in spatial terms (routing hydraulics through a single point in airliners; siting multiple telecommunications hostels in the basement of a single building, etc...) But it can also be thought of in temporal terms - adding more risk to a single artifact or system over time.

Information security systems often aggregate risks in poorly-understood ways, by re-using protection mechanisms which are assumed to be very strong but whose strength is in fact poorly understood.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 1-16, 2005.

2 Instantaneous Protection Decay

We consider two information protection scenarios which are common in today's information systems, and discuss how each of these scenarios leads to aggregation of risk.

Encrypted Storage

Information systems often protect sensitive data against disclosure by encrypting the data when it is stored on media.
In this scenario:

- The asset (for example, a file or database) is protected for a long time.
- The protected form of the asset is in the possession of its owner.
- Compromise of one asset does not (necessarily) diminish the protection of other assets.
- Compromise of a single cryptographic key can devalue multiple assets – but smart key management can help with this by ensuring that a unique key is used for each asset, as can smart management of physical media and access thereto.
- Compromise of a mechanism (e.g. cryptosystem or block cipher mode of operation) can devalue multiple assets, but smart management of physical media and accesss thereto can help with this by allowing the owner to fall back to simple physical access protections when he learns that the mechanism has been broken.

Digital Rights Management

Media distribution systems often protect valuable content against unlicensed use by encrypting the data before it is distributed, and relying upon specialized media players to prevent copying or other misuse of the content in violation of license terms.

In this scenario:

- The asset is protected for a long time.
- The asset (at least in its protected form) is in the possession of the enemy.
- Compromise of one asset does devalue other instances of the same asset, but does not necessarily weaken the protection of instances of different assets.
- Compromise of a single key may devalue multiple assets, but smart key management may be able to help with this, by ensuring that a unique key is used to protect each asset.
- Compromise of a mechanism can devalue all assets.

3 Risk Aggregation

[1] introduced a taxonomy of asset types to facilitate discussions of the evolution of risk over time. That paper went on to discuss risk associated with "Alice-type"

artifacts (title deeds, for example, which can be moved back and forth between bit-space and atom-space because they are just pointers to things with value in the physical world), and "Bob–type" artifacts (like digitized movies, which can't move back and forth, but remain in bit-space all the time because their digital form has intrinsic value).

The discussion in [1] assumed that the effective strength of keys and algorithms decays at some rate over time, and that the value of artifacts might also change in various ways over time, and went on to describe how to protect artifacts of various types under that assumption.

The current discussion will make a different set of assumptions: it will assume that artifacts of all types share a single value, which remains constant over time. It will also assume that cryptographic keys and protection mechanisms decay in effectiveness instantaneously (that is, that their effectiveness goes effectively to zero instantaneously, as the result of a conceptual advance on the part of attackers), but at an unpredictable time.

It should be clear that security mechanisms aggregate risk over time as they are used to protect more assets; assuming that a mechanism uses a unique key to protect each asset and that a mechanism uses a single cryptosystem to protect all assets, figure 1 shows how the maximum risk exposure grows over time as the mechanism is used to protect more and more assets (the y-axis legend represents the value of a population of assets; recall that we've already assumed all assets have equal values which do not change over time):

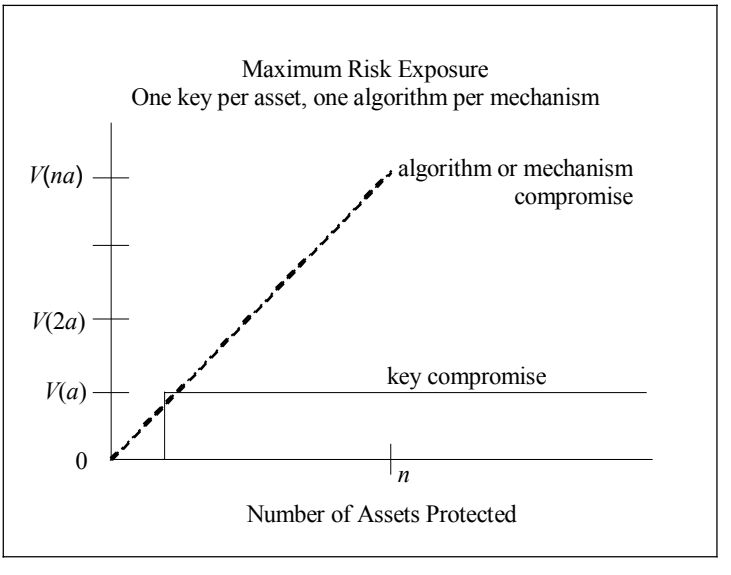


Figure 1

It should be noted here that risk due to compromise of one key remains constant over time, but that risks due to compromise of cryptographic algorithms and security mechanisms grows without bound as time passes.

In this context, we will introduce the following concepts:

- **key risk** is the aggregate decline in value caused by an instantaneous loss of effectiveness of a single cryptographic key.
- **cryptosystem risk** is the aggregate decline in value caused by an instantaneous loss of effectiveness of a single cryptographic algorithm.
- **mechanism risk** is the aggregate decline in value caused by an instantaneous loss of effectiveness of a single information security protection mechanism.

Consider the two scenarios described above, and assume that the same assets are being protected in each scenario.

The risks in the two scenarios are quite different:

Key Risk

In the encrypted storage scenario, this varies between zero (in case the owner learns of the key compromise, shuts off physical access to the system before any attacker can access the protected assets, and re-protects exposed assets using an unbroken key) and the sum of the values of all assets protected using the compromised key (in case the owner doesn't learn about the compromise until after the horse is out of the barn).

However, if multiple instances of an asset are protected under different keys, the instances protected under the uncompromised keys remain protected.

In the digital rights management scenario, this is always the sum of the values of all instances of all assets protected using the compromised key (because the assets are in possession of the enemy, even if the owner learns of the compromise, he can't prevent the enemy from using the compromise to use the assets in violation of their license terms, and he also can't prevent the enemy from making copies of the assets and distributing unprotected versions of them to the holders of all other protected instances).

Assuming that we use one key per asset in both scenarios, figure 2 shows how risk varies with the duration of an exposure due to key compromise.

Cryptosystem Risk

In the encrypted storage scenario, this varies between zero (in case the owner learns of the cryptosystem compromise, shuts off physical access to the system before any attacker can access the protected assets, and re-protects exposed assets using an unbroken cryptographic algorithm) and the sum of the values of all assets protected using the compromised cipher (in case the owner doesn't learn about the compromise until after the horse is out of the barn). However, if multiple instances of an asset are protected under different cryptosystems, the instances protected under the uncompromised systems remain protected.

In the digital rights management scenario, this is always the sum of the values of all instances of all assets protected using the compromised cryptosystem (because the assets are in possession of the enemy, even if the owner learns of the compromise, he can't prevent the enemy from using the compromise to use the assets in violation of their license terms, and he also can't prevent the enemy from making copies of the assets and distributing unprotected versions of them to the holders of all other protected instances).

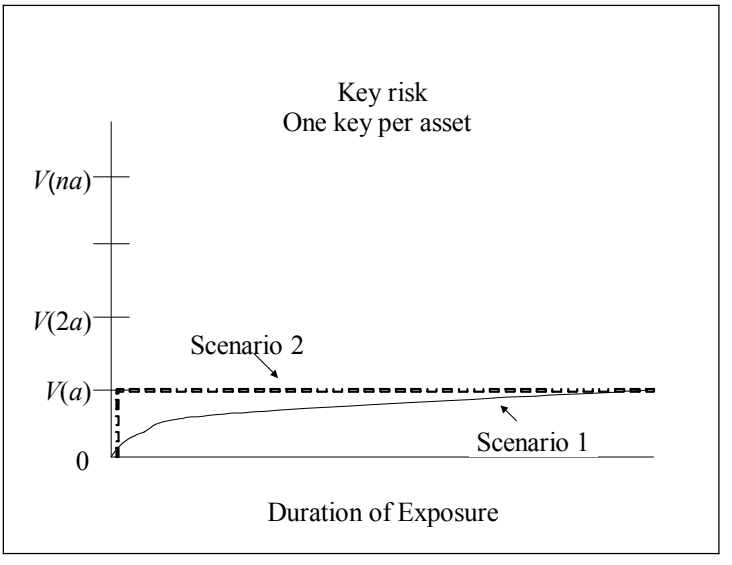


Figure 2

Again assuming one key per asset, and assuming also that protection mechanisms use only a single cryptosystem, figure 3 shows how risk varies with the duration of an exposure due to cryptosystem compromise:

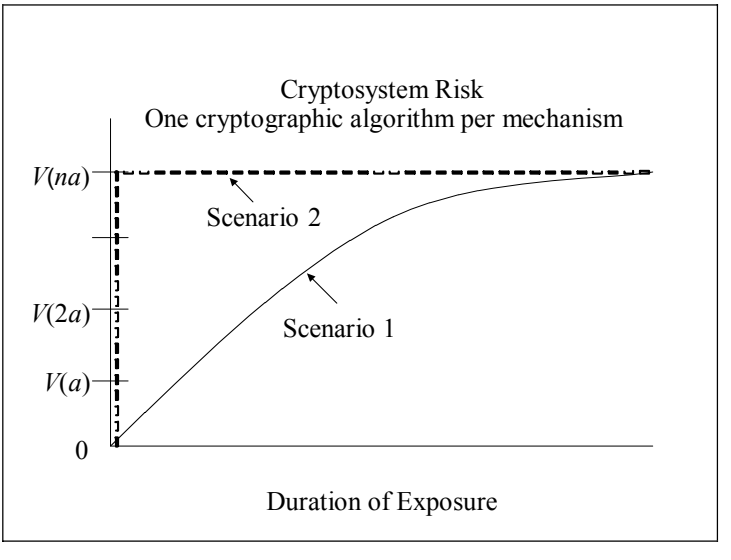


Figure 3

Mechanism Risk

In the encrypted storage scenario, this varies between zero (in case the owner learns of the key compromise, shuts off physical access to the system before any attacker can access the protected assets, and re-protects exposed assets using an unbroken key) and the sum of the values of all protected assets (in case the owner doesn't learn about the compromise until after the horse is out of the barn).

In the digital rights management scenario, this is always the sum of the values of all assets protected using the compromised mechanism (because the assets are in possession of the enemy, even if the owner learns of the compromise, he can't prevent the enemy from using the compromise to use the assets in violation of their license terms).

Again assuming one key per asset, and assuming that protection mechanisms use only a single cryptosystem, figure 4 shows how risk varies with the duration of an exposure due to mechanism compromise:

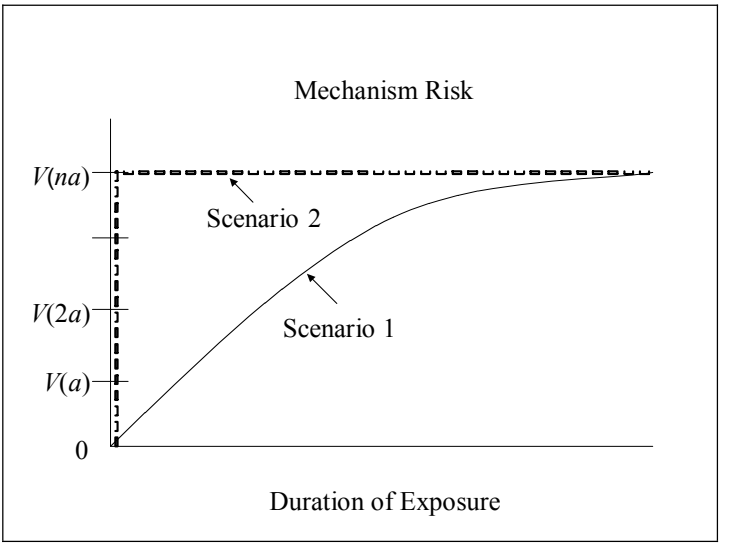


Figure 4

Note that in all cases the risks created by the digital rights management scenario are greater than or equal to the greatest risk possible in the encrypted storage scenario. It is worth considering why this is so. In the encrypted storage scenario, the consequences of a risk are mitigated by the ability of the owner to respond to news of a compromise by employing an uncompromised protection measure to prevent exploitation of the compromised key, cryptosystem, or mechanism by a potential attacker.

In the digital rights management scenario, on the other hand, the distribution of protected assets to the enemy aggregates key, cryptosystem, and mechanism risks *in time* by ensuring that the enemy's access to assets protected using the compromised protection artifact cannot be cut off. We call this type of risk aggregation

"instantaneous protection decay". In the Encrypted storage scenario, we can have instantaneous effectiveness decay for keys, cryptographic algorithms, and protection mechanisms without causing instantaneous protection decay, because of the feasibility of interposing a new mechanism between assets and attackers before all the assets have been compromised. In the Digital Rights Management scenario, instantaneous decay of a protection mechanism always causes instantaneous protection decay for all assets; instantaneous decay of cryptographic algorithms or cryptographic keys always causes instantaneous protection decay for all assets protected by those algorithms or keys.

4 System Characteristics Contributing to Risk Aggregation Problems

Risks are aggregated in information security systems when:

- Multiple assets are protected using the same cryptographic key.
- Multiple assets are protected using the same cryptographic algorithm, but different keys.
- Multiple assets are protected using the same protection mechanism, which employs different cryptographic algorithms.
- Protected forms of assets are exposed to enemies for extended periods of time.
- Enemies are able to copy and retain protected forms of assets.
- Protected forms of assets are exposed to multiple enemies at the same time.

These conditions are cumulative in the sense that when more conditions apply, risk aggregation becomes a more serious concern. A system which employs a single protection mechanism based on a single cryptographic algorithm and a single key, and which irrevocably distributes all its protected assets to everyone is the worst possible system from the viewpoint of risk aggregation.

5 Recovery Actions to Limit Risk Aggregation

To limit the types of risk aggregation discussed in this paper, system designers could consider:

- Ensuring that assets cannot be accessed until they are required, even when they are in protected form.
- Ensuring that the owner of an asset can cut off all access to protected assets in the event of a protection compromise, in order to re-protect assets using new methods.
- Using inherently robust protection mechanisms (i.e. mechanisms like one-time pads and secret-sharing systems, which are provably immune to compromise except by brute-force, instance-by-instance methods)
- Using multiple protection mechanisms per asset to ensure that detected compromises can be addressed before significant value loss occurs.

We should note that the first two of these techniques essentially address temporal aspects of risk aggregation; the third addresses mechanism strength, and the last addresses an essentially topological (or "spatial") aspect of risk aggregation.

6 To Cancel Half a Line

So far, we have treated system decay, algorithm decay and message decay as both inexorable and irreversible.

This posture is natural enough, and well precedented. According to Fitzgerald [4], one of the 12th Century's foremost mathematicians wrote something along the following lines:

> *The moving finger writes, and having writ,*
> *moves on. Nor all your piety and wit*
> *shall lure it back to cancel half a line,*
> *nor all your tears wash out a word of it.*

This suggests, among other things, that there is no way to retroactively force forgetfulness of any aspect of your communications on a collaborator or an opponent who has had access to them.

Forgetting

However, in the context of games, one of the 20th Century's foremost mathematicians coauthored these sentiments in three astounding pages [7] of a very influential book:

> Anteriority (*i. e.* the chronological ordering of the moves) possesses the property of transitivity
>
> Preliminarity [essentially the property that the "preliminary" move is known by the player making the current move] implies anteriority, but need not be implied by it
>
> preliminarity need not be transitive. Indeed it is neither in Poker nor in Bridge,
>
> in Bridge...this intransitivity... involves only one player, [and] … the necessary "forgetting"... was achieved by "splitting the personality" of [player] 1 into [North] and [South].
>
> All important examples of intransitive preliminarity are games containing chance moves. This is peculiar, because there is no apparent connection between these two phenomena.

We imagine a game like the following:

At time t we have n players pl_1 … pl_n. Each player pl_i has been "dealt" some set of information $pl_{i,t}$.

Within the game there is a function *fitok* which turns information into knowledge, so that each player has, at time t, knowledge $kn_{i,t} = fitok(p_{i,t})$.

Can we rig the game so that at time t+1, we distribute additional information such that every player's information set either grows or stays the same (i.e. so that $pl_{i,t} \subseteq pl_{i,t+1}$ for all i), while at least one player's knowledge *decreases* (i.e. $kn_{i,t+1} \subset kn_{i,t}$ for some i)?

von Neumann and Morgenstern might have had difficulty convincing Omar Khayyam, but the answer is "yes".

We here propose, within cryptography, an analog of the game-theoretic enforced forgetfulness which Bridge exhibits. We propose to call it *lethal secret sharing* in memory of the underworld river [6] of enforced lethe.

For simplicity we describe it only within the context of k-out-of-n threshold schemes [2, 5], though it can also easily be implemented within the more general context of *d*-from-*k*-out-of-*n* ramp schemes [3]. Extending lethality from threshold schemes to access structures, however, is not straightforward and will not be considered in this paper.

Background: Secret Sharing

As a brief reminder, before launching into the details, we note that a (k, n) threshold scheme amounts to the Shannon perfectly secure (1, k, n) case of the (more efficient but merely Shannon relatively secure [3]) notion of ramp scheme. And both can be implemented in either Blakley (geometric) or Shamir (algebraic) fashion.

A (k, n) threshold scheme employs a dealer, Delia, who has chosen the positive integers k and n (obeying the inequality $k \leq n$), as well as a -- probably quite large -- finite field $F = GF(p^n)$. She also possesses a random number generator G which produces members of a F cheaply and quickly in a manner subject to a uniform probability density function. Or, at least, she and her collaborators and opponents are unable to profit from an assumption to the effect that they are not dealing with independently uniformly distributed outputs from said generator.

Additionally, Delia knows every member p of a set P of n players. Any subset $C \subseteq P$ is called a coalition. In a (k, n) secret sharing scheme, such a subset C is a *small coalition* if it consists of fewer than k players. A subset C of P is called an *allowed coalition* if it contains at least k players.

A source of secrets, Fern, furnishes a secret s (chosen somehow from the field F) to Delia, who employs G in the performance of a *deal, Δ,* which shares s among the players.

This sharing process is done in such a way that every "share" is a clue regarding a safe $f = f(s)$ which conceals (or protects, if you prefer) the secret s. In the major practical instances of secret sharing, the clues reveal aspects of the location of f.

This deal Δ uses the arithmetic of F to combine the secret s with outputs of the random number generator G so as to give each player p his own *share* $h = h(p) = h(k, n, F, \mathrm{P}, p, s, \Delta)$ of the secret s.

Each share h can be a member of F or something else defined in terms of F (such as a subspace of a space which depends on F, or a point on a graph in $F \times F$).

Every share yields information about the safe f. The more shares you have, the more you know about (the location of) f. But you remain in total ignorance about the secret s until you have k shares.

A *showdown* is defined to be a meeting of the members of a coalition *C* to pool -- and to spend as much time and effort as they deem appropriate in processing -- their shares in such a manner as to find as much information as possible about the safe *f* protecting the secret *s*.

A showdown does not logically require a "referee" (an additional participant who may or may not have knowledge and powers the players lack). But lethality requires a referee, Rory. And the secret sharing process could even be rigged in such a way that it is impossible for the players to recover the secret s without the consent and participation of Rory.

If all the members of a coalition *C* are members of a coalition *C**, then the coalition *C** embodies *no less knowledge* about the safe f than the coalition C does.

In a (*k*, *n*) threshold scheme, an allowed coalition *C* knows everything about the safe *f* (though perhaps only with the help of Rory). With this knowledge, it is a simple matter to calculate the secret *s* quickly with certainty during a coalition *C* showdown.

When a small coalition *C* stages a showdown, it can assemble a considerable body of knowledge about the safe *f,* but (even with the active assistance of Rory) not enough to alter its total ignorance of the secret *s*, in the sense in which Shannon perfect security defines total ignorance.

Small coalitions learn nothing about *s* (or, more exactly, the showdown teaches them nothing about *s* which was not common knowledge before the deal). Allowed coalitions learn everything about *s* (or, more exactly, the showdown replaces their common-knowledge pre-deal assessment of likely and unlikely values of *s* by a "certainty" that they know what *s* – the secret being shared – is).

The exact description of this Shannon perfect security feature is cast in probabilistic language as follows.

"Common knowledge before the deal" amounts to an *a priori* probability density function, presumed to be used by everybody (whether collaborator, opponent or bystander)

$Pri : F \rightarrow [\ 0, 1\ ]$

This pdf tells how probable the occurrence of a member *m* of the field *F* is (as a typical message emanating from the source Fern who has secrets to share). For this pdf,

Pri (*m* = *s*) = PROB (*m* = *s* according to common knowledge)

And for each coalition *C* there is an *a posteriori* pdf

$PosC : F \rightarrow [\ 0, 1\ ]$

which *C* calculates on the basis of both *Pri* and all the showdown information (including referee input if that is required) pooled by it. For this pdf,

PosC (*m* = *s* | all the C information, and referee input, together with *Pri*)

At showdown time there are two completely different sorts of outcomes. If *C* is an allowed coalition, then the function *PosC* is a Dirac delta, having value 1 at some one point *m* (it will, of course, be equal to s) of *F*, and therefore having value 0 everywhere else. But if *C* is a small coalition, then the pdf *PosC* is equal to the pdf *Pri*.

For example, in a Blakley scheme the safe *f* is a point and the secret *s* is its first coordinate. Larger and larger coalitions can determine smaller and smaller affine subspaces containing *f*. But s remains totally unknown until the subspace is just $\{f\}$.

In a Shamir scheme the safe f is a polynomial, and the secret s is the value of f at 0. Larger and larger coalitions can tell more and more about combinations of the coefficients of f. But s remains totally unknown until f is uniquely specified.

It is clear that publication of the share of one player in a deal Δ of a $(k + 1, n + 1)$ threshold scheme employed upon a secret s confronts the other n players with exactly the same recovery problem as a deal Δ^* of a (k, n) threshold scheme employed upon that same secret s.

So here we have Omar's moving finger. If it writes a player's share out for all the world to see, it automatically converts a $(k + 1, n + 1)$ threshold scheme recovery of s into (k, n) threshold scheme recovery of s. How could you go about disremembering such a revelation?

Lethal Secret Sharing

Is there a von Neumann/Morgenstern type of lethe expedient available? The best possible affirmative answer to this question would be a way to start with a $(k + 1, n + 1)$ threshold scheme T, perceive the leak which (in effect) degrades T to a (k, n) scheme as soon as it occurs, and then immediately snap T back into $(k + 1, n + 1)$ status.

Failing that, which partial rehabilitation from (k, n) status would be preferable? To $(k + 1, n)$ status? Or to $(k, n + 1)$ status? We say the former.

We will define a *lethal secret sharing system* as one in which we can recover from a leak of one share and rehabilitate the scheme from (k, n) status to $(k + 1, n)$ status.

The job, then, is to figure out an early modification (perhaps even at deal time or before) of the workings of a $(k + 1, n + 1)$ threshold scheme. Let us recall than an allowed coalition in such a scheme has at least $k + 1$ members. A coalition with only k members is small.

Constructing a Lethal Secret Sharing System

The sort of modification we seek will be called a lethe ("forgetting" is too weak a locution) of the share $h(n+1)$ assigned to the (now discredited and disenrolled) player $p(n+1)$. It must be a procedure which renders any coalition of k of the remaining n players incapable of finding s in a showdown even if they are also making use of the additional $k + 1^{st}$ share $h(n+1)$. The way we propose involves a referee Rory.

The modified threshold scheme will be called a lethal $(k + 1, n + 1)$ threshold scheme. In such a scheme,

The dealer Delia encrypts the secret s, which Fern furnished, as a “pseudosecret” $\sigma = H(\kappa, s)$ in cryptosystem H using key κ.

In the ordinary threshold scheme fashion, Delia produces "preshares"

$$\pi(1), \pi(2),..., \pi(n), \pi(n+1)$$

of the "pseudosecret" $\sigma = H(\kappa, s)$.

Delia encrypts the preshares $\pi(w)$ so as to produce "pseudoshares"

$$\psi[1] = J[1](\lambda(1), \pi(1)),$$
$$\psi[2] = J[2](\lambda(2), \pi(2)),$$
$$...,$$
$$\psi[n] = J[n](\lambda(n), \pi(n)),$$
$$\psi[n+1] = J[n+1](\lambda(n+1), \pi(n+1)),$$

where cryptosystems

$$J[1], J[2],..., J[n], J[n+1]$$

employ keys

$$\lambda(1), \lambda(2),..., \lambda(n), \lambda(n+1)$$

to encrypt the plaintexts

$$\pi(1), \pi(2),..., \pi(n), \pi(n+1).$$

Delia deals these pseudoshares $\psi(w)$ as if they were shares, so that the w^{th} player $p(w)$ gets $\psi[w] = J[w](\lambda(w), \pi(w))$. But, in addition, the w^{th} player $p(w)$ also learns what cryptosystem $J[w]$ he is associated with.

But player $p(w)$ knows neither H nor $\lambda(w)$ nor $\pi(w)$.

Delia tells Rory all her cryptosystem information H, $J[1]$, $J[2]$,..., $J[n]$, $J[n+1]$ and all her key information κ, $\lambda(1)$, $\lambda(2)$,..., $\lambda(n)$, $\lambda(n+1)$

Rory, however, knows neither the secret s, nor the pseudosecret σ, nor any information about the safe f. Nor does he know anything about any $\psi(w)$ or any $\pi(w)$.

Outsiders know the field F, the random number generator G, the integers k and n, all the personalities and roles and rules. But they know no Js, no λs, no πs, and no ψs. Nor do they know H, κ, f, or s.

And Fern and Delia promptly die (as a consequence of which they forget everything, since they have to drink from Lethe in order to get into Hades, of course). This is merely for clarity. There is no logical necessity for Fern or Delia, or indeed any human being whatsoever, to know anything about the secret s at any time whatsoever.

Nevertheless we do away with them simply to emphasize that -- immediately after the deal in the lethal threshold scheme -- literally nobody on earth knows anything about the secret s. The lethe affects only the referee Rory and the players. In fact, it affects only what Rory and a coalition C can come to learn from subsequent revelation of a share of a player who is not a member of C.

Recovery of a Secret

Before turning to the lethe feature, consider how the recovery of a secret is carried out when nothing untoward happens. In such a case the showdown-based recovery process is obvious.

An allowed coalition C forms and meets with Rory. Recall that C has more than k members (because the scheme is $(k+1,n+1)$, not (k,n)).

Rory knows the cryptosystems H and the $J[w]$, as well as the keys κ and the $\lambda[w]$. He now reveals them to the coalition C.

Consequently, he and they can emplace key $\lambda[w]$ in cryptosystem $J[w]$ so as to decrypt the pseudoshare $\psi(w)$ and thus recover the preshare $\pi(w)$.

They can do this for each w such that player $p[w]$ belongs to the coalition C.

Rory and the coalition C can then perform the ordinary showdown operations to recover the pseudosecret $\sigma = H(\kappa, s)$.

Thereafter, he and C can emplace key κ in cryptosystem H so as to decrypt σ and thus recover the original secret s.

So far a lethal threshold scheme merely looks like a cumbersome threshold scheme. It is that. But not merely that.

Lethe: Forgetting a Share

But now consider why it is possible to recover from the unauthorized publication of a pseudoshare in the $(k+1,n+1)$ scheme.

Suppose that one player's share is publicized. There is no harm in assuming that he is player $p[n + 1]$.

The referee Rory hears about this. He produces n new keys

$$\mu(1), \mu(2),..., \mu(n)$$

at random and sends them over open channels to the players with the instruction that each player $p(w)$ use key $\mu(w)$ in cryptosystem $J[w]$ to encrypt $\psi(w)$ so as to produce a modified pseudoshare

$$\psi^*(w) = J[w] (\mu(w), \psi(w)).$$

Rory doesn't know any $\psi^*(w)$.

Outsiders know nothing other than $\mu(1), \mu(2),..., \mu(n)$, and $\psi(n + 1)$

Suppose that an allowed coalition C assembles. It has at least $k + 1$ members, none of whom is player $p(n + 1)$.

Its members know the $\psi^*(w)$ and the $\psi(w)$. But without Rory they don't know the keys $\lambda(w)$ to insert into the cryptosystems they know.

So Rory uses the keys $\mu(w)$ in the cryptosystems $J[w]$ to decrypt the $\psi^*(w)$ (just to keep the players honest) to obtain the $\psi(w)$. He then uses the keys $\lambda(w)$ in the cryptosystems $J[w]$ to obtain the preshares $\pi(w)$. He and the members of C then use the usual showdown methodology to find the pseudosecret $\sigma = H (\kappa, s)$.

Rory then emplaces the key κ into the cryptosystem H to decrypt σ and obtain s.

A merely k-member coalition C^*, accompanied by player $p(n+1)$, would be a set of $k+1$ "players", but Rory would not cooperate with them.

It might be objected that Rory represents a possible single point of failure (whose demise would make successful recovery of the secret impossible). Such a flaw would be alien to the spirit of a threshold scheme. But there is nothing to prevent cloning him, or even taking more complicated expedients, which we will not consider here.

The point is that within secret sharing, as within game theory, it is possible to lure the moving finger back to cancel half a line (not an entire line in either case). And in neither endeavor have the authors encountered a way to do this without both

incorporating randomness and splitting up a single human being's natural activities and prerogatives.

7 An Example of Lethe

Let's see how a lethal secret-sharing scheme could be used to protect digital content.

Imagine that Delia is a digital movie distributor. Imagine further that she's got a Rory chip built into a digital movie player. Finally, imagine that she wants to distribute the movie to the player, and a set of licenses to people who have paid to see the movie (for the purposes of argument let's say two people have purchased licenses). Delia splits the movie into n+1 pseudoshares using a lethal secret-sharing scheme. She distributes n-1 pseudoshares ("the movie") to the player. She gives Rory the necessary key and cryptosystem information. And she distributes one pseudoshare each ("the licenses") to the two licensees. Figure 5 below shows the situation after Delia has distributed her information:

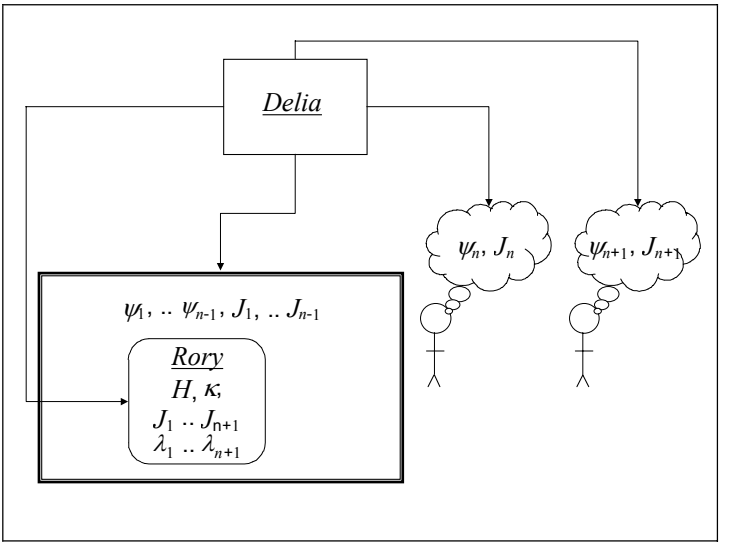


Figure 5

Now let's say one of the two licensees (for purposes of argument, the one who received pseudoshare n+1) publishes his pseudoshare to facilitate free viewing of the movie, as illustrated in figure 6.

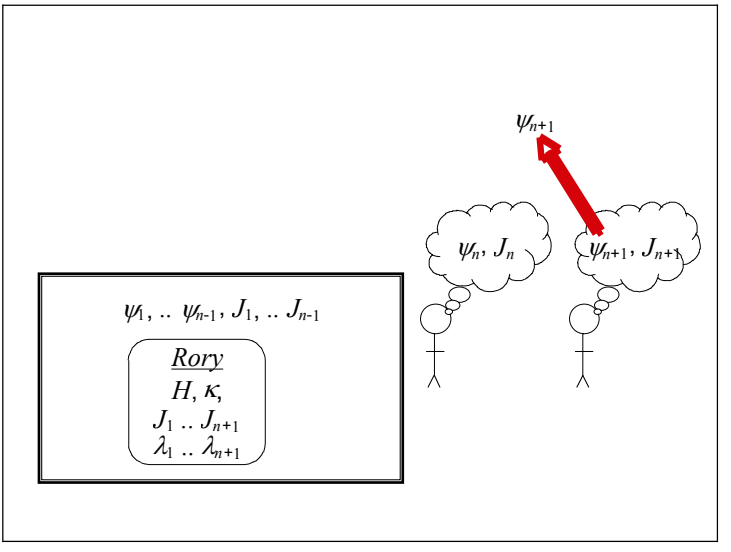


Figure 6

As soon as Rory learns about the publication of pseudoshare n+1, he can invalidate it without denying access to the user who received pseudoshare n by distributing a new set of pseudoshare-transformation keys as illustrated in figure 7.

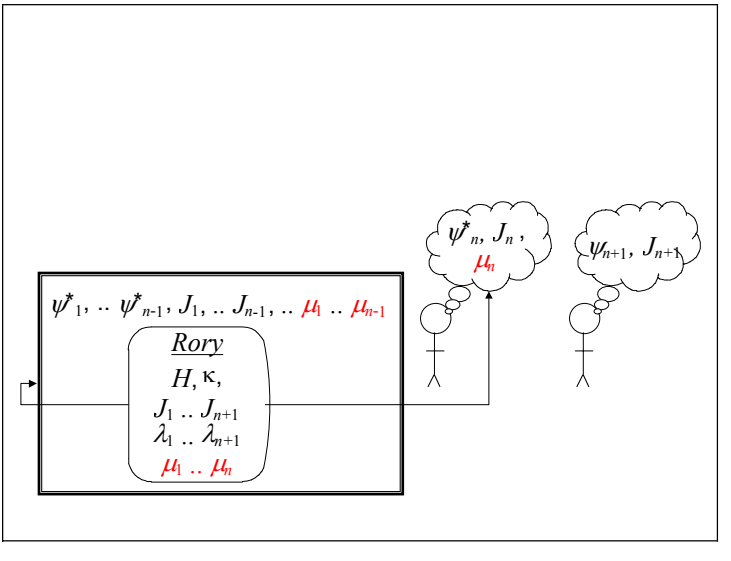


Figure 7

At this point, the user who received pseudoshare n and its transformation key μ_n can use his transformed pseudoshare ψ^*_n to view the movie, but no one who knows

pseudoshare n+1, including the user who originally received it, can use it to view the movie. In fact, no one can use *any* untransformed pseudoshare to view the movie.

8 Conclusion and Discussion

Designers of information security systems must consider the corrosive effects of time and space on the systems they build. [1] considered the effects of time on the protection of a single artifact, and counseled system designers to calibrate the strength (over time) of each protection mechanism to the value of the artifact it protects.

This paper goes further and asks designers to consider the camel's back: even a strong protection mechanism becomes an unacceptable risk when too much value is loaded onto it. Designers should take care to understand how much risk a mechanism is bearing in the real world already before using them to protect new assets, in much the same way as the designer of a building might consider how much load the structure is bearing now, and how much load it has borne over its previous history, before deciding whether it is safe to add another floor or a communications antenna to the top.

Finally, the paper demonstrates how a random deal combined with a referee enables "lethe": a type of forced forgetfulness which enables distribution of information about knowledge in a way which permits revocation of the ability to transform the information into the knowledge.

9 References

[1] Blakley, B., and G.R. Blakley, "All Sail, No Anchor I: Cryptography, Risk, and e-Commerce", ed. E. Dawson, A. Clark, C. Boyd, "Information Security and Privacy: Proceedings of 5th Australasian Conference on Information Security and Privacy", LNCS vol. 1841, Springer Verlag, Berlin, 2000, pp. 471-6.

[2] Blakley, G.R. "Safeguarding Cryptographic Keys", Proceedings of the National Computer Conference, American Federation of Information Processing Societies Press, vol. 48 (1979), June 1979, pp 242-68.

[3] Blakley, G.R., and C. Meadows, "Security of Ramp Schemes", Advances in Cryptology: Proceedings of Crypto '84, LNCS vol. 196, Springer-Verlag, Berlin, 1985, pp. 243-68.

[4] FitzGerald, Edward, tr., "The Rubaiyat of Omar Khayyam", ed. Christopher Decker, Bibliographical Society of the University of Virginia, 1997.

[5] Shamir, A. "How to Share a Secret", Communications of the ACM, vol. 24 no. 11, November 1979, pp. 612-13.

[6] Virgil, "Opera", ed. R. A. B. Mynors, Oxford University Press, Aeneid 6.705, Oxford, 1969.

[7] Von Neumann, J., and O. Morgenstern, "Theory of Games and Economic Behavior", Princeton University Press, Princeton, NJ, 1953.

Traversing Middleboxes with the Host Identity Protocol

Hannes Tschofenig[1], Andrei Gurtov[2], Jukka Ylitalo[3], Aarthi Nagarajan[4], and Murugaraj Shanmugam[4]

[1] Siemens, Germany
hannes.tschofenig@siemens.com
[2] Helsinki Institute for Information Technology, Finland
gurtov@cs.helsinki.fi
[3] Ericsson Research NomadicLab, Finland
jukka.ylitalo@nomadiclab.com
[4] Technical University Hamburg-Harburg, Germany
{murugaraj.shanmugam}@tu-harburg.de

Abstract. The limited flexibility of the Internet to support mobility has motivated many researchers to look for alternative architectures. One such effort that combines security and multihoming together is the Host Identity Protocol (HIP). HIP is a signaling protocol that adds a new protocol layer to the Internet stack between the transport and the network layer. HIP establishes IPsec associations to protect subsequent data traffic. Though the security associations are established solely between the communicating end hosts, HIP also aims to interwork with middleboxes such as NATs and firewalls. This paper investigates this interworking aspect and proposes a solution for secure middlebox traversal.

Keywords: Identifier-Locator Split, Host Identity Protocol, Middlebox, Network Address Translators (NATs), Firewalls, Authentication, Authorization.

1 Introduction

In the classical Internet architecture, an IP address serves as an address for packet delivery and as an identifier for the communicating end points. These roles are known as the locator and identifier respectively. The dual use of an IP address, although originally intended, nowadays limits the flexibility with regard to mobility and multihoming. In recent years, there have been many efforts to overcome this limitation through different approaches at different layers in the protocol stack. Existing solutions propose new indirection infrastructures, transport layer enhancements to support multiple locators, or adding new shim protocol layers. This paper looks at the compatibility issues of the Host Identity Protocol with NATs or firewalls and proposes a generic middlebox security solution.

The Host Identity Protocol (HIP) [1] is being developed by the IETF HIP working group. It is an identifier-locator separation mechanism that operates

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 17–28, 2005.

between the transport layer and the network layer. The Host Identity Protocol heavily relies on public key cryptography where every host generates a pair of keys: a private key and a public key. The public key is called the Host Identity (HI). A Host Identity Tag (HIT) is a 128-bit hash of the host's public key. The interface to the transport layer uses Host Identity Tags in place of IP addresses, while the interface to the Internet layer uses conventional IP addresses. In simple terms, transport connections and security associations are bound to HITs that do not change with changes of IP addresses. HIP is initialized with a base exchange mechanism that is used to quickly authenticate the hosts, exchange the keys to protect the rest of the base exchange and to form the required security associations to protect the payload.

HIP [1] starts with one of the hosts looking up the HI and IP of the peer in the DNS. The host then sends an initial I1 message requesting a state to be established with the peer. Messages R1, I2 and R2 are exchanged successively in order to create an association.

Once the base exchange is completed, the data traffic between the communicating hosts is protected using IPsec. When one of the hosts changes its IP address, the new address needs to be updated with the peer. For this purpose, HIP uses a readdressing procedure. Additionally, readdressing can be accompanied with a new SPI value and/or new keys for the existing security association.

All packets except the base exchange and readdressing messages are protected using IPsec ESP. IPsec has traditionally been known to be a Network Address Translation (NAT) sensitive protocol. To allow IPsec protected traffic to traverse a NAT, it is either possible to provide UDP encapsulation [5] or to allow the NAT to participate in the signaling message exchange. A mechanism to detect a NAT along the path between two IPsec endpoints has be provided for IKEv1 [4] and has been incorporated into IKEv2 [6]. Additionally, firewall traversal faces routing asymmetry problems. A number of IETF working groups such as the MIDCOM, PANA and NSIS [13] have encountered this problem.

2 Problem Statement

Most networks today still use IPv4 addresses even though IPv6 is ready for deployment. Apart from the communicating end hosts, many middleboxes are also present between the hosts on the network, each meant for a specific functionality. For instance, to combat the IPv4 address depletion problem, private networks use NATs [12] to reuse and share global IPv4 addresses. For security reasons, firewalls are placed at the border of a network. When HIP is deployed into an existing network, NATs need to be retained for the sake of already existing IPv4 applications. For security reasons, HIP will need to deal with firewalls as well.

In the current Internet, IP addresses are used both for identifying hosts and identifying their topological locations. This semantic overloading is deeply related to most of well-known NAT problems [9]. IPsec is an example of a protocol that suffers from the related NAT traversal problems [10]. UDP encapsulation of IPsec packets allows a NAPT[12] to modify the UDP header and to perform

the demultiplexing [4]. Unfortunately, the approach unnecessarily increases the packet size and may cause configuration difficulties, e.g., in firewalls.

In this proposal we try to address the following functionalities that are expected of a HIP aware NAT or firewall:

1. Interception : IPsec use <Destination IP, Destination SPI, Protocol >to identify a particular security association. Middleboxes can also be thought to use the same flow identifier information for a flow. This can be achieved by making the NAT/FW HIP aware and to intercept the SPI values carried within HIP signaling messages.
2. Authentication : Many middlebox traversal mechanisms do not have any security at all. A HIP aware NAT/FW must be able to authenticate the requesting HIP nodes before creating a NAT binding or a firewall pinhole.
3. Authorization : A HIP aware NAT/FW must be able to authorize the requesting HIP nodes using identity dependent or identity independent methods. A potential solution must respect the property of the middleboxes before roaming outside the network.
4. Denial of Service attack resistance : The authentication and authorization mechanisms should not introduce new DoS attacks at the middlebox.
5. Registration Procedure - A firewall might require authentication and authorization of one of the end points prior to allowing signaling (and data traffic) to bypass. Depending on the architecture and environment, this protocol step might be required.
6. Avoiding unwanted traffic : In the wireless environment an end host might want to stop receiving unwanted traffic. A signaling protocol is needed to indicate what traffic to receive and what traffic to drop. It must also be assumed that end-to-end communication is not always possible prior to the interaction of the end hosts.
7. Soft-state Nature : To deal with failures and route changes, it is important to design a protocol in such a way that the state allocated at middleboxes times out after a certain period of time. Periodic transmission of refresh messages is therefore required. SPI multiplexed NAT (SPINAT) is an example of a HIP aware NAT that uses HIP to establish a NAT binding and to establish the security state [7].

3 HIP and NAT/FW Traversal

This section describes our proposal for traversing middleboxes with HIP. We use HIP as a protocol to communicate with middleboxes.

3.1 HIP Base Exchange and NAT

A HIP aware NAT/FW needs to inspect the HIP base exchange to learn the <Destination IP, Destination SPI, Protocol>triplet for a specific host. The HIT values are also required and can subsequently be used to verify future signaling

messages. The approach presented in [7] is also relevant here which requires the usage of hash chains to update the binding in a HIP aware NAT device. All HIP messages carry a standard HIP header with the HIT of the initiator and the HIT of the receiver. It must be noted that IPsec SAs are unidirectional and hence two SPI values (for the Initiator and for the Responder) need to be negotiated. Subsequently, message I2 carries the SPI value of the Initiator, SPI(I), and message R2 carries the SPI value of the Responder, SPI(R). For authorization, SPKI certificates [2] or SAML assertions [3] may turn out to be useful since the Host Identities might be ephemeral and anonymity for the end hosts is an important aspect. Providing authorization based on information in the SPKI certificates or SAML assertions can be used to enable the middlebox to execute the necessary protocol actions (e.g., opening a pinhole) without the need for authentication.

3.2 HIP Base Exchange and Firewalls

NATs establish state and modify IP address information and thereby force IP packets to flow through also in the reverse direction. This makes the interception mechanism for NATs much easier compared to that of the firewalls. In the presence of a generic middlebox (or firewalls in particular) or a topology with a mixture of NATs and firewalls, routing asymmetry needs to be considered. Figure 1 shows a HIP exchange through a firewall. In firewalls, forward paths may differ from the reverse paths. Then, messages I1 and I2 from the initiator to the receiver take a different path from messages R1 and R2 sent from the receiver to the initiator. For instance, the Initiator generates its SPI(I) and sends it to the Responder in a message I2 through FW(R). However, FW(I) needs this information to create the state for the Initiator. Similarly, the Responder generates its SPI(R) and sends it to I in the R2 message through FW(I). However, FW(R) needs to create the flow identifier information for R as shown in Figure 1.

Hence, new solutions need to be provided for tackling the routing asymmetry problem with respect to the firewalls and flow identifier interception. These solutions have to be handled without changing the existing HIP base exchange significantly.

3.3 HIP Readdressing, Re-keying and NAT/FW

Even after the HIP base exchange is finished, a NAT/FW still needs to keep updating its state for the flow identifier in case an IP address or an SPI value changes for an end host. For example, whenever a HIP end point is mobile and informs its peer about the new IP address, the states at FW(I) and FW(R) also need to be updated. Additionally, if the hosts decide to choose a new SPI value for the same security association or a new pair of keys along with the readdressing, routing asymmetry may cause additional complications. Middleboxes must authorize state modifications to avoid a number of attacks including redirection, black holing or third party flooding. A desired property in this case is sender invariance, which states: “A party is assured that the source of the communication

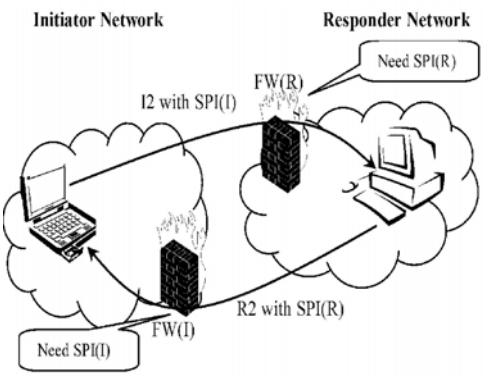


Fig. 1. Routing asymmetry with firewalls.

has remained the same as the one that started the communication, although the actual identity of the source is not important to the recipient."(Section 3 of [8]).

4 HIP Aware NAT/FW

Many middleboxes today do not support any security. State is created based on data traffic without authentication, authorization or DoS protection. The complexity to support different types of NAT/FWs influences the design of the protocol to a certain extent. The middlebox could fall into some of the following categories:

1. A NAT/FW could support only the present Internet Protocol and can be completely incompatible with HIP. These falls into the category of "HIP-unaware NAT/FW "that does not require security capabilities.
2. A "Transparent NAT/FW "could need weak authentication techniques security for simple state establishment, for instance, using the SPINAT functionality. However, here the base exchange becomes vulnerable to a DoS attack because the initiator's HI is encrypted in the I2 packet and the NAT/FW box is unable to verify the I2 message. As a consequence, an attacker may send a spoofed I2 message before the authentic initiator does that. The spoofed I2 message may contain a spoofed SPI value resulting in an inconsistent state at NAT/FW. The problem can be solved, either by including the initiator's SPI value both to the I1 and I2 messages or sending the initiator's HI as plain text in I2 packet. While the former solution creates a state at the NAT/FW and the peer host even before the puzzle is solved, the later interferes with anonymity. Fortunately, the NAT/FW may verify the responder's SPI in R2 packet with signature, because responder's HI is sent in plain text.
3. A third set of NAT/FW may opt to complete authentication and authorization before establishing state for a host. These are the "Registration

Requiring NAT/FW "that run a registration protocol, a variant of the HIP base exchange between the end host and the middlebox.

4.1 The HIP Registration Protocol

To introduce a new registration protocol, it is necessary to deal with the general protocol design issues such as mutual authentication capability, Denial of Service attack resistance and efficiency in the number of roundtrips. Furthermore, it is helpful if the end-to-end protocol and the registration protocol support the same credentials. These requirements motivate to reuse the HIP protocol for the purpose of authentication, authorization and the establishment of a security association. However, it should be noted that the establishment of an IPsec security association is not necessary here.

To deal with mobility it is necessary to periodically refresh the state at the firewall. The update of packet filters can either be sent directly to the firewall or indirectly with the help of an end-to-end HIP exchange. The former might be necessary for a data receiver installing packet filters to prevent unwanted traffic from consuming an expensive wireless resource where the data receiver might get charged for.

Factors giving an advantage to the HIP registration protocol are follows:

1. Reuses the same puzzle mechanism to prevent Denial of Service attacks.
2. The Initiator has to solve the puzzle in order to prove its interest in a successful protocol exchange. This allows the Responder to delay state creation until receiving I2. The puzzle is made up of the corresponding HITs and a random number; the difficulty of the puzzle can be increased based on the trust of the Initiator. This cookie mechanism prevents the Responder from some Denial of Service attacks.
3. Provides an end-to-end authentication, using signature verifications.
4. Both the Initiator and the Responder can authenticate each other; Initiator authenticates Responder in the R1 packet by verifying the signature using HI(R) and the Responder authenticates the Initiator by verifying the signature of the I2 packet using HI (I).
5. Uses HMAC to protect the integrity of the messages and prevents DoS using signature verifications.
6. Both the peers obtain the shared secret key and calculate the corresponding derived keys using the authenticated Diffie-Hellmann exchange. Responder uses one of the keys to calculate HMAC in the R2 packet in order to prove the key confirmation.
7. Uses SPKI certificates (or SAML assertions) for authorization.
8. The Initiator may send the authorization certificate immediately after the I2 message, to be authorized by the middlebox. This is a significant improvement in design of the middleboxes, as currently most middleboxes do not provide authorization.

4.2 SPISIG Message

The generic registration protocol that we have introduced can be used for all middleboxes that require authentication and authorization for a host-middlebox binding. This is mostly the case for NATs and firewalls at network borders for outgoing traffic. However, the firewall for the incoming traffic needs to maintain state information for the host to forward its packets. The registration protocol can be reused here between the incoming traffic firewall and the host to make sure that the firewall maintains the proper state for the legitimate host. Even after the registration, the state is still not complete as FW(R) is unable to intercept SPI(R) sent in R2 and FW(I) is unable to intercept SPI(I) sent in I2 as was shown in Figure 1.

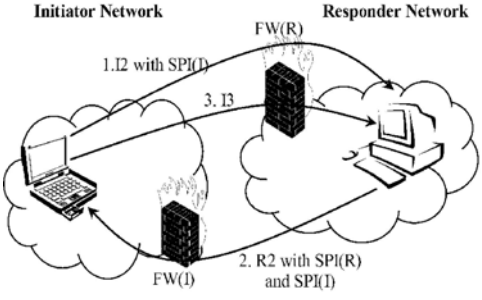


Fig. 2. Extending the base exchange with I3.

One possible solution to this problem could be following. Once the Responder receives the SPI (I) in message I2, it could resend the SPI (I) along with SPI(R) in message R2. This could help the FW (I) intercept the SPI (I) information. Since the receiver R has to remain stateless until the solution in I2 is verified, the SPI(R) cannot be sent in R1 and hence not resent in I2. The only other option would be to create a new message I3 as that carries the SPI(R) from the Initiator to the Responder such that all middleboxes in the path can intercept and form the flow identifier information for the receiver. However, such a solution of changing the base exchange messages for the sake of firewall traversal is unsatisfactory and undesired.

$I \rightarrow FW(I) \rightarrow R : I1 \subset Trigger\ exchange$
$I \leftarrow FW(R) \leftarrow R : R1 \subset$
$Puzzle, \{DH(R), HI(R), HIP_{Transform}, ESP_{Transforms}\}SIG$
$I \rightarrow FW(I) \rightarrow R : I2 \subset$
$\{Solution, SPI(I), DH(I), HIP_{Transform}, ESP_{Transform}, \{H(I)\}\}SIG$
$I \leftarrow FW(R) \leftarrow R : R2 \subset \{SPI(R), SPI(I), HMAC\}SIG$
$I \rightarrow FW(I) \rightarrow R : I3 \subset \{SPI(R), HMAC\}SIG$

An alternative solution could be that once the base exchange is complete and a state is established at the communicating HIP hosts, the local host could signal its firewall in a SPISIG message about the SPI value that it has chosen for the particular security association. The firewall would have already intercepted the IP and HIT values from the initial messages of the base exchange. It can then create the flow identifier information using the SPI value that it obtains from the hosts within the private network.

5 Formal Analysis

The protocol has been analyzed by means of formal method analysis using the High Level Protocol Specification Language (HLPSL) - an expressive language for modeling communication and security protocols.

We used the tool OFMC[1] (On-the-Fly Model-Checker), from the AVISPA project [15] (Automated Validation of Internet Security Protocols and Applications"), which uses a rich specification language for formalizing protocols, security goals, and threat models of industrial complexity.

The HIPSL file was then translated into an Intermediate format using another tool named HLPSL2IF, which is a translator, which maps security protocol specifications into rewriting systems. This intermediate format can be executed to analyze the threats of the protocol. From the results, which we got, no attacks were found for the following attacks:

- Man in the Middle Attack (MitM)
- Denial of Service attack (DoS)
- Replay Attack
- Server Authentication to the client (server spoofing)
- Client Authentication to the server (client spoofing)

5.1 Informal Analysis

The HIP registration protocol uses an authenticated Diffie-Hellmann Key Exchange and generates session keys to defend against the Man-in-the-Middle attacks. The Initiator provides key confirmation in the I2 packet by encrypting the Host Identity and the Responder performs key confirmation by sending the HMAC in the R2 packet. When the Initiator chooses anonymous HIs, the protocol suffers from the Man-in-the-Middle attacks. The usage of authorization certificates provides a solution for this purpose but the formal modeling tool will produce an error.

This protocol also provides some protection for the Initiator, since the messages from the Responder are signed. One potential problem could be the following case: R1 message contains the signature and the Initiator, first, has to verify it. Here the Intruder might try some DoS attacks. But in order to launch this attack the Intruder has to act as a Man-in-the-Middle adversary and act quickly to send spoofed R1 packets.

[1] The tool is available on-line at http://www.avispa-project.org/web-interface/

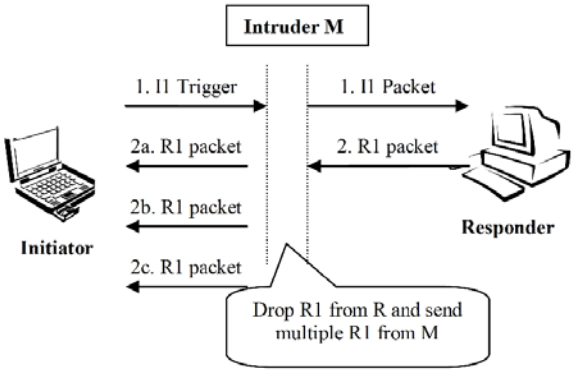


Fig. 3. Sending multiple bogus R1 packets.

After the I2 message, the Responder may wait for the certificates. Here the Intruder can send some bogus certificates with signatures and forcing the Responder to verify, this might cause DoS attacks. This kind of attack can be resisted, if the responder is designed not to accept more than one certificate during the base exchange.

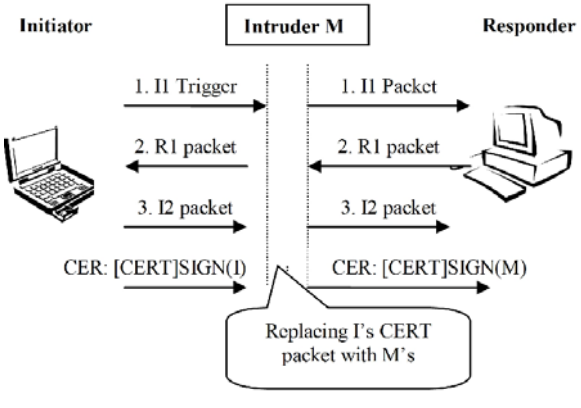


Fig. 4. Sending bogus certificates.

This protocol uses an R1 counter to protect against replay attacks. The R1 generation counter is a monotonically increasing 64-bit value and this counter indicates the current generation of puzzles. The system can avoid replay attacks by simply increasing the value of the counter to show the validity of the packet.

The Initiator can check the counter to determine whether it received a new high counter value or not. The server authenticates the client in the R1 packet by sending his HI in clear text and also signs the message. The Initiator can verify the HI and signature as it knows the Responder's public key/HI from the DNS look up.

Client authentication to the server can be done because the server verifies the Initiator's Public key/HI with the received HIT. Since HIT is the Hash of the HI, after the receiving the I2 packet, the Responder can verify the Initiator's identity by cross checking the HIT and HI.

Thus, the registration protocol provides enough resistance to protect against the above listed attacks.

5.2 Implementation

We have implemented a prototype for the registration protocol [5]. For simplicity the current implementation assumes that the Initiator obtained the SPKI certificate using an out-of-band mechanism [6].

We found out that the minimum memory needed for storing the state information at a middle-box is 2286 bytes. The approximate time taken for each packet is summarized in table below. Computing the Diffie-Hellman derived session key takes almost 80% of the time and signature verification takes 10% of the time.

Packets	I1 (ms)	R1 (ms)	I2 (ms)	R2 (ms)
Creation	0.030	15	95	25
Processing	0.007	300	75	15

Table 1. Time taken for the packets.

A more detailed performance investigation is in progress. To establish an arbitrary number of HIP sessions and to check the throughput and packet loss requires some protocol enhancements. The current implementation establishes a state, if there is a change in the IP address or in the HIT. Changing the IP address or HIT for high-performance tests does not seem to be adequate. seems really difficult in a short interval of time. A different session identification (added for testing purpose to the HIP registration protocol) allows creating an arbitrary number of concurrent exchanges.

[5] We used two Pentium II 266 Mhz Linux based machines as an Initiator and the Responder (Middlebox), both of them residing in a single LAN.

[6] The throughput between the Initiator and Responder, (measured by using ttcp) was 8.5 Mbps, the round trip time was 0.16 ms (measured with ping) and the average time taken to complete the registration was approximately 0.94 seconds.

6 Conclusions

For a long time the focus of HIP was on solving problems affecting mainly the endpoints. In future, the IETF HIP research group [17] will also address the middlebox traversal problem for HIP. To avoid including a HIT into every data packet and to provide end-to-end protection of data traffic, IPsec ESP is used between the end points. Unfortunately, IPsec protected data traffic is known to cause problems with middleboxes (particularly with regard to NAT traversal). Middleboxes need to participate in the HIP signaling exchange to allow these devices to perform their function. This interaction requires certain security goals to be met. A solution can be complicated by a number of factors including routing asymmetry, combination of different types of middleboxes and state updates due to mobility. Our proposal tries to raise the attention of the community based on a simple protocol proposal.

To enable HIP-aware middleboxes, we use a registration procedure. The registration procedure reuses the common base exchange mechanism, removing the ESP transforms and SPI fields. Authorization functionality is added using SPKI certificates or SAML assertions. It is a first step toward deployment of HIP friendly NATs and firewalls that performs their functionality with enhanced security.

7 Acknowledgements

This document is a by-product of the Ambient Networks Project, partially funded by the European Commission under its Sixth Framework Programme. It is provided “as is ”and without any express or implied warranties, including, without limitation, the implied warranties of fitness for a particular purpose. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Ambient Networks Project or the European Commission.

References

1. Moskowitz R., Nikander P., Jokela P. and Henderson T., Host Identity Protocol draft-ietf-hip-base-01.txt (work in progress), October 2004.
2. Ellison C., Frantz B., Lampson B., Rivest R., Thomas B. and Ylnen T., SPKI Certificate Theory . RFC 2693, September 1999.
3. Maler, E., Philpott R., and Mishra P., Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V1.1 , September 2003.
4. Kivinen, T., Swander, B., Huttunen, A. and Volpe, V., Negotiation of NAT-Traversal in the IKE, RFC 3947, January 2005.
5. Huttunen A., Swander, B., Volpe, V., DiBurro, L. and Stenberg M., UDP Encapsulation of IPsec ESP Packets . RFC 3948, January 2005.
6. Kaufman, C., Internet Key Exchange (IKEv2) Protocol. draft-ietf-ipsec-ikev2-17.txt (work in progress), September 2004.

7. Ylitalo, J., Melen, J., Nikander, P. and V. Torvinen Re-thinking Security in IP based Micro-Mobility 7th Information Security Conference (ISC-04), Palo Alto, September 2004.
8. Automated Validation of Internet Security Protocols and Applications (AVISPA) IST-2001-39252, Deliverable v1.0, November, 2003.
9. Moore K., Things that NATs break Unpublished, http://www.cs.utk.edu/ moore/what-nats-break.html, October. 2003.
10. Aboba B. and Dixon W., IPsec-Network Address Translation (NAT) Compatibility Requirements RFC 3715, March 2004.
11. Ylitalo, J., P. Jokela, J. Wall and P. Nikander., End-point Identifiers in Secure Multi- Homed Mobility in Proc. of the 6th International Conference On Principles Of DIstributed Systems (OPODIS 02), pp. 17-28, France, Dec., 2002.
12. Giving K. and Francis P., Network Address Translator RFC 1631, May 1994.
13. Next Steps in Signaling (nsis) Working Group Charter http://www.ietf.org/html.charters/ nsis-charter.html (February 2005).
14. Kent S. and Atkinson R., IP Encapsulating Security Payload, RFC2406, November 1998.
15. Automated Validation of Internet Security Protocols and Applications Webpage, http://www.avispa-project.org/, (February 2005).
16. Kent, S. and Seo K., Security Architecture for the Internet Protocol, draft-ietf-ipsec-rfc2401bis-05.txt, (work in progress), December 2004.
17. Host Identity Protocol (HIP) IRTF Research Group, http://www.irtf.org/charters/hip.html (February 2005)
18. Jokela P, Moskowitz R, Nikander P, Using ESP format with HIP draft-jokela-hip-esp-00.txt (work in progress), Febrauary 2005.

An Investigation of Unauthorised Use of Wireless Networks in Adelaide, South Australia

Phillip Pudney & Jill Slay

School of Computer and Information Science
Advanced Computing Research Centre
University of South Australia
Mawson Lakes, South Australia
phillip@pudney.net.au & Jill.Slay@unisa.edu.au

Abstract. While it is known that wireless networks experience unauthorised connections, little is known about the nature or frequency of the connections. This study seeks to investigate the unauthorised use of wireless networks, and to dispel the myth that attacks on wireless networks are simply an attempt to obtain Internet access. Three wireless honeypots were deployed to collect data about unauthorised use of wireless networks in the Adelaide CBD. The data collected from the honeypots was then analysed for trends and evidence of malicious activity. The results of the study show that insecure wireless networks regularly experience unauthorised activity, ranging from harmless probes through to intrusion attempts.

1. Introduction

Wireless network technology has enabled true mobile computing, by allowing a remote user to connect to a corporate computer network from anywhere within the coverage area of the wireless network. It is convenient, easy to use, inexpensive and throughput is now comparable to wired networks, and hence wireless networks have become an attractive option to corporate users and consumers alike. Accordingly, the growth in wireless hardware sales has sharply increased in the past few years as it is being used as both to complement and to replace wired networks. However, as wireless networks become ubiquitous, concerns about their security are escalating.

This paper discusses how honeypots can be used to investigate unauthorised use of wireless networks based on the IEEE 802.11 standard. It describes the results of a study conducted by the authors, who deployed a series of wireless honeypots to ascertain the extent of attacks on wireless networks in Adelaide.

To contextualise the problem, the next sections briefly discuss wireless networks and the security of wireless networks. Next, the concept of honeypots are introduced, including how they can be applied to wireless technology, and what research is currently being undertaking using the technology. The details and results of a study conducted by the researchers are then discussed. Finally, the paper concludes with a discussion about the results and further research that can be conducted using wireless honeypots.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 29-39, 2005.

2. Wireless Network Insecurity

The IEEE 802.11 standard defines two modes of operation—ad-hoc mode, where communication occurs directly between wireless clients, or infrastructure mode, where all communication passes through an *access point* (AP) that often connects to a wired network. The advantage of this is that clients on the wireless network can communicate with clients on the wired network and vice-versa.

However unlike wired networks, wireless networks cannot be protected through physical security. Signals from wireless networks can pass through walls and physical obstacles, and can propagate beyond the physical confines of the environment in which the wireless network was intended to operate. Consequently, these signals can be intercepted by anyone within range. An infrastructure wireless network without security can potentially expose the wired network, making it a valuable target to attackers.

2.1 IEEE 802.11 Security Controls

The 802.11 designers recognised the inherent differences between the wired and wireless environments [7] and included the Wired Equivalent Privacy (WEP) protocol. WEP was designed to provide security equivalent to that of wired networks [5]. However numerous flaws in the WEP protocol have since been exposed. The result is that a WEP key can be recovered in as little as a few hours [7]. Several freely available tools have been released that automate recovery of WEP keys, including WEPCrack and AirSnort.

Authentication is provided by one of two mechanisms—open system authentication or shared key authentication. Open system authentication is essentially a null authentication process, granting access to any request to associate with the wireless network and provides no security at all. Shared-key authentication on the other hand uses a challenge and response along with a shared secret key to authenticate a client. However, the key is the same as that used by WEP, and the protocol is vulnerable to a passive attack [2]. Moreover, it is not capable of differentiating between individual users, and it does not protect against rogue access points since authentication is only performed in one direction and is not mutual.

Apart from the security controls included in the 802.11 standard, some hardware vendors have implemented other security mechanisms, including MAC-based access control lists, ‘closed system’ authentication, and security that operates at higher network layers such as virtual private networks (VPNs). However, few of these solve the current security problems wireless networks.

Last year, the IEEE approved 802.11i to address the embarrassing security flaws in the original 802.11 standard. 802.11i defines two mechanisms, Temporal Key Integrity Protocol (TKIP) and Counter mode with CBC-MAC Protocol (CCMP). TKIP is an immediate replacement for WEP and is designed to run on existing hardware. CCMP is seen as the long-term solution for wireless LAN security. It adds the Advanced Encryption Scheme (AES) and bears little resemblance to WEP.

2.2 Trends in Wireless Security

Despite the flaws in the original 802.11 security controls, they still function as a deterrent against casual network attacks. However, many organisations who use wireless network technology are not even implementing rudimentary security. In a study conducted of wireless networks in Australian CBDs, Hannan & Turnbull [4] discovered that of 729 operational wireless networks detected, only half implement 802.11's WEP, while at least 15% had failed to implement any security at all.

There are few reports of wireless networks being exploited, even though the weaknesses have been well publicised [1]. However 'war driving', the practice of driving around looking for wireless networks [10], is well documented in the literature. A search on the subject using popular Internet search engine Google returns about quarter of a million results!

Despite what is understood about the weaknesses and attacks of wireless networks, only a modest amount of research has focused on determining the extent of attacks. Particulars about the frequency of attacks, the number of attackers, methods of intrusion, and the motivations of attacks remain uncertain.

There are several ways in which data can be collected to ascertain the extent of attacks on wireless networks. These include collating data from case studies of actual attacks, or using security tools like intrusion detection systems (IDS). However, a more suitable method is to use honeypots—a relatively new concept in security. Honeypots are systems that appear to be an interesting target to hackers, but actually gather data about them.

3. Honeypots

The concept was described in the literature as far back as 1990, in Stoll's book 'The Cuckoo's Egg' [13] that discusses a series of true events where an attacker who infiltrated a computer system was monitored. A paper by Cheswick [3] describes a system that was built to be compromised by an intruder, and to study what threatening activity was happening on the network. His paper documents the first case of a true honeypot [12]. However, freely available honeypot solutions did not exist until the Deception Toolkit was released in 1997. Since then, honeypots have evolved and are starting to be accepted as a legitimate network security tool.

Spitzner, moderator of the honeypots mailing list and one of the leading honeypot experts, defines a honeypot as "*an information system resource whose value lies in unauthorised or illicit use of that resource*". Essentially, it is a resource that has no production value on a network. Since it only imitates production resources and does not contain any critical data, it should see no network activity. Any interaction with a honeypot is therefore likely to be unauthorised, malicious activity, such as a probe or port scan [11].

This is a contrast from most security tools, which set out to address specific problems. For example, firewalls are deployed around a network perimeter to control connections into and out of the network. IDSs detect attacks by monitoring network activity [12]. An IDS inspects all frames on a production network, and compares them to a set of predefined rules. If a rule is matched, then an alert is triggered. IDSs do not

always accurately report an intrusion, and often generate masses amounts of false positives. Furthermore, for an IDS to recognise an attack, a signature must exist for the specific attack. If no signature exists, then an intrusion detection system has no way of triggering an alert. Since a honeypot should not see any network traffic, all connections are, in theory, unauthorised, and therefore data that is logged is almost certainly the result of a probe or an attack on the system.

A common misconception about honeypots is that their purpose is to lure attackers. However, since a honeypot has no legitimate business, an attacker would need to actively search for the resource and try to compromise it at their own will.

3.1 Types of Honeypots

Honeypots generally fall into one of two categories—production and research [11]. Production honeypots are systems that that help to secure networks. They can provide systems for prevention, detection and response to intrusions. Research honeypots, the focus of this paper, contribute little to the direct security of a network. Instead, they offer value through their information gathering capabilities. They are concerned with providing a platform to study threats. This is what makes honeypots a useful method for gaining intelligence about attacks on wireless networks.

Honeypots are often characterised by the level of interaction they afford to attackers. Low-interaction honeypots work by emulating operating systems and the services that run on the system. They expect a specific type of behaviour, and are programmed to react in a predetermined way.

High-interaction honeypots on the other hand do not simulate an environment, but instead involve real operating systems and applications. They make no assumptions about how the attacker will behave, allowing a researcher to learn the full extent of their behaviour since they make no assumptions about how the attacker will interact with the system where all activity is captured.

Clearly, high-interaction honeypots are the most desirable for researching attackers since they are capable of providing a vast amount of information. However since the attacker has access to a real operating system, the level of risk is much greater than that of a low-interaction honeypot. There is more that can go wrong, and there is a possibility that the attacker can use the honeypot to attack other systems. Consequently, the researchers chose to use a low-interaction honeypot for the study, since it was concerned with the bigger picture, and not how an attacker interacts with an individual system.

4. Previous Research Using Wireless Honeypots

There appears to be very little data about the extent and nature of attacks on wireless networks, apart from a handful of case studies where attackers have been caught. That's not to say that organisations are not collecting data for their own use—but it is not being published in the literature.

Four studies in particular have provided researchers with some insight into unauthorised use of wireless networks. However, none provide substantial detail about the extent of unauthorised use.

In the middle of 2002, the first organised wireless honeypot was documented. Researchers at the US government contractor Science Applications International Corporation (SAIC) set up the Wireless Internet Security Experiment (WISE) to 'develop effective information security, intrusion detection, and incident response, and forensic methodologies for wireless networks' according to the project's web page [6]. The page appears to have since been taken offline. In an article by Poulsen [9], the network was reported to consist of five access points and a handful of deliberately vulnerable computers as bait at a secret location in Washington D.C. After details about WISE were leaked in 2002, no literature or results of the experiment have appeared since.

Not long after, a similar study was conducted by Tenebris Technologies Inc. over a three month period in Ottawa, Canada [14]. The study consisted of a single wireless honeypot located in an undisclosed area outside of the area's central business district. During the three months that the wireless honeypot was active, it logged 40 wireless associations on 10 different days. One of the associations was reported to have clearly been an intrusion with criminal intent. During the intrusion, the attacker spent 20 minutes attempting to access various web sites, performing port scans and trying to connect to particular services on the honeypot network. A port scan is often used for reconnaissance by would-be intruders to determine what hosts exist on a network, and what services they are running. Given enough information, the hacker could determine vulnerabilities in the hosts on the network and exploit them. This figure would have undoubtedly been much higher had the honeypot been in the central business district, where the concentration of wireless networks is much higher and therefore more attractive to an attacker.

In June 2003, Turnbull, Nicholson & Slay conducted an experiment where they deployed a honeypot attached to a wireless network in the CBD of the City of Adelaide for two weeks [15]. During the two week period that the honeypot was capturing data, two attempts were made to connect to the honeypot. One of the attempts was possibly the result an accidental connection. The second attempt, where the intruder initiated a port scan, was more likely to be targeted with malicious intent.

It is evident from previous studies that wireless networks are being attacked. However, none involve the widespread deployment of honeypots to ascertain the full extent of unauthorised use.

5. Experiment Details

The study was conducted by deploying wireless honeypots around the Adelaide CBD between October 2004 and January 2005.

For this experiment, the researchers decided to place honeypots at three different locations. The rationale was that having honeypots at different locations allowed a much greater area to be studied. Furthermore, the results from each of the locations could be compared to determine if the same client interacted with more than one wire-

less honeypot, and to ascertain any trends based on the surrounding environment. Fig. 1 shows the location of each of the wireless honeypots.

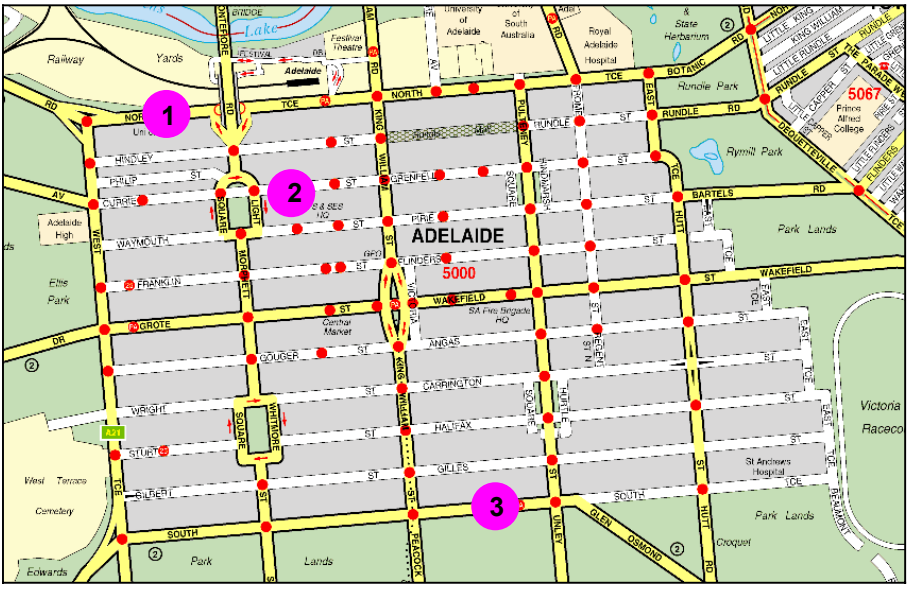


Fig. 1. Map showing the location of the three wireless honeypots

Site 1 was located on the fourth floor of a building at the University of South Australia's City West Campus on North Terrace. Site 2 was placed on the third floor of building on Currie Street. The area immediately surrounding it is densely populated with office buildings, and the access point was placed in the corner of the building with an unobstructed view for many kilometres to the west, and views to adjacent streets. Site 3 was located on the ground floor of a school building on South Terrace. One side of South Terrace is occupied by parklands. The other side is a combination of residential units, business offices and motels, with ample roadside parking available during the daytime and evening.

At each site, a survey was conducted to establish the number of nearby wireless networks. At both site 1 and site 2, three other wireless networks were detected, while no wireless networks were detected at site 3.

Each wireless honeypot consisted of the honeypot itself, which was attached to a wireless access point (refer to Fig. 2). To ensure minimal exposure to risk, the wireless honeypot was isolated from any network, and was not connected to the Internet.

A Netgear WG602 access point with a 2dBi antenna was used at each site. The access point was configured to broadcast its SSID, and no security was configured to simulate an out-of-box wireless network installation.

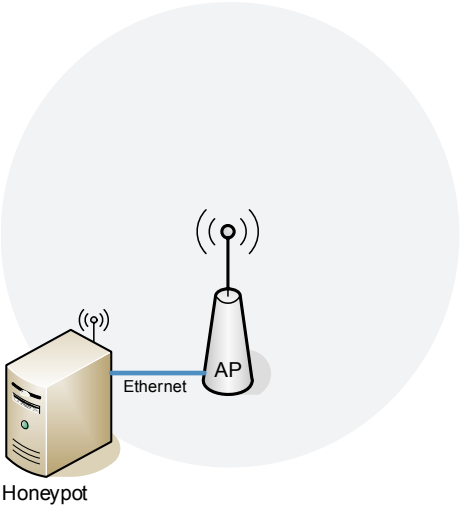


Fig. 2. Honeypot design

Honeyd was installed on each honeypot. Honeyd is an open-source, low-interaction honeypot daemon developed by Niels Provos from the University of Michigan. It creates virtual hosts on a network that emulate numerous operating systems. For each virtual host, an arbitrary number of services can be configured. Service scripts, which provide an environment through which an attacker interacts with applications, can be customised or written from the ground up in almost any language supported by the host operating system. Furthermore, multiple virtual hosts can listen for activity simultaneously, effectively allowing Honeyd to simulate an entire network running a variety of different operating systems with different applications. A DHCP daemon was installed on the honeypot to automatically issue IP addresses to wireless clients that connected to it.

A wireless network card was also installed in each honeypot. This was used to capture all network traffic that was transmitted to or from the access point. The capture dumps were later analysed to determine the number of unauthorised connections and evidence of malicious activity.

6. Results

To discover and connect to wireless networks, clients transmit probe requests to which an active wireless network responds. Many war driving tools also transmit probes in an attempt to locate insecure wireless networks. The responses to these probe requests were captured together with the MAC address of the client to determine how many unique clients interacted with the access point connected to each of the honeypots. A summary of the number of unique clients seen is shown in Table 1.

Table 1. Number of unique clients that interacted with each wireless honeypot

Honeypot	Number of unique clients	Period (days)	Average interactions per day
Site 1	400	88	4.5
Site 2	562	89	6.3
Site 3	40	64	0.6

While these figures do not suggest malicious activity, they do show that there are a high number of active wireless network devices, each capable of making an unauthorised connection or participating in malicious activity. They also suggest that the likelihood of interaction is much higher where other wireless networks are detected nearby, compared to when there are none.

Of those clients who interacted with the wireless honeypots, a list of those that connected to the honeypot was also generated (refer to Table 2).

Table 2. Unauthorised connections experienced by each wireless honeypot

Honeypot	Unauthorised connections	Average connections per week	% of interactions
Site 1	26	1.0	6.5
Site 2	102	10.6	18.1
Site 3	6	0.7	15.0

All of the three wireless honeypots experienced unauthorised connections, and between 6.5% and 18.1% of wireless clients that interacted with the access point connected without authorisation. More significantly, up to 10.6 unauthorised connections (on average) were counted by the honeypot immersed by surrounding wireless networks.

The results show that an insecure wireless network will inevitably experience unauthorised connections, and that a wireless network near other active wireless networks is at a much higher risk. Each unauthorised connection was further analysed to ascertain the nature of the connection.

The analysis showed that the majority of unauthorised connections attempted to access Internet hosts, evident by attempted DNS queries originating from the connecting clients. In most cases, DNS queries were for:

- popular web sites, such as www.google.com or www.yahoo.com;
- instant messaging (IM) applications;
- software update sites—predominantly Windows Update, although several other sites for security software updates were also present;
- peer-to-peer download sites, such as those using the BitTorrent protocol; and
- e-mail servers.

Each unauthorised connection to the wireless honeypots was investigated for evidence of intrusion and malicious activity. Activities such as port scans, attempts to penetrate the wireless honeypot's virtual hosts, or other unusual behaviour were identified. Identification was undertaken by scrutinising the type of and sequence of inter-

actions in the Honeyd log files and capture dumps for each unauthorised connection. The results of the investigation are shown in Table 3.

Table 3. Malicious connections experienced by each honeypot

Honeypot	Malicious connections	% of unauthorised connections
Site 1	0	0
Site 2	3	2.9
Site 3	2	33.3

In total, five malicious connections were identified, with two out of three honeypots experiencing an attack. This clearly indicates that the risk of an insecure wireless networks suffering an intrusion is high and cause for concern. Furthermore, the results show a trend where the risk of connections that result in an intrusion is inversely proportional to the number of nearby wireless networks.

7. Discussion

The fact that the wireless honeypots were not connected to the Internet limited conclusions that could be drawn from the connections. However, the researchers believe that most connections were not malicious, and were the cause of innocent users accidentally connecting to the access point (and in many cases, without knowing that they had done so). This agrees with a survey conducted mid-2002 [8], where the greatest percentage of wireless security incidents reported by participating organisations was users connecting to wrong access points.

Nevertheless, many business-class Internet connections include download caps that limit the amount of data that can be downloaded over the connection per month. Data usage in excess of these caps often incurs excess usage charges. Even if an unauthorised connection is not intended to be malicious, this study confirms that most unauthorised connections lead to theft of bandwidth, potentially costing organisations with insecure wireless networks charges for excess data.

The problem is partly because some wireless software can be set to automatically connect to any available wireless network. As soon as a wireless client is within range of an access point, it automatically associates with it. Consequently, clients connecting to wrong access points may suffer disruptions to their network and Internet connectivity. Furthermore, users often do not use a host-based firewall to control access through their wireless network interface since they treat it as ‘trusted’. Each time they connect to an un-trusted wireless network, they risk intrusion from black hats connected to the same network, or infection from worms and other malware. These could be spread to their own network when the infected client connects.

Clearly, the number of unauthorised connections experienced by the honeypots in this study shows that operators of wireless networks need to implement security, even if it is the minimal WEP. In three months, two out of the three honeypots experienced multiple intrusions with malicious intent that could have resulted in loss or theft of valuable data, external attacks, or worse. Organisations also need to develop security

policies to prohibit wireless clients from connecting to any available wireless networks.

8. Further Research

The increase in the number of wireless "hot spots" has created unique security problems. Hot spots enable wireless Internet access from areas outside of the corporate wireless network, such as from parks, coffee shops and hotels.

Authentication to a hot spot provider typically occurs at a gateway between the wireless network and the provider's Internet connection rather than at the access point itself. However, few hot spot providers support encryption. Since authentication is not performed at the access point, an attacker can associate with the hot spot network and intercept the unencrypted traffic without the need for a network account. Furthermore, systems that do not have a firewall installed to protect the wireless interface may be vulnerable to intrusion by attackers via the hot spot wireless network.

Hot spot problems may be merely academic since no literature is currently available to prove or disprove the theory. Research wireless honeypots could be deployed within hot spot networks to survey whether or not the problem exists, and production honeypots could be used to deter would-be attackers or to monitor and deny access to them.

Honeypots could also be used in emerging wireless technologies such as Bluetooth and 3G mobile networks to provide a broader representation of wireless crime.

9. Conclusion

A number of weaknesses in 802.11 wireless network security allow intruders to intercept confidential data and illegally connect to networks, while some implement no security at all.

The results of the study showed that over a period of 89 days, up to 562 unique computers interacted with the access point connected to one of the honeypots. Between 6.5 and 18.1% of computers established an unauthorised connection. In general, an unauthorised connection resulted in Internet traffic and bandwidth theft. In most cases, unauthorised connections appeared to be accidental. However two out of three wireless honeypots experienced intrusions with malicious intent.

It is imperative that steps are taken to secure wireless networks. Furthermore, security policies need to be developed to ensure that wireless clients configured so that they do not automatically connect to un-trusted wireless networks.

References

1. Arbaugh, WA 2003, 'Wireless security is different', *Computer,* vol. 36, no. 8, August 2003, pp. 99-101.

2. Arbaugh, WA, Shankar, N & Wan, YCJ 2002, 'Your 802.11 wireless network has no clothes', *IEEE Wireless Communications*, December 2002, pp. 44-51.
3. Cheswick, B 1990, 'An evening with Berferd in which a cracker is lured, endured, and studied', in *Proceedings of USENIX*, January 20 1990.
4. Hannan, M & Turnbull, B 2004, 'Wireless network security in Australia: A study of 7 Australian capital cities', in *Proceedings of PACIS 2004*, Shanghai, China.
5. IEEE 1999, *ANSI/IEEE Std 802.11, 1999 Edition*, LAN MAN Standards Committee of the IEEE Computer Society, USA.
6. Lemos, R 2002, *Catching wireless hackers in the act*, updated 02/09/2002, ZDNet Australia, viewed 10/03/2004, <http://www.zdnet.com.au/news/security/0,2000061744,20267853,00.htm>.
7. Petroni, NL & Arbaugh, WA 2003, 'The dangers of mitigating security design flaws: a wireless case study', *IEEE Security & Privacy Magazine,* vol. 1, no. 1, pp. 28-36.
8. Phifer, L 2002, 'Understanding wireless LAN vulnerabilities', *Business Communications Review*, no. September 2003, September 2003, pp. 26-32.
9. Poulsen, K 2002, *Wi-Fi honeypots a new hacker trap*, updated 29/07/2002, SecurityFocus, viewed 23/02/2004, <http://www.securityfocus.com/news/552>.
10. Shipley, P 2001, *Open WLANs: the early results of wardriving*, viewed 25/08/2004, <http://www.dis.org/filez/openlans.pdf>.
11. Spitzner, L 2003a, *Honeypots: definitions and value of honeypots*, updated 29/04/2003, viewed 10/03/2004, <http://www.tracking-hackers.com/papers/honeypots.html>.
12. Spitzner, L 2003b, *Honeypots: tracking hackers*, Addison-Wesley, Boston.
13. Stoll, C 1989, The Cuckoo's Egg: tracking a spy through the maze of computer espionage, Pan Books, London.
14. Tenebris Technologies Inc. 2002, *Tenebris Wireless Honeypot Project: Assessing the threat against wireless access points*, updated 19/11/2002, Ottawa, Canada, viewed 4/08/2004, <http://www.tenebris.ca/docs/TWHP20021119.pdf>.
15. Turnbull, B, Nicholson, D & Slay, J 2003, 'Wireless Network Security - A Practical Summary of 802.11b in Adelaide, Australia', in *Proceedings of 4th Australian Information Warfare & IT Security Conference*, Adelaide, Australia, 20-21 November 2003.

An Efficient Solution to the ARP Cache Poisoning Problem

Vipul Goyal and Rohit Tripathy

OSP Global, Town Center, Andheri(E),
Mumbai, India
{vgoyal, rtripathy}@ospglobal.com

Abstract. ARP cache poisoning is a long standing problem which is known to be difficult to solve without compromising efficiency. The cause of this problem is the absence of authentication of the mapping between IP addresses and MAC addresses. Due to lack of the required authentication, any host on the LAN can forge an ARP reply containing malicious IP to MAC address mapping causing ARP cache poisoning. In fact, there are a number of tools freely available on the internet using which, even a newbie can launch such an attack. In this paper, we present a new cryptographic technique to make ARP secure and provide protection against ARP cache poisoning. Our technique is based on the combination of digital signatures and one time passwords based on hash chains. This hybrid system prevents the ARP cache poisoning attack while maintaining a good system performance at the same time.

1 Introduction

Local Area Networks running TCP/IP over Ethernet are the most common networks these days. Each host on such a network is assigned an IP address (32 bits). Hosts also posses a network interface card (NIC) having a unique physical address (48 bits) also called the MAC address. For the final delivery of any packet destined to some host, its MAC must be known to the sender. Thus, the address resolution protocol is used to resolve an IP address into a MAC address. Resolved addresses are kept in a cache so as to avoid unnecessary work for already resolved addresses every time they are needed. Resolution is invoked only for entries expired or absent from the cache, otherwise cache entries are used.

The ARP Poisoning attack involves maliciously modifying the association between an IP address and its corresponding MAC address so as to receive the data intended to someone else (victim). By performing ARP poisoning, an attacker forces a host to send packets to a MAC address different from the intended destination, which may allow her to eavesdrop on the communication, modify its content (e.g., filtering it, injecting commands or malicious code) or hijack the connection. Furthermore, when performed on two different hosts at the same time, ARP poisoning enables an adversary to launch a “man in the middle” (MiM) attack. With MiM attacks, the traffic between two hosts is redirected through a third one, which acts as the man in the middle, without the two knowing it. The MiM may simply relay the traffic after in-

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 40-51, 2005.

specting it or modify it before resending it. Note that MiM attacks are also possible at various other network layers. ARP cache poisoning allows performing such an attack at data link layer.

ARP Cache poisoning can also be used to launch a Denial-of-Service (DoS) attack [20]. Furthermore, this attack is not just confined to Ethernet networks but layer 2 switched LANs and 802.11b networks are also vulnerable. In [2], various scenarios are described where a wireless attacker poisons two wired victims, a wireless victim and a wired one, or two wireless victims, either through different access points or a single one. Even a newbie can launch sophisticated poisoning attack using easily available tools and tutorials on internet, [5, 7, and 10].

In this paper, we propose a solution to the ARP cache poisoning problem based on an extension of the ARP protocol. We introduce a set of functionalities that enable an integrity and authenticity check on the content of ARP replies, using a combination of digital signatures and one time passwords based on hash chains.

Rest of the paper is organized as follows. Section 2 illustrates the problem considered in this paper and recalls how ARP works and why it is vulnerable to cache poisoning. Section 3 discusses the related work. Section 4 describes the proposed solution. Section 5 concludes the paper.

2 Problem Definition

2.1 Address Resolution Protocol

Hosts and applications in a network work with domain names which are converted to the IP address by a DNS server. But once packets containing application data arrive on the local Ethernet network of the host, they can be transmitted only if the MAC address buried in the NIC of the destination host is known to the switch. Thus a conversion is needed from IP addresses to MAC addresses and vice versa. This conversion is done by the Address Resolution Protocol or ARP in short [8, 6].

ARP works as follows. When a host needs to send an IP datagram as an Ethernet frame to another host whose MAC address it does not know, it broadcasts a request for the MAC address associated with the IP address of the destination. Every host on the subnet receives the request and checks if the IP address in the request is bound to one of its network interfaces. If this is the case, the host with the matching IP address sends a unicast reply to the sender of the request with the <IP address, MAC address> pair. Every host maintains a table of <IP, MAC> pairs, called the ARP cache, based on the replies it received, in order to minimize the number of requests sent on the network. No request is made if the <IP, MAC> pair of interest is already present in the cache. ARP cache entries have a typical lifetime of 20 minutes, after which the entry should be refreshed.

In ARP, a reply may be processed even though the corresponding request was never received, i.e., it is a stateless protocol. When a host receives a reply, it updates the corresponding entry in the cache with the <IP, MAC> pair in the reply. While a cache entry should be updated only if the mapping is already present, some operating

systems, e.g., Linux and Windows, cache a reply in any case to optimize performance.

2.2 ARP Cache Poisoning

Since ARP replies are not authenticated, an attacker can send an ARP reply containing a malicious <IP, MAC> association to any host on the network thus poisoning the ARP cache of that host. The attacker may supply her MAC address in the sent malicious association which enables her to receive all the packets sent by that host to the IP address specified in the association. This way the attacker may receive all the frames originally directed to some other host. If also the cache of the real destination host is poisoned, both communication flows are under the attacker's control. In this situation, the attacker could launch a man in the middle, where she can forward the received packets to the correct destination after inspecting and possibly modifying them. The two end points of the communication will not notice the extra hop added by the attacker if the packet TTL is not decremented.

Although cache can easily be poisoned when there is an entry in the cache for the targeted IP address, some operating systems, e.g. Solaris, will not update an entry in the cache if such an entry is not already present when an unsolicited ARP reply is received. Although this might seem a somewhat effective precaution against cache poisoning, the attack is still possible. The attacker needs to trick the victim into adding a new entry in the cache first, so that a future (unsolicited) ARP reply can update it. By sending a forged ICMP echo request as if it was from one of the two victims, the attacker has the other victim create a new entry in the cache. When the other victim receives the spoofed ICMP echo request, it replies with an ICMP echo reply, which requires resolving first the IP address of the original ICMP request into an Ethernet address, thus creating an entry in the cache. The attacker can now update it with an unsolicited ARP reply.

3 Related Works

ARP-based attacks are not easily prevented in current architectures. There are a handful of actions often recommended for mitigation. The first of these is employing static ARP, which renders entries in an ARP cache immutable. Thus any address resolution protocol is not employed at all. This is currently the only true defense [17], but is impractical. Windows machines ignore the static flag and always update the cache. In addition, handling static entries for each client in a network is unfeasible for all but the smallest networks. An administrator must deploy new entries to every machine on the network when a new client is connected, or when a network interface card (NIC) is replaced. Furthermore, this prevents the use of some DHCP configurations which frequently change MAC/IP associations during lease renewal.

The second recommended action is enabling port security on the switch. Also known as MAC binding, this is a feature of high-end switches which ties a physical port to a MAC address. This fixed address can be manually set by the administrator to

a range of one or more addresses, or can be auto-configured by the switch during the first frame transmission on the port. These port/address associations are stored in Content Addressable Memory (CAM) tables [15], a hardware-based reverse lookup device. A change in the transmitter's MAC address can result in port shutdown, or other actions as configured by the administrator. However, port security is far from ubiquitous and does nothing to prevent ARP spoofing [16]. Consider a man-in-the-middle attack as presented in [11]. An attacker X only needs to convince victim A to deliver frames meant for B to X, and vice versa for victim B. When sending forged ARP replies to achieve this, at no time must X forge its MAC address – only the cache of the clients is manipulated. Port security validates the source MAC in the frame header, but ARP frames contain an additional source MAC field in the data payload, and it is this field that clients use to populate their caches [6]. It should be said, however, that port security does prevent other attacks mentioned in [11] such as MAC flooding and cloning.

Virtual LANs (VLANs) create network boundaries which ARP traffic cannot cross, limiting the number of clients susceptible to attack. However, VLANs are not always an option and have their own set of vulnerabilities as detailed in [18].

Arpwatch [11] allows notification of MAC/IP changes via email. IDS and personal warn the user that the entry in the cache is changed. In these solutions, the decision is left to the user and his/her awareness. Given the particularly sophisticated level of operation in this case, it is doubtful that the average user will take the proper actions. Still, detection is an important step in mitigation. Solutions such as a centralized ARP cache or a DHCP server broadcasting ARP information, as they are deployed in IP over ATM networks [4], have their own problems as the attacker could spoof the source of the broadcast and poison the whole LAN [12].

Some kernel patches exist that try to defend against ARP poisoning. "Anticap" [1] does not update the ARP cache when an ARP reply carries a different MAC address for a given IP from then one already in cache and will issue a kernel alert that someone is trying to poison the ARP cache. "Antidote" [9] is more sophisticated. When a new ARP replies announcing a change in a <IP, MAC> pair is received, it tries to discover if the previous MAC address is still alive. If the previous MAC address replies to the request, the update is rejected and the new MAC address is added to a list of "banned" addresses. If Antidote is installed, an attacker may spoof the sender MAC address and force a host to ban another host.

The only kernel patch which assures mutual authentication between the requester and the replier even on the first message is Secure Link Layer [3]. SLL provides authenticated and encrypted communication between any two hosts on the same LAN. SLL requires a Certification Authority (CA) to generate SLL certificates for all legitimate hosts on the network. SLL handles authentication and session key exchange before any messages are transferred from one host to another. Elliptic curve cryptography algorithms are used for both operations. SLL defines three authentication messages that hosts send each other to perform mutual authentication and session key exchange. After authentication, the payload data field of all Ethernet frames sent between two hosts is encrypted with Rijndael using a 128-bit key and 128-bit long blocks. However, such a mechanism is too slow and complex for the purpose of ARP. Mutual authentication between two hosts is sufficient for avoiding ARP poisoning. Encrypting ARP replies does not yield any additional security since the associa-

tion between IP and MAC addresses should be public. Furthermore, SLL also maintains all the cryptographic keys in kernel-space. Note that the amount of memory required could be considerable in case of class B networks. It is not recommended to use kernel memory with information that could be as well managed in user space, such as keys. Hence although SLL is sufficiently secure, it has an unacceptable impact on the system performance.

S-ARP was recently proposed [12]. Each host has a public/private key pair certified by a local trusted party on the LAN, which acts as a Certification Authority. Each ARP reply is digitally signed by the sender, thus preventing the injection of malicious replies. At the receiving end, the cache entry is updated if and only if the signature is correctly verified. Cryptographic keys are maintained in the user space. Unnecessary services provided by SLL and not required for the sake of ARP are removed. Thus S-ARP is more efficient than SLL.

SLL and S-ARP are probably the only secure solutions for preventing the ARP cache poisoning. We however note that S-ARP still requires all ARP replies to be digitally signed. Recall that asymmetric key cryptography and digital signatures are considerably slow when compared to symmetric key cryptography and one way hash functions. Roughly, RSA signature generation is about 10,000 times slower than calculation of a one way hash function [19]. Thus, even S-ARP may have unacceptable impacts on the system performance.

4 The Proposed Solution

4.1 The Basic Idea

Our solution for the prevention of ARP cache poisoning is based on a combination of digital signatures and one time passwords. One time passwords are based on hash chains. Cryptographic techniques can hardly be avoided as the receiver has to authenticate the ARP reply; however an intelligent use of cryptography is desired to avoid unacceptable performance penalties.

Our protocol requires periodic generation of digital signatures, the rate of generation being independent of the number of ARP requests being received. For this, we identify two different components of an ARP reply:

1. The <IP address, MAC address> mapping
2. The recency of the mapping

The first component requires a digital signature since the <IP, MAC> mapping must be authentic and its authenticity must be publicly verifiable. Our idea is to however to use the same digital signature again and again in ARP replies for a long time.

Here one option could be to trust the ARP reply for recency and the only check performed on the content of replies would be validation of the digital signature. But then an attacker could get hold of that digital signature by simply sending an ARP request to the target system and getting it in reply. It could then wait for the target system to go down or it could crash the target system using known attacks or Denial of Service attacks. As soon as the target system goes down or gets disconnected from the network, the attacker could change her MAC address to that of the target system

and thus receive all packets sent to it. Now, even when the target system comes up later, it cannot claim back its MAC address and has to change it. The attacker may continue to poison the ARP cache of other hosts using the stored digital signature and thus receive the packet sent to the target system.

Hence we need a method to somehow securely indicate the recency of the mapping indicated in the digital signature. This is done by including a one time password in the ARP reply. Thus, the basic idea of our protocol is:

Generate a digital signature *S* containing the IP address to MAC address mapping, the local clock time and the tip of a hash chain used for verifying one time passwords. Now, for the first 20 minutes (cache entry validity time), the host answers ARP requests by sending *S* as the ARP reply. For the next 20 minutes, the host sends *S* and the first one time password (first link of the hash chain) as ARP reply and so-on. In general for the i^{th} 20 minute slot, the host sends *S* and the $(i\text{-}1)^{th}$ one time password as the ARP reply.

This process is continued till the one time passwords (or the links of the hash chain) do not get exhausted or the <IP, MAC> association of the host does not change. After this, a new signature S' should be generated and the whole process be repeated.

We now describe the one time password system being used and then move on the detailed description of the proposed protocol using that.

4.2 One Time Passwords

We use a variant of the one time password system designed by Leslie Lamport [13, 14]. This system is popularly known as Lamport Hashes or S/KEY. This scheme is used to authenticate a client to an untrusted server and is based upon the concept of Hash chains. No security sensitive quantities are stored at the server, i.e., the password verifying token is public.

The one time passwords are generated using a secret *K* known only to the client. The client chooses an integer *N* and a random number *R* acting as the nonce. For system initialization, the client then somehow securely sends *N* and $H^{N+1}(K||R)$ to the server[1] (e.g. using digital signatures).

At any point of time, the server maintains the following 3-tuple entry for each client:

$$< id, n, H^{n+1}(K||R) > \quad \text{with } n=N \text{ initially}$$

The client authenticates by sending $H^n(K||R)$ to the server (along with its id). The server computes its hash and then compares it with the stored $H^{n+1}(K||R)$. If they match, $H^{n+1}(K||R)$ is replaced with $H^n(K||R)$, the value of stored *n* is decremented and the client is successfully authenticated.

When *n* reaches 0 at the server, i.e., when the client authenticates with $H(K||R)$, the list of one time passwords is considered to be exhausted. At this point, a new value of *R* must be chosen and the system should be reinitialized.

[1] H is a one way hash function like MD5 and || denotes concatenation

4.3 Network Setup

The setup phase in our system is similar to that in S-ARP. Every host on the network is identified by its own IP address and has a public/private key pair. Besides, there is a trusted host on the network called the Authoritative Key Distributor (AKD) which handles the task of key distribution and clock synchronization.

Note that this AKD based architecture can easily be converted to Certificate Authority (CA) based architecture. This can be done by distributing a certificate containing the IP address to public key mapping to each host on the network. We elaborate more on this issue in section 4.6.1.

The first step when setting up a LAN that uses our protocol is to identify the AKD and distribute through a secure channel its public key and MAC address to all the other hosts. Such an operation may be performed manually when a host is installed on the LAN for the first time. On the other hand, a host that wants to connect to the LAN must first generate a public/private key pair and send the public key along with its IP address to the AKD. Here the correctness of the information provided is verified by the network manager and the host public key together with its IP address is entered in the AKD repository. This operation has to be performed only the first time a host enters the LAN. If a host wants to change its key, it communicates the new key to the AKD by signing the request with the old one. The AKD will update its key and the association is correctly maintained. Section 4.6 explains the protocol behavior when IP addresses are dynamically assigned by a DHCP server.

Once connected to the LAN, a host synchronizes its local clock with the one received from the AKD. To avoid the reply of old clock value from an adversary during clock synchronization, the host generates a random number R which it sends along with the synchronization request to the AKD. The AKD replies back with the current time t along with a digital signature on (t, R).

4.4 Message Format

The ARP request message format remains the same except for the addition of two new fields- “timestamp” and “type”. “timestamp” is the value of the local clock at the time of request generation. The value of “type” field may either be 1 or 2 to distinguish among the following two types of requests:

1. New entry request
2. Entry refreshment request

We discuss in the next section about how a host determines the type of request to send in any scenario.

Our protocol adds an extension to the ARP reply header. The extension comprises of a new field called “type” and a variable length payload called “data”. The field “type” distinguishes among the following six types of messages:

1. New entry creation (reply only)
2. Entry refreshment (reply only)
3. Public key management (request/reply)
4. Time synchronization (request/reply)

Messages of type 1 and 2 are exchanged between hosts of the LAN. The other types of messages are exchanged only between a host and the AKD.

4.5 The Protocol Description

Every host on the network first chooses an integer N, a secret K and a random number R. In practice, the first 128 bit of the private key of the host suffices as K. The significance and the choice of N will be clear later on.

The host now computes a signature[2]

$$S(IP, MAC, N, H^{N+1}(K||R), T)$$

Where T is the timestamp. This signature S will be useful for a period $(N+1)T_e$ where T_e = the cache entry validation time (usually equal to 20 min), i.e., the signature will be valid for a time T_e and can be renewed upto N times using N one time passwords. This signature is stored by the host and will be used later in answering ARP request.

4.5.1 Cache Entry Creation

Consider the scenario when a host H_i needs to know the MAC address of H_j. H_i checks its cache and consequently finds no entry for H_j. It queries H_j with an ARP request having field type=1, i.e., a new entry creation request is sent to H_j.

H_j now computes

$$n = (N+1) - \lfloor (t - T) / T_e \rfloor$$

Where t is the current time and the function $\lfloor x \rfloor$ is the floor function returning the largest integer smaller than its argument e.g., $\lfloor 3.9 \rfloor = 3$. Informally, $\lfloor (t - T) / T_e \rfloor$ is the number of 20 minute time slots passed after the signature S was computed.

H_j now calculates $H^n(K||R)$, i.e., the $(N+1-n)^{th}$ one time password. Note that if the time elapsed between the ARP request and the signature generation is less than T_e (20 min), $n = N+1$ and the one time password is $H^{N+1}(K||R)$ (already specified in the signature). Hence this can be seen as the zeroth one time password. Further $(N+1-n)$ cannot be negative or less than 1 as when it reaches 1, a new signature with new R should be computed.

H_j now sends an ARP reply to H_i with type=1 and data = $S(IP, MAC, N, H^{N+1}(K||R), T), n, H^n(K||R)$, i.e., data contains the signature S and the computed one time password. Upon receipt of the ARP reply, H_i verifies the signature using the public key of H_j (if it does not already have the required key, it obtains it from the AKD, see the next sub-section), validates the one time password supplied using N and $H^{N+1}(K||R)$ given in signature S and computes $t = T+T_e*(N+1-n)$. If t is within time T_e (i.e., 20 min) from the current local clock time, the reply is accepted and the following five tuple entry is created for H_j

$$<IP, MAC, n, H^n(K||R), t>$$

The last three values are stored in order to avoid having H_j send the signature S everytime and to avoid the overhead of signature verification in each ARP reply.

---

[2] This signature S represents the data IP, MAC, N, $H^{N+1}(K||R)$, T as well as the digital signature on it

4.5.2 Cache Entry Refreshment

Now consider the scenario in which H_i requires the MAC address of H_j and find a cache entry for it. It checks the stored values n and t. If t is within time T_e (i.e., 20 min) from the current local clock time, the cache entry is considered to be good. No ARP request is sent and the stored MAC address is used. Otherwise, it computes the value $t+nT_e$ and compares it with the local clock time. If the local clock time is more than $t+nT_e$, this corresponds to $(N+1–n)$ being less than 1, i.e., it corresponds to requiring one time password with index more than N which is non-existent (only N one time passwords exist). This means that the parent signature of the existing cache entry should have expired beyond renewal. At this point, the cache entry is deleted and an ARP request is sent with type=1, i.e., for the creation of a new cache entry. H_j replies as described in the previous subsection.

Finally, if the local clock time is less than $t+nT_e$, a request of type=2, i.e., a request for entry renewal is sent. H_j now computes the new n and the required one time password $H^n(K||R)$ as described in the previous sub-section. The only difference lies in the ARP reply which is of type=2 and the signature S is not included in the data to be sent. The ARP renewal reply can be validated by computing the hash of the sent one time password $(n'-n)$ number of times and comparing with the stored password, where n' is the value of new n included in the ARP reply. If they match, a check on the quantity $t+T_e*(n'–n)$ is performed which should be with time T_e (i.e., 20 min) from the current local clock time. If the check succeeds, t is replaced with $t+T_e*(n'–n)$, $H^n(K||R)$ with $H^{n'}(K||R)$, n is replaced with n' and the ARP reply is accepted.

Now, we discuss the issue when H_j changes its MAC address. In that case, H_j just needs to discard its old computed signature and should recompute it using the new MAC, a new R and correspondingly all new values including T. The entry creation requests proceeds without any change. In case of entry renewals, however, since the one time password supplied in this case will not be correctly validated by H_j storing the old entry corresponding to the old signature and the old MAC, H_i discards its stored cache entry and sends a new ARP request with type=1.

4.5.3 A Rough Performance Comparison with SARP

On the contrary to S-ARP, our scheme requires a constant number of digital signatures per unit time irrespective of the number of ARP entry creation/renewal request received, e.g. with N=100 and T_e = 20 min, we require only one digital signature computation for a time period of $t = (N+1)T_e$ = 33.6 hours. Thus considering a period of one month (30 days) with an average of one ARP request per second, our scheme requires (30*24)/33.6 < 22 signature computations while S-ARP requires 259,200 signature computations. Given that signature computation is about 10,000 times slower than the computation of a hash function [19], it is easy to see that our scheme dramatically improves the performance of the system. Note that our scheme is still exactly as secure as S-ARP.

4.6 Key Management

Key management in our protocol is pretty much the same as in S-ARP [12]. Note that special care is required to be taken when dealing with dynamically assigned IP ad-

dresses. Hence, we consider key management in networks with statically and dynamically assigned IP addresses separately.

We will use the following notations in this section:

AKD	Authoritative Key Distributor
$\boldsymbol{H_i}$	Generic host *i*
Rq(a)	Request for object *a*
Rp(a)	Reply carrying object *a*
T	Local clock Time-stamp
$\boldsymbol{A_H}$	Host *H*'s IP address
$\boldsymbol{M_H}$	Host *H*'s MAC address
$\boldsymbol{P_H}$	Host *H*'s Public Key
$\boldsymbol{S_H(x)}$	Message x digitally signed by host *H*

4.6.1 Static Networks

In such networks, the mapping between the keys and the IP addresses is static. Hence, when a host joins the network for the first time, a key pair and an IP address is assigned to it and inserted into the AKD repository. Now consider that a generic host *i* broadcasts an ARP request to find host *j*'s MAC address and upon receiving the reply finds that it does not have the public key of host *j*. Host *i* then contacts the AKD to request host *j*'s public key. AKD then sends the required key in a digitally signed message

The sequence of messages exchanged is as follows.

$\boldsymbol{H_i} \rightarrow \boldsymbol{AKD}$: $Rq(PH_j)$

$\boldsymbol{AKD} \rightarrow \boldsymbol{H_i}$: $S_{AKD}(Rp(PH_j) \parallel H_j \parallel T)$

Now, since the public key used for verifying the host's signatures has been securely released by the AKD and the private key corresponding to that public key is known only to the legitimate host, an attacker cannot produce a valid signature for an IP address other than its own. Thus an attacker can no longer send valid malicious ARP replies to poison a host's cache.

Here another possible way of key management is to provide digitally signed certificates to each host containing the mapping between its IP address and the public key. In this case, we require a CA instead of AKD and no active participation of CA in the protocol would be required. A third type of ARP request can be created for which the reply would contain the issued certificate. But such a mechanism will have to deal with intrusions like certificate revocation in case of key compromise. Hence we choose to stick with the option of AKD. Contrary to the public key infrastructures where certificates are used, we see no problem in having an online AKD in our case for the sake of ARP. Any host on the network as identified by the network administrator could act as AKD. Another reason for this choice is the possibility of dynamic networks (discussed next) where such certificates are not possible as IP addresses are dynamically assigned.

4.6.2 Dynamic Networks

In such a network, a DHCP server dynamically assigns IP addresses to the hosts. Since the IP address of a host is not fixed, keys cannot be bound to IP addresses at generation time.

If an organization deploys a secure DHCP server with secure ARP, only well known hosts that have been enrolled in the system and authorized in some way can enter the LAN. Enrollment procedure is a one time activity which takes places when the host joins the network for the first time. Enrollment involves generating a key pair and the corresponding certificate. In this case, IP field of the certificate is empty. AKD manually inserts this certificate into the repository with null IP address and the corresponding public key, using a secure channel.

When a host H joins the LAN, it requests the DHCP server to assign to it an IP address. In order to allow the DHCP server and the AKD to identify it and validate the request, H timestamps and signs the request and sends its public key along with the request. Before assigning an IP address to H, the DHCP server contacts the AKD to verify whether H is authorized to be added to the LAN, i.e., if H's key is in the AKD repository and is valid, and to inform the AKD of the IP address the host will be assigned. The message is signed by the DHCP server and includes the supplied public key along with the proposed IP address. The AKD searches its database for the given public key and replies to DHCP with a signed ACK or a NACK along with some other fields to prevent replay attacks. If the response from the AKD is ACK, the DHCP server proceeds with the assignment of the new IP address to H, while the AKD updates H's entry in the repository binding H's new IP address to H's key. Else, the DHCP server will not release a new IP to the host. The message exchange sequence in case of a positive response from the AKD is as follows:

H → DHCP: $P_H \,||\, S_H$(DHCP request $||\, T$)
DHCP → AKD: $S_{DHCP}(P_H \,||\, A_H \,||\, T)$
AKD → DHCP: $S_{AKD}(ACK \,||\, P_H \,||\, A_H \,||\, T)$
DHCP → H: S_{DHCP}(DHCP reply $||\, A_H \,||\, T$)

5 Conclusion

ARP cache poisoning occurs due to lack of message authentication since any host on the LAN can spoof ARP replies containing malicious IP to MAC mapping. There is no satisfactory solution to cache poisoning since all the proposed solutions are either insecure or have unacceptable penalties on system performance.

We propose a new solution to the problem of ARP cache poisoning. Our solution is based on an efficient combination of digital signatures and one time passwords based on hash chains. Digital signatures are almost eliminated in the sense that the system requires less than one digital signature per day. Thus, the performance of our protocol is significantly better than S-ARP which requires digital signature computation for each ARP reply. Further, we do not compromise security at any point and the security of our scheme is the same as S-ARP. Hence, our scheme is efficient as well as secure at the same time.

References

1. M. Barnaba. anticap. http://cvs.antifork.org/cvsweb.cgi/anticap, 2003.
2. B. Fleck. Wireless access points and arp poisoning [online document]. Available at http://www.cigitallabs.com/resources/papers/download/arppoison.pdf.
3. F. Hunleth. Secure link layer. http://www.cs.wustl.edu/fifhunleth/projects/projects.html.
4. M. Laubach. Classical IP and ARP over ATM. RFC 1577, 1994.
5. A. Ornaghi and M. Valleri. A multipurpose sniffer for switched LANs. http://ettercap.sf.net.
6. D. C. Plummer. An ethernet address resolution protocol. RFC 826, 1982.
7. D. Song. A suite for man in the middle attacks. http://www.monkey.org/fidugsong/dsniff.
8. R. W. Stevens. TCP/IP Illustrated, vol 1. Addison Wesley, ISBN 0-201-63346-9, 2001.
9. I. Teterin. Antidote. http://online.securityfocus.com/archive/1/299929.
10. R. Wagner. Address resolution protocol spoofing and man in the middle attacks. http://rr.sans.org/threats/address.php, 2001.
11. S. Whalen. An introduction to arp spoofing [Online document]. Available at http://packetstormsecurity.nl/papers/protocols/intro_to_arp_spoofing.pdf, 2001.
12. Bruschi, D., Ornaghi, A., Rosti, E., S-ARP: a Secure Address Resolution Protocol, in Proceedings of 19th Annual Computer Security Applications Conference (ACSAC), 2003.
13. Leslie Lamport, “Password Authentication with Insecure Communication”, Communications of the ACM 24.11 (November 1981), pp 770-772.
14. N Haller, "The S/KEY One-Time Password System", Proceedings of the ISOC Symposium on Network and Distributed System Security, pp 151-157, February 1994.
15. A. Stemmer, “CAMs Enhance Network Performance”, System Design [Online document], Jan. 98, Available HTTP: http://www.eedesign.com/editorial/1998/systemdesign9801.html
16. http://cert.uni-stuttgart.de/archive/vulndev/2002/ 01/msg00295.html
17. Whalen, S. H., Towards Layer 2 Authentication: Preventing Attacks based on Address resolution Protocols Spoofing., 2003. http://wp.netscape.com/eng/ssl3/draft302.txt, 2002.
18. S. Convery, “Hacking Layer 2: Fun with Ethernet Switches”, Blackhat [Online document], 2002, Available HTTP: http://www.blackhat.com/presentations/bh-usa-02/bhus-02-convery-switches.pdf
19. S. Micali “NOVOMODO: Scalable Certificate Validation and Simplified PKI Management” First Annual PKI Research Workshop - Proceeding, April 2002.
20. Hacking UNIX 2003, a tutorial for performing various attacks including ARP poisoning attack, on UNIX systems. Available at http://duho.cjb.net.
21. M. V. Tripunitara and P. Dutta. A middleware approach to asynchronous and backward compatible detection and prevention of arp cache poisoning. In Proc. 15th Annual Computer Security Application Conference (ACSAC), pages 303-309, 1999.

On Stern's Attack Against Secret Truncated Linear Congruential Generators

Scott Contini and Igor E. Shparlinski

Department of Computing, Macquarie University,
NSW 2109, Australia
{scontini,igor}@ics.mq.edu.au

Abstract. In 1987, Stern showed how the parameters for secret truncated linear congruential generators could be derived in polynomial time. Here, we present a modification to that algorithm which makes it simpler, more robust, and require less data. We then present a more careful analysis of the algorithm, and establish some limits of its applicability. Thus, secret truncated linear congruential generators may not necessarily be insecure for properly chosen parameters. Unfortunately, as in the original algorithm, all the results remain heuristic, however we present results of numerical experiments which support our conclusions.

1 Introduction

Linear congruential generators are a well known method for producing pseudo-random sequences [8]. They work as follows. Let m be a *modulus*, a an integer *multiplier* such that $\gcd(m, a) = 1$, and b an integer additive *shift*. Define the sequence $(x_i)_{i\geq 0}$ by

$$x_{i+1} \equiv ax_i + b \bmod m,\ 0 \leq x_i \leq m-1, \qquad i = 0, 1, \ldots, \tag{1}$$

where x_0 is a given *seed*.

Although such sequences are known to have good pseudo-randomness properties such as a large period length and uniformity of distribution, see [6,10,13,14], they are not suitable for cryptographic purposes. Knuth [7] suggests to keep the parameters a, b and m secret and output only a portion of the most significant bits of the sequence. However, it has turned out that in many cases such generators can still be broken, see [2,3,4,5,9,16]. In fact, even in the hardest case, when only the most significant bits of each x_i are available and all parameters are kept secret, such a generator has been cryptanalyzed by Stern in [16], see also [5]. All the aforementioned algorithms exploit the celebrated lattice basis reduction algorithm of Lenstra, Lenstra and Lovász [11], namely the so-called *LLL algorithm*.

Stern's attack [5,16] runs in polynomial time, but there is a potential problem with it from a practical viewpoint. Specifically, if one uses a single "wrong" polynomial (see Section 2 for details), the algorithm fails altogether. In our research, we modify the second step of the Stern algorithm. Our modification

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 52–60, 2005.

makes it simpler, more robust (bad polynomials do not prevent success), and require less data.

We then investigate whether the Stern attack is always guaranteed to work. In fact, we find that the algorithm breaks down if only about $\log k$ bits are output from each x_i, where k is the bit length of m and $\log z$ denotes the binary logarithm of $z > 0$. In this case, we do not know if the truncated linear congruential generator can still be cryptanalyzed.

Finally, we update the analysis of parameters given by Stern [5,16] to take into consideration this breakdown condition and more recent results in lattice reduction, see [1,15].

2 Stern's Algorithm

2.1 Initial Settings

Consider the sequence of data from (1). Let k denote the bit length of the modulus m, that is, $k = \lfloor \log m \rfloor + 1$. Assume that s most significant bits are output for every x_i. More formally, we assume that for every integer i we are given some "approximations" y_i such that

$$x_i = 2^{k-s} y_i + z_i, \quad \text{where } 0 \le z_i < 2^{k-s} \tag{2}$$

Thus, the y_i are formed by the bits that are output while the z_i are unknown. We also denote by $\alpha = s/k$ the proportion of the bits which are output.

Stern's algorithm [5,16] has two steps. The first step generates several polynomials $P_j(X)$ such that $P_j(a) \equiv 0 \bmod m$, $j = 1, 2, \ldots$. The second step uses these polynomials to first determine m and then a. Once m and a are known, the generator becomes completely predictable following the results of [4].

2.2 First Step

Consider the vectors

$$V_i = \begin{pmatrix} y_{i+1} - y_i \\ y_{i+2} - y_{i+1} \\ \vdots \\ y_{i+t} - y_{i+t-1} \end{pmatrix}$$

for $i = 1, \ldots, n$, where t and $n > t$ are certain positive integer parameters to be discussed later. We seek a small vector $(\lambda_1, \ldots, \lambda_n)$ such that

$$\sum_{i=1}^{n} \lambda_i V_i = 0. \tag{3}$$

Such a relation is guaranteed to exist with all $|\lambda_i| \le B$ where

$$B = 2^{t(\alpha k + \log n + 1)/(n-t)},$$

see [5]. By applying the LLL algorithm [11] to the lattice spanned by the columns of the following matrix

$$\begin{pmatrix} KV_1 & KV_2 & \dots & KV_n \\ 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}, \tag{4}$$

where $K = \lceil \sqrt{n} 2^{(n-1)/2} B \rceil$, we are assured to find a vector $\lambda = (\lambda_1, \dots, \lambda_n)$ whose Euclidean norm $\|\lambda\|$ satisfies $\|\lambda\| \le K$ and the the equation (3) holds.

For properly chosen parameters, it turns out that

$$U = \sum_{i=1}^{n} \lambda_i W_i \tag{5}$$

is 0, where

$$W_i = \begin{pmatrix} x_{i+1} - x_i \\ x_{i+2} - x_{i+1} \\ \vdots \\ x_{i+t} - x_{i+t-1} \end{pmatrix}.$$

In other words, a linear relation among the vectors involving partially known bits implies a linear relation among the vectors involving all of the bits.

It is easy to verify that $x_{i+j+1} - x_{i+j} \equiv a^j (x_{i+1} - x_i) \bmod m$. Thus, the vectors W_i satisfy

$$W_i \equiv \begin{pmatrix} a^{i-1} (x_2 - x_1) \\ a^i (x_2 - x_1) \\ \vdots \\ a^{i+t-2} (x_2 - x_1) \end{pmatrix} \bmod m.$$

So, if $U = 0$, then we get the polynomial

$$f(X) = (x_2 - x_1) \sum_{i=1}^{n} \lambda_i X^{i-1},$$

which satisfies $f(a) \equiv 0 \bmod m$ (from the top row of the W_i). We assume $\gcd(x_2 - x_1, m) = 1$ and instead use the polynomial

$$P(X) = \sum_{i=1}^{n} \lambda_i X^{i-1}$$

which has the same property, see [5] for justification of this assumption.

In [5,16], it is shown that by choosing

$$t > 1/\alpha \quad \text{and} \quad n \approx \sqrt{2\alpha t k}, \tag{6}$$

the first step is expected to work. We sketch the Stern proof [5,16] since we modify the parameter choices later. Because of (3), we have

$$U = \sum_{i=1}^{n} \lambda_i Z_i$$

where

$$Z_i = \begin{pmatrix} z_{i+1} - z_i \\ z_{i+2} - z_{i+1} \\ \vdots \\ z_{i+t} - z_{i+t-1} \end{pmatrix}.$$

Note that $\|Z_i\| < \sqrt{t}2^{(1-\alpha)k}$. Using the Cauchy-Schwarz inequality, it can then be shown that $\|U\| < M$, where $M = \sqrt{nt}2^{(1-\alpha)k}\|\lambda\|$. With the parameter choices suggested above and the bound $\|\lambda\| \leq K$, we see that

$$\begin{aligned} M &< \sqrt{nt}2^{(1-\alpha)k}K = O\left(n\sqrt{t}2^{(1-\alpha)k+(n-1)/2}B\right) \\ &= O\left(n\sqrt{t}2^{(1-\alpha)k+(n-1)/2+t(\alpha k+\log n+1)/(n-t)}\right) \end{aligned}$$

In particular, if $n = o(\log m)$ and $t = o(n)$, then $M \leq m^{1-\alpha+o(1)}$.

On the other hand, the vectors W_i are in the lattice spanned by the columns of the following matrix:

$$\begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ a & m & 0 & \dots & 0 \\ a^2 & 0 & m & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a^{t-1} & 0 & 0 & \dots & m \end{pmatrix}. \tag{7}$$

The determinant of this lattice is m^{t-1}, so we expect its smallest vectors to be on the order of $m^{1-1/t}$. By the choice of $t > \frac{1}{\alpha}$, we have made M much smaller than this value. Thus, unless the lattice has a vector that is much smaller than expected, the only possibility for U is 0. See [5,16] for more rigorous details, and [4] for a statement on the probability of the lattice having exceptionally small nonzero vectors.

2.3 Second Step

By repeatedly applying the first step, we get a sequence of polynomials (P_j) of degree $n-1$ such that $P_j(a) \equiv 0 \bmod m$. These polynomials can be written uniquely as an integer linear combination of the polynomials $Q_i(X) = X^i - a^i$ for $1 \leq i \leq n-1$ and the constant polynomial $Q_0(X) = m$. By identifying the P_j with $\mathbb{Z}^n$ column vectors, we see that the vectors are in the span of the lattice L given by the matrix

$$\begin{pmatrix} m & -a & -a^2 & \dots & -a^{n-1} \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix},$$

which has determinant m. Stern [5,16] argues that given slightly more than n polynomials, we expect them to generate the same lattice. Hence, by computing the determinant of the lattice from our polynomials, we can discover m. Although a rigorous proof of success is not available, both the experiments in [5,16] and our experiments confirm this argument.

Upon finding m, one can then determine a. Stern [5,16] has suggested a few algorithms to do so. In [16] it is claimed that there exists an algorithm to reconstruct the Q_i. Clearly, if $m = p$ is prime, one can easily find a by solving congruences of the form $P_i \equiv 0 \bmod p$. In [5], a certain lattice based method is given that allows us to determine Q_1 without computing the other Q_i.

3 Practical Difficulties with Stern's Algorithm

The second step of Stern's algorithm described in Section 2.3 is extremely fragile: if we accidentally obtain a polynomial that does not satisfy $P_j(a) \equiv 0 \bmod m$, then the second step does not work. While it may be possible to use some type of an efficient algorithm to eliminate the bad P_j, we believe that our solution in Section 4 is more appealing, especially since it requires less data.

To motivate this problem more, consider the following example. Let

$$\begin{aligned} m &= 10734367385013619889\ (k = 64), \\ a &= 9807963723765715717, \\ b &= 7226300108115682840, \\ x_0 &= 2877244225168654778. \end{aligned}$$

We output $\alpha = \frac{1}{2}$ of the most significant bits. The first 28 data points are given in the table below.

475990822	277168665	951085457	509438822	1408360423	487756910	1233629515
731412381	1071978143	1524265123	1101665588	1800377470	1598061595	2219298046
1762030258	995024674	935996983	2369274372	577879031	1145304475	1279507381
1843354788	457858814	1627871073	1984671691	146090605	728684809	1646993503

According to the asymptotic formulas, we may be able to succeed with $t = 3$ and $n = 14$. Using the Magma computer algebra package [12] to perform the LLL algorithm on the lattice (4), we obtain with the following reduced lattice, where $K = 356131$:

0	0	0	0	0	0	0	0	0	0	0	K	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	K
0	0	0	0	0	0	0	0	0	0	0	0	K	0
67	44	49	26	3	47	36	5	57	113	1687	-24	32	83
-96	4	-82	97	66	89	-22	-72	-28	-143	-715	81	82	72
-29	-45	-102	-3	-70	85	-33	-82	214	825	-1075	-24	32	111
-19	54	5	-45	-6	32	-100	134	66	805	860	5	44	79
-35	-59	-38	-42	135	-8	-24	37	116	-709	1129	-17	83	34
21	-4	-12	61	-89	-88	-7	48	148	-1275	447	-38	-41	70
41	-47	-37	-31	-33	46	77	85	-11	-264	-364	7	27	73
-96	34	28	-64	51	26	82	-32	-86	-220	557	-4	14	-73
14	-55	87	-59	76	-17	5	-4	131	730	-1182	-81	9	10
-40	-23	-12	141	-26	-141	-28	69	-115	968	123	63	-64	-24
2	-127	-70	-3	-28	22	3	-103	9	-201	1669	6	-28	-25
-41	24	84	17	-43	45	-195	82	-40	-717	-226	40	-26	52
30	-83	-38	23	60	-6	41	57	-48	100	-443	32	36	33
-79	-25	-14	47	-34	52	-22	-63	15	-27	978	21	-5	45

One would expect that we can use the shortest returned vector (the first one), though Stern remarks that often there is more than one valid polynomial. But how can we be confident that any vector, including the shortest, actually results in a valid polynomial?

The answer to this question partially rests in a practical use of the asymptotic formulas. The attacker requires vectors λ satisfying

$$M = \sqrt{nt}2^{(1-\alpha)k}\|\lambda\| < 2^{k(1-1/t)} \tag{8}$$

to have a reasonable chance of success. Substituting our parameters, this means $\|\lambda\| < 2^{7.97}$. We actually got nine vectors with relatively small norms. In order, their norms are $\approx 2^{7.61}$, $2^{7.67}$, $2^{7.72}$, $2^{7.79}$, $2^{7.85}$, $2^{7.87}$, $2^{8.02}$, $2^{8.06}$, and $2^{8.50}$. If we were to take the inequality (8) literally, then we would expect the first six vectors to produce valid polynomials. But, in fact only the fourth, fifth, and ninth vectors work! So, even if we would have restricted to only using the first (shortest) vector, Stern's algorithm would have failed.

Now, the reader may not be very satisfied with our example, since the inequality (8) should not be taken literally. A simple solution would be to only take vectors that are much smaller than the threshold bound. However, our view is that this is not necessary since an improved algorithm exists which allows us to determine both m and a from only the single reduced lattice basis given above. In contrast, Stern's algorithm requires several LLL calls, all of which *must* produce valid polynomials.

4 Improving the Second Step

Each iteration of Stern's first step involves a single LLL call. If we assume that we get one valid polynomial per iteration, then we need at least n LLL calls to apply the second step. In comparison, our modified second step only requires a

constant number of calls to LLL. Moreover, it allows us to include vectors that may not produce valid polynomials (they may not even satisfy the inequality (8)) without having the algorithm fail altogether. Finally, our algorithm requires less data points: $n+t+2$ data points are sufficient, whereas Stern's algorithm requires more than $2n+t$ data points (again, assuming one polynomial taken per iteration, and note that data points must be consecutive[1]). Similar to Stern's second step, our solution is polynomial time but not rigorously proved, though always works in practice.

Our modification is quite simple. If $P_i(a) \equiv 0 \bmod m$ and $P_j(a) \equiv 0 \bmod m$, then the resultant of P_i and P_j is divisible by m. If we have a third polynomial $P_k(a) \equiv 0 \bmod m$, we can compute three resultants. Taking their greatest common divisor, we are very likely to get m or else a small multiple of it.

In fact, it is still polynomial time if we include all vectors returned from LLL (of norm less than K) and consider all combinations of three. If we have an incorrect polynomial such that $P_j(a) \not\equiv 0 \bmod m$, then the resultant is most likely to have either no common factor or a very small common factor with resultants of other polynomials. So false computations can easily be identified, and correct ones are identifiable by candidates for m of the right size. Furthermore, if one has more than three correct polynomials, then there are several candidates $\widetilde{m}$ which are no more than a small multiple of m. In the example from Section 3, the resultants from the fourth, fifth, and ninth polynomials yield m exactly.

It is not difficult to find the correct m from even a single candidate $\widetilde{m}$ (though we will generally have many candidates) which is a small multiple of m. From the known results about the uniformity of distribution of linear congruential generators, see [6,13,14], we conclude that for the choice of n of order about $\sqrt{\log m}$ (see (6)) with overwhelming probability the largest value

$$\mu = \max_{i=1,\ldots,n} 2^{k-s} y_i$$

falls in the interval $m/2 < \mu \le m$, and thus can be used to find m amongst the divisors of $\widetilde{m}$. We remark that concrete forms of such uniformity of distribution results depend on the arithmetic structure of m and also on the period length of the generator (that is, the multiplicative order of a modulo m) but hold for almost all generators, see [6,10,13,14] for more details.

Stern's second step also requires n correct polynomials to compute a, except when m is prime. Our modification involves using the polynomials that revealed m. We simply take the greatest common divisor of these polynomials modulo m to find the root a. The Euclidean greatest common divisor algorithm succeeds unless the leading coefficient of one of the polynomials becomes a divisor of m during the process. In this case, one has found a factor of m in polynomial time, which allows him or her to repeat the process modulo the factors and reassemble the result with the Chinese Remainder Theorem. Thus, we find both m and a using less data than Stern's solutions.

[1] If a data point in the middle is omitted, then the resulting lattice has much smaller determinant than the lattice (7), so the whole algorithm breaks down.

5 The Limits of Stern's Algorithm

For Stern's algorithm to work, we expect to need the inequality (8) to hold. This can be rewritten as

$$\sqrt{nt}\|\lambda\| < 2^{k(\alpha-1/t)}.$$

Note that in order for any linear relation to exist among the vectors V_i, we almost certainly need $n \geq t$. Being overly optimistic, we assume $n = t$ and $\|\lambda\| = 1$, and then take logarithms to get

$$\log t < k(\alpha - \frac{1}{t}). \tag{9}$$

Let $\alpha \leq k^{-1} \log k$, so we require $\log t < \log k - k/t$ or $k/t < \log(k/t)$, which is impossible. Thus, outputting about $\log k$ bits per iteration may be safe from Stern's attack.

One has to be careful in drawing such conclusions. As we have seen in Section 3, it is sometimes the case that one finds valid polynomials even though the inequality (8) is not satisfied. We expect that our overly optimistic assumptions of $n = t$ and $|\lambda| = 1$ counteract this effect. Our experiments using the example from Section 3 seem to confirm this. In fact, taking α as large as $1/8$, we found no valid polynomials for all $t \leq 50$ and $n \leq 100$.

6 Updating Stern's Parameter Choices

Stern shows that asymptotically, we need $t > \alpha^{-1}$. In fact, this is clear from the inequality (9). However, for a given value of k, it is possible to choose $t > \alpha^{-1}$ and still not satisfy the inequality (9). Now, if we wanted to maximise our chance of success, we might try to choose a value of t such that $k(\alpha - 1/t) - \log t$ is as large as possible. On the other hand, this strategy results in extremely large values of t (implying high computation and data requirements) that offer no practical benefit. Thus, we recommend starting with a value towards the lower end bound, and increasing t if necessary.

Stern's analysis [5,16] shows that $n \approx \sqrt{2\alpha tk}$ minimises the bound M on the value of U, where U is defined in (5). The bound on M in [5,16] is based upon the LLL approximation factor of $\sqrt{n}2^{(n-1)/2}$ which appears in our constant K. Since then, an asymptotically better value of the approximation factor has been discovered. Namely, in [1] a probabilistic polynomial time algorithm is given which has the approximation factor $2^{cn \log\log n/\log n}$ for any constant $c > 0$. A slightly larger value $2^{cn(\log\log n)^2/\log n}$ is given by Schnorr [15] (it is useful to remark that the algorithm of [15] is also very practical and deterministic). In particular, using the results of [1] leads us to to the bound

$$M \leq \sqrt{t}2^{(1-\alpha)k+t(\alpha k+\log n+1)/(n-t)+cn\log\log n/\log n}$$

where $c > 0$ is an arbitrary constant. To asymptotically balance the two terms in the exponent which depend on n we now choose a slightly larger value of n, namely,

$$n \approx \sqrt{c^{-1}\alpha t k \frac{\log(\alpha t k)}{\log\log(\alpha t k)}} \ .$$

Note that this implies that the larger value of n has a higher chance of success, not that we are required to use it. If we want to minimise the data requirements, we only require $n = o(k)$.

References

1. M. Ajtai, R. Kumar and D. Sivakumar, 'A sieve algorithm for the shortest lattice vector problem', *Proc. 33rd ACM Symp. on Theory of Comput.*, ACM, 2001, 601–610.
2. J. Boyar, 'Inferring sequences produced by pseudo-random number generators', *J. ACM*, **36** (1989), 129–141.
3. J. Boyar, 'Inferring sequences produces by a linear congruential generator missing low–order bits', *J. Cryptology* **1** (1989) 177–184.
4. A. M. Frieze, J. Hastad, R. Kannan, J. C. Lagarias, and A. Shamir. 'Reconstructing truncated integer variables satisfying linear congruences', *SIAM J. Comp.*, **17** (1988), 262–280.
5. A. Joux and J. Stern, 'Lattice reduction: A toolbox for the cryptanalyst', *J. Cryptology*, **11** (1998), 161–185.
6. S. V. Konyagin and I. Shparlinski, *Character sums with exponential functions and their applications*, Cambridge Univ. Press, Cambridge, 1999.
7. D. E. Knuth, 'Deciphering a linear congruential encryption', *IEEE Trans. Inf. Theory* **31** (1985), 49–52.
8. D. E. Knuth, *The art of computer Programming: Seminumerical algorithms, Vol.2*. Addison-Wesley, 1981.
9. H. Krawczyk, 'How to predict congruential generators', *J. Algorithms*, **13** (1992), 527–545.
10. P. Kurlberg and C. Pomerance, 'On the period of the linear congruential and power generators', *Acta Arith.*, (to appear).
11. A. K. Lenstra, H. W. Lenstra and L. Lovász, 'Factoring polynomials with rational coefficients', *Mathematische Ann.*, **261** (1982), 513–534.
12. Magma Computer Algebra Package, `http://magma.maths.usyd.edu.au/magma/`.
13. H. Niederreiter, 'Quasi-Monte Carlo methods and pseudo-random numbers', *Bull. Amer. Math. Soc.*, **84** (1978), 957–1041.
14. H. Niederreiter, *Random number generation and Quasi–Monte Carlo methods*, SIAM Press, 1992.
15. C. P. Schnorr, 'A hierarchy of polynomial time basis reduction algorithms', *Theor. Comp. Sci.*, **53** (1987), 201–224.
16. J. Stern, 'Secret linear congruential generators are not cryptographically secure', *Proc. 28th IEEE Symp. on Found. of Comp. Sci.*, IEEE, 1987, 421–426.

On the Success Probability of χ^2-attack on RC6

Atsuko Miyaji* and Yuuki Takano

Japan Advanced Institute of Science and Technology.
{miyaji, ytakano}@jaist.ac.jp

Abstract. Knudsen and Meier applied the χ^2-attack to RC6. The χ^2-attack can be used for both distinguishing attacks and key recovery attacks. Up to the present, the success probability of key recovery attack in any χ^2-attack has not been evaluated theoretically without any assumption of experimental results. In this paper, we discuss the success probability of key recovery attack in χ^2-attack and give the theorem that evaluates the success probability of a key recovery attack without any assumption of experimental approximation, for the first time. We make sure the accuracy of our theorem by demonstrating it on both 4-round RC6 without post-whitening and 4-round RC6-8. We also evaluate the security of RC6 theoretically and show that a variant of the χ^2-attack is faster than an exhaustive key search for the 192-bit-key and 256-bit-key RC6 with up to 16 rounds. As a result, we succeed in answering such an open question that a variant of the χ^2-attack can be used to attack RC6 with 16 or more rounds.

Keywords : block cipher, RC6, χ^2 attack, statistical analysis

1 Introduction

The χ^2-attack makes use of correlations between input (plaintext) and output (ciphertext) measured by the χ^2-test. The χ^2-attack was originally proposed by Vaudenay as an attack on the Data Encryption Standard (DES) [14], and Handschuh *et al.* applied that to SEAL [4]. The χ^2-attack is used for both distinguishing attacks and key recovery attacks. Distinguishing attacks have only to handle plaintexts in such a way that the χ^2-value of a part of ciphertexts becomes significantly a high value. On the other hand, key recovery attacks have to rule out all wrong keys, and single out exactly a correct key by using the χ^2-value. Therefore, key recovery attacks often require more work and memory than distinguishing attacks.

RC6 is a 128-bit block cipher and supports keys of 128, 192, and 256 bits [12]. RC6-$w/r/b$ means that four w-bit-word plaintexts are encrypted with r rounds by b-byte keys. In [3,8], the χ^2-attacks were applied to RC6. They focused on the fact that a specific rotation in RC6 causes the correlations between input and output, and estimated their key recovery attack directly from results of a

* Supported by Inamori Foundation.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 61–74, 2005.

distinguishing attack [8]. The χ^2-attacks to a simplified variant of RC6 such as RC6 without pre- or post-whitening or RC6 without only post-whitening are further improved in [11] or [5], respectively. We may note that key recovery attacks in [11,5] differ from that in [8]: The variance of χ^2-value is taken into account to recover a key in [11,5] but not in [8]. They also pointed out the significant difference between the distinguishing attack and the key recovery attack: The distinguishing attack succeeds if and only if it outputs high χ^2-value, but the key recovery attack does not necessarily succeed even if it outputs high χ^2-value. In fact, their key recovery attack can recover a correct key in the high probability with a rather lower χ^2-value. This indicates that the security against the key recovery attack cannot be estimated directly from that against the distinguishing attack. Table 1 summarizes the previous results on RC6.

Table 1. Attacks on RC6

Attack	Target RC6	Rounds	#Texts
Linear Attack [1]	RC6	16	2^{119}
Multiple Linear Attack [15]	192-bit-key RC6	14[1]	$2^{119.68}$
χ^2 Attack [8]	128-bit-key RC6	12	2^{94}
	192-bit-key RC6	14	2^{108}
	256-bit-key RC6	15	2^{119}
χ^2 Attack [11]	128-bit key RC6W[2]	17	$2^{123.9}$
χ^2 Attack [5]	128-bit key RC6P[3]	16	$2^{117.84}$
Our result	192-bit-key RC6	16	$2^{127.20}$
	256-bit-key RC6	16	$2^{127.20}$

1: A weak key of 18-round RC6 with 256-bit key can be recovered by $2^{126.936}$ plaintexts with the probability of about $1/2^{90}$.
2: RC6W means RC6 without pre- or post-whitening.
3: RC6P means RC6 without post-whitening.

Theoretical analysis on χ^2-attack has been done by [16,10]. In [16], the average of χ^2-value based on the distinguishing attack [8] on RC6 is theoretically computed, which enables to compute the necessary number of plaintexts for the χ^2-value with a certain level. As a result, the necessary number of plaintexts for distinguishing attacks can be estimated theoretically in each round. However, this cannot evaluate the success probability of key recovery attacks directly since there is the significant difference between the distinguishing attack and the key recovery attack as mentioned above. On the other hand, theoretical difference between a distinguishing attack and a key recovery attack on RC6 without post-whitening [5] has been discussed in [10]. They make use of the idea of the theoretical and experimental complexity analysis on the linear cryptanalysis [6,13] to fit it in the theoretical and experimental complexity analysis on the χ^2-attack. They also present the theorem to compute the success probability of key recovery attacks by using the results of distinguishing attack, and, thus, they

can succeed to estimate the security against key recovery attack on RC6 with rather less work and memory. However, their estimation requires experimental results of distinguishing attacks. Up to the present, the success probability of key recovery attack in χ^2-attack has not been evaluated theoretically without any assumption of experimental results.

In this paper, we investigate the success probability of key recovery attack in χ^2-attack, for the first time, and give the theorem that evaluates the success probability of a key recovery attack without any experimental result. First we deal with a key recovery attack on RC6 without post-whitening [5] and give the theorem that evaluates the success probability theoretically. We make sure the accuracy of our theorem by comparing our approximation with the experimental results [5]. With our theory, we also confirm that 16-round 128-bit-key RC6 without post-whitening can be broken, which reflects the experimental approximation [5]. Then we improve the key recovery attack to work on RC6 itself. The primitive extension to RC6 are shown in [5], but it does not seem to work. We give the theorem that evaluates the success probability of the key recovery attack on RC6 theoretically. We also demonstrate our theorem on 4-round RC6-8 and make sure the accuracy by comparing our approximation with the experimental results. With our theory, we confirm that 16-round 192-bit-key and 256-bit key RC6 can be broken. As a result, we can answer the open question of [8], that is, whether χ^2-attack can be used to attack RC6 with 16 or more rounds.

This paper is organized as follows. Section 2 summarizes the notation, RC6 algorithms, the χ^2-test, and statistical facts used in this paper. Section 3 reviews the χ^2-attack against RC6 without post-whitening and the theoretical relation between a distinguishing attack and a key recovery attack. Section 4 presents the theorem of success probability of key recovery attacks on RC6 without post-whitening and investigates the accuracy by comparing the approximations of success probability to 4-round RC6 without post-whitening with implemented results. Section 5 improves the key ercovery algorithm on RC6 without post-whitening to that on RC6 and presents the theorem of success probability of the key recovery attacks on RC6. We investigate the accuracy by demonstrating the key recovery algorithm on RC6-8. We also discuss the applicable round of key recovery attack. A conclusion is given in Section 6.

2 Preliminary

We summarize the χ^2-test, statistical facts, and RC6 algorithms[12], used in this paper.

2.1 Statistical Facts

We make use of the χ^2-statistics [9] to distinguish a distribution with an unknown probability distribution $\mathbf{p}$ from an expected distribution with a probability distribution π. Let $X = X_0, ..., X_{n-1}$ be a sequence of $\forall X_i \in \{a_0, \cdots, a_{m-1}\}$ with unknown probability distribution $\mathbf{p}$, and $N_{a_j}(X)$ be the number of X which

takes on the value a_j. The χ^2-statistic of X which estimates the distance between the observed distribution and the expected distribution $\pi = (\pi_1, \cdots, \pi_m)$ is defined:

$$\chi^2 = \sum_{i=0}^{m-1} \frac{(N(a_i) - n\pi_i)^2}{n\pi_i}. \tag{1}$$

After computing the χ^2-statistic of X, we decide which hypothesis holds.

$$\begin{cases} H_0 : \mathbf{p} = \pi & \text{(null hypothesis)} \\ H_1 : \mathbf{p} \neq \pi & \text{(alternate hypothesis)} \end{cases} \tag{2}$$

The following Theorems 1 and 2 on χ^2-statistic are known.

Theorem 1 ([17]). *When H_0 is true, χ^2 statistic given by equation (1) follows χ^2 distribution whose freedom is $m-1$ approximately. In addition, the expected mean or variance is calculated by $E_{H_0}(\chi^2) = m-1$ or $V_{H_0}(\chi^2) = 2(m-1)$, respectively.*

Theorem 2 ([17]). *When H_1 is true, χ^2 statistic given by equation (1) follows noncentral χ^2 distribution whose freedom is $m-1$ approximately. In addition, the mean or variance is computed by $E_{H_1}(\chi^2) = m-1+n\theta$ or $V_{H_1}(\chi^2) = 2(m-1)+4n\theta$, respectively, where $n\theta$ so called noncentral parameter is $n\theta = n\sum_{i=0}^{m-1} \frac{(\pi_i - P(a_i))^2}{\pi_i}$, where $P(a_i)$ is the probability of occurence of a_i.*

In our case of which distinguishes a non-uniformly random distribution from uniformly random distribution [7,8,9], the probability π is equal to $\frac{1}{m}$ and, thus, equation (1) is simply described as follows.

$$\chi^2 = \frac{m}{n} \sum_{i=0}^{m-1} \left(n_i - \frac{n}{m}\right)^2. \tag{3}$$

Table 2 presents threshold for a 63 degrees of freedom. For example, (level, χ^2_{63}) = (0.95, 82.53) in Table 2 means that the value of the χ^2-statistic exceeds 82.53 in the probability of 5% if the observation X is uniform.

Table 2. χ^2-distributions with a 63 degree of freedom

Level	0.50	0.60	0.70	0.80	0.90	0.95	0.99
χ^2_{63}	62.33	65.20	68.37	72.20	77.75	82.53	92.01

Let us describe other statistical facts together with the notation.

Theorem 3 (Central Limit Theorem [2]). *Choose a random sample from a population which mean or variance is μ or σ^2, respectively. If the sample size n is large, then the sampling distribution of the mean is closely approximated by the normal distribution, regardless of the population, where the mean or variance is given by μ or σ^2/n, respectively.*

We also follow commonly used notation: the probability density and the cumulative distribution functions of the standard normal distribution are denoted by $\phi(x)$ and $\Phi(x)$; the probability of distribution X in the range $X \leq I$ is denoted by $\Pr(X \leq I)$; and $\mathcal{N}$ is used for the normal distributions. The probability density function of the normal distribution with the mean μ and the variance σ^2, $\mathcal{N}(\mu, \sigma^2)$, is given by the following equation,

$$\phi_{(\mu,\sigma^2)}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right].$$

2.2 Block Cipher RC6

Before showing the encryption algorithm of RC6, we give some notation.

$\{0,1\}^k$: k-bit data
$\mathrm{lsb}_n(X)$: least significant n-bit of X;
$\mathrm{msb}_n(X)$: most significant n-bit of X;
$\oplus$: bit-wise exclusive OR;
$a \lll b$: cyclic rotation of a to the left by b-bit;
S_i : i-th subkey (S_{2i} and S_{2i+1} are subkeys of the i-th round);
r : number of rounds;
(A_i, B_i, C_i, D_i) : input of the i-th round ;
(A_0, B_0, C_0, D_0) : plaintext;
$(A_{r+2}, B_{r+2}, C_{r+2}, D_{r+2})$: ciphertext after r-round encryption;
$f(x)$: $x \times (2x+1)$;
$F(x)$: $f(x) \pmod{2^{32}} \lll 5$;
$x||y$: concatenated value of x and y.

The detailed algorithm of RC6 is given:

Algorithm 1 (RC6 Encryption Algorithm)
1. $A_1 = A_0$; $B_1 = B_0 + S_0$; $C_1 = C_0$; $D_1 = D_0 + S_1$;
2. `for` $i = 1$ `to` r `do:` $t = F(B_i)$; $u = F(D_i)$; $A_{i+1} = B_i$;
   $B_{i+1} = ((C_i \oplus u) \lll t) + S_{2i+1}$; $C_{i+1} = D_i$; $D_{i+1} = ((A_i \oplus t) \lll u) + S_{2i}$;
3. $A_{r+2} = A_{r+1} + S_{2r+2}$; $B_{r+2} = B_{r+1}$; $C_{r+2} = C_{r+1} + S_{2r+3}$; $D_{r+2} = D_{r+1}$.

Parts 1 and 3 of Algorithm 1 are called pre-whitening and post-whitening, respectively. A version of RC6 is specified as RC6-$w/r/b$. In this paper, we simply write RC6 if we deal with RC6-32. We also call the version of RC6 without post-whitening to, simply, RC6P.

2.3 A Transition Matrix

A transition matrix describes input-output transition, which was introduced in [14] and applied to RC6-8 and RC6-32 in [16]. In [16], the transition matrix can compute the expected χ^2-values on $\mathrm{lsb}_5(A_{r+2})||\mathrm{lsb}_5(C_{r+2})$ when plaintexts with $\mathrm{lsb}_5(A_0) = \mathrm{lsb}_5(C_0) = 0$ are chosen, which is denoted by `TM` in this paper. So `TM`

also gives the probability of occurence of $\mathrm{lsb}_5(A_{r+2})||\mathrm{lsb}_5(C_{r+2})$. We apply TM to compute the expected χ^2-values and the variance on $\mathrm{lsb}_3(A_{r+2})||\mathrm{lsb}_3(C_{r+2})$ when plaintexts with a fixed value of $\mathrm{lsb}_5(B_0) = \mathrm{lsb}_5(D_0)$ are chosen.

3 χ^2 Attack on RC6P

In this section, we review χ^2-attack on RC6P [5] and the success probability [10], which is computed by using the result of distinguishing attack.

Intuitively, the key recovery algorithm fixes some bits out of $\mathrm{lsb}_n(B_0)||\mathrm{lsb}_n(D_0)$, checks the χ^2-value of $lsb_3(A_r)||\mathrm{lsb}_3(C_r)$ and recovers $\mathrm{lsb}_2(S_{2r})||\mathrm{lsb}_2(S_{2r+1})$ of r-round RC6P. Let us set:
$(y_b, y_d) = (\mathrm{lsb}_3(B_{r+1}), \mathrm{lsb}_3(D_{r+1}))$, $(x_c, x_a) = (\mathrm{lsb}_5(F(A_{r+1})), \mathrm{lsb}_5(F(C_{r+1})))$, $(s_a, s_c) = (\mathrm{lsb}_2(S_{2r}), \mathrm{lsb}_2(S_{2r+1}))$ and $s = s_a||s_c$, where x_a (resp. x_c) is the rotation amounts on A_r (resp. C_r) in the r-th round.

Algorithm 2 ([5])

1. Choose a plaintext (A_0, B_0, C_0, D_0) with $(\mathtt{lsb}_5(B_0), \mathtt{lsb}_5(D_0)) = (0, 0)$ and encrypt it.
2. For each (s_a, s_c), decrypt $y_d||y_b$ with a key $0||s_a, 0||s_c$ by 1 round to $z_a||z_c$, which are denoted by a 6-bit integer $z = z_a||z_c$.
3. For each s, x_a, x_c, and z, update each array by incrementing $\mathtt{count}[s][x_a][x_c][z]$.
4. For each s, x_a, and x_c, compute $\chi^2[s][x_a][x_c]$.
5. Compute the average $ave[s]$ of $\{\chi^2[s][x_a][x_c]\}_{x_a,x_c}$ for each s and output s with the highest $ave[s]$ as $\mathtt{lsb}_2(S_{2r})||\mathtt{lsb}_2(S_{2r+1})$.

We may note that Algorithm 2 can be easily generalized to recover an e-bit key for an even e. In such a case, z is an $(e+2)$-bit number, on which χ^2-value is computed. The success probability of Algorithm 2 is derived theoretically from Theorem 4, where the success probability means the probability of recovering a correct key in Algorithm 2.

Theorem 4 ([5]). *Let $n \geq 10$ and $r \geq 4$. The success probability Ps of Algorithm 2 on r-round RC6P with 2^n plaintexts can be evaluated by using the distribution of χ^2-values in the distinguishing attack as follows,*

$$Ps = \int_{-\infty}^{\infty} f_{c[r,n]}(x) \cdot \left(\int_{-\infty}^{x} f_{w[r,n]}(u)du \right)^{2^e-1} dx, \tag{4}$$

where $f_{c[r,n]}(x)$ or $f_{w[r,n]}$ is a probability density function of distribution of χ^2-values on a correct or wrong key in Algorithm 2, given by

$$f_{c[r,n]}(x) = \phi_{(\mu_{d[r-1,n-10]}, \sigma^2_{d[r-1,n-10]}/2^{10})}(x) \tag{5}$$

or

$$f_{w[r,n]}(x) = \phi_{(\mu_{d[r+1,n-10]}, \sigma^2_{d[r+1,n-10]}/2^{10})}(x), \tag{6}$$

respectively, and $\mu_{d[r,n]}(\sigma^2_{d[r,n]})$ is mean (variance) of distribution of χ^2-values on $lsb_3(A_{r+1})||lsb_3(C_{r+1})$ of r-round RC6P with $lsb_5(B_0)||lsb_5(D_0) = 0$ by using 2^n plaintexts.

4 Success Probability of χ^2 Attack on RC6P

This section gives the theorem to compute the success probability of Algorithm 2 without any assumption of distinguishing attack.

4.1 Theoretical Mean and Variance of χ^2-values

To compute the success probability of Algorithm 2 without any experimental results of distinguishing attack, we have to compute the mean and variance, $\mu_{d[r,n]}$ and $\sigma^2_{d[r,n]}$, theoretically, that is, we have to compute θ_r. In our case, θ_r is given as

$$\theta_r = 2^6 \sum \left(P(\mathrm{lsb}_3(A_{r+1})||\mathrm{lsb}_3(C_{r+1})) - \frac{n}{2^6} \right)^2, \tag{7}$$

where the summation is over $\mathrm{lsb}_3(A_{r+1})||\mathrm{lsb}_3(C_{r+1}) \in \{0,1\}^6$ and $P(\mathrm{lsb}_3(A_{r+1})||\mathrm{lsb}_3(C_{r+1}))$ is the probability of occurrence of $\mathrm{lsb}_3(A_{r+1})||\mathrm{lsb}_3(C_{r+1})$. Thus, θ_r can be given by computing $P(\mathrm{lsb}_3(A_{r+1})||\mathrm{lsb}_3(C_{r+1}))$ and derived theoretically by `TM` in Section 2, which follows the discussion below.

Algorithm 2 is based on such a distinguishing attack that chooses $\mathrm{lsb}_5(B_0) = \mathrm{lsb}_5(D_0) = 0$ and computes the χ^2-value on $\mathrm{lsb}_3(A_{r+1})||\mathrm{lsb}_3(C_{r+1})$ which are outputs of r-round RC6P. Therefore we can apply `TM` to our distinguishing attack by assuming that (A_1, B_1, C_1, D_1) is a plaintext since $A_1 = B_0, C_1 = D_0$, and both B_1 and D_1 are random number. On the other hand, we compute the χ^2-value on $(e+2)$-bit $\mathrm{lsb}_{e/2+1}(A_{r+1})||\mathrm{lsb}_{e/2+1}(C_{r+1})$ in e-bit-key-recovery Algorithm 2, whose probability of occurrence is derived by using `TM` from the following Lemma 1.

Lemma 1. *The probability of occurrence of $lsb_{e/2+1}(A_{r+1})||lsb_{e/2+1}(C_{r+1})$, denoted by $P(lsb_{e/2+1}(A_{r+1})||lsb_{e/2+1}(C_{r+1}))$, is computed from the probability of occurrence of $lsb_5(A_{r+1})||lsb_5(C_{r+1})$ as follows*

$$P(lsb_{e/2+1}(A_{r+1})||lsb_{e/2+1}(C_{r+1})) = \sum_{i=0}^{2^\beta-1} \sum_{j=0}^{2^\beta-1} P(i||lsb_{e/2+1}(A_{r+1})||j||lsb_{e/2+1}(C_{r+1})),$$

where $\beta = 5 - (e/2+1)$ and e is an even integer from 2 to 10.

Proof. Lemma 1 holds because

$$\begin{aligned} &\mathrm{lsb}_5(A_{r+1})||\mathrm{lsb}_5(C_{r+1}) \\ &= \mathrm{msb}_\beta(\mathrm{lsb}_5(A_{r+1}))||\mathrm{lsb}_{e/2+1}(A_{r+1})||\mathrm{msb}_\beta(\mathrm{lsb}_5(C_{r+1}))||\mathrm{lsb}_{e/2+1}(C_{r+1}). \end{aligned}$$

■

We show theoretical and experimental results of mean and variance of χ^2-values of 3- or 5-round RC6P in Tables 3, respectively. Experiments are done by using 100 keys × 100 kinds texts. We see that both mean and variance of χ^2-value can be computed theoretically.

Table 3. χ^2-values of 3- or 5-round RC6P

3 rounds					5 rounds				
#texts	Theoretical		Experimental		#texts	Theoretical		Experimental	
	mean	variance	mean	variance		mean	variance	mean	variance
2^8	63.20	126.82	63.18	126.50	2^{24}	63.20	126.80	63.30	125.72
2^9	63.41	127.64	63.27	126.78	2^{25}	63.40	127.60	63.43	128.48
2^{10}	63.82	129.29	63.79	125.02	2^{26}	63.80	129.19	63.72	128.94
2^{11}	64.64	132.57	64.33	130.48	2^{27}	64.60	132.34	64.50	132.11
2^{12}	66.29	139.14	65.92	139.85	2^{28}	66.19	138.78	66.16	141.22

Table 4. Theoretical and experimental success probabilities of 4-round RC6P ($e = 4$).

# texts	2^{18}	2^{19}	2^{20}	2^{21}	2^{22}
Theoretical	0.16	0.31	0.70	0.99	1.00
Experimental	0.10	0.17	0.34	0.75	1.00

4.2 Success Probability of Algorithm 2 on RC6P

By using the theoretical mean and variance in Section 4.1, the success probability of Algorithm 2 is proved as follows.

Theorem 5. *The success probability of e-bit-key-recovery Algorithm 2 of r-round RC6P is given as follows,*

$$Ps_{rc6p,e}(n) = \int_{-\infty}^{\infty} \phi_{((k-1)+m\theta_{r-1},(2(k-1)+4m\theta_{r-1})/2^{10})}(x) \cdot \left(\int_{-\infty}^{x} \phi_{((k-1)+m\theta_{r+1},(2(k-1)+4m\theta_{r+1})/2^{10})}(u)du \right)^{2^e-1} dx, \quad (8)$$

where 2^n is the number of texts; $m = 2^{n-10}$; $k = 2^{e+2}$; $m\theta_r$ is r-round non-central parameter; and e is an even integer from 2 to 10.

Proof. Ps in Theorem 4 is derived by mean $\mu_{d[r,n]}$ and variance $\sigma^2_{d[r,n]}$ of distribution of χ^2-values, which are computed by non-central parameter from Theorem 2. On the other hand, θ_r is computed by using Lemma 1. Thus we get $Ps_{rc6p,e}(n)$. ■

Table 4 shows the success probability of Algorithm 2. According to Table 4, the theoretical estimation gives the upper bound of results. It seems rather rough upper bound. We will discuss the reason in Section 5.

4.3 Applicable Round of RC6P

By computing θ_r of each round r, we derive the number of texts to recover a correct key by Algorithm 2. We approximate Equation (8) to reduce the computation amount to get (8) for an even large e.

Theorem 6. *The sufficient condition for $Ps_{rc6p,e}(n) \geq 0.95$ is given as*

$$\widetilde{Ps}_{rc6p,e}(n) \geq 1 - \frac{1}{20(2^e - 1)}, \tag{9}$$

where

$$\widetilde{Ps}_{rc6p,e}(n) = \int_{-\infty}^{\infty} \phi_{(k-1+m\theta_{r-1},(2(k-1)+4m\theta_{r-1})/2^{10})}(x) \cdot \int_{-\infty}^{x} \phi_{(k-1+m\theta_{r+1},(2(k-1)+4m\theta_{r+1})/2^{10})}(u)du\ dx;$$

$m = 2^{n-10}$; $k = 2^{e+2}$; $m\theta_r$ is r-round non-central parameter; and e is an even integer from 2 to 10.

Proof. We show that n satisfied with Equation (9) is sufficient for $Ps_{rc6p,e}(n) \geq 0.95$. First of all, we consider the following equation

$$F(e) = \left(1 - \frac{1}{20(2^e - 1)}\right)^{2^e - 1}.$$

When $e \geq 1$, $F(e)$ is a monotonically increasing function, satisfies $F(e) \geq 0.95$ and

$$\lim_{e \to \infty} \left(1 - \frac{1}{20(2^e - 1)}\right)^{2^e - 1} \approx 0.951.$$

On the other hand, Equation (8) becomes

$$Ps_{rc6p,e}(n) = \int_{-\infty}^{\infty} \phi_{(k-1+m\theta_{r-1},(2(k-1)+4m\theta_{r-1})/2^{10})}(x) \cdot \left(\int_{-\infty}^{x} \phi_{(k-1+m\theta_{r+1},(2(k-1)+4m\theta_{r+1})/2^{10})}(u)du\right)^{2^e-1} dx$$

$$\geq \left(\int_{-\infty}^{\infty} \phi_{(k-1+m\theta_{r-1},(2(k-1)+4m\theta_{r-1})/2^{10})}(x) \cdot \int_{-\infty}^{x} \phi_{(k-1+m\theta_{r+1},(2(k-1)+4m\theta_{r+1})/2^{10})}(u)du\ dx\right)^{2^e-1}$$

Thus, if $m = 2^{n-10}$ satisfies

$$\left(\int_{-\infty}^{\infty} \phi_{(k-1+m\theta_{r-1},(2(k-1)+4m\theta_{r-1})/2^{10})}(x) \cdot \int_{-\infty}^{x} \phi_{(k-1+m\theta_{r+1},(2(k-1)+4m\theta_{r+1})/2^{10})}(u)du\ dx\right)^{2^e-1} \geq F(e),$$

then $Ps_{rc6p,e}(n) \geq 0.95$. Therefore, if n satisfies

$$\widetilde{Ps}_{rc6p,e}(n) = \int_{-\infty}^{\infty} \phi_{(k-1+m\theta_{r-1},(2(k-1)+4m\theta_{r-1})/2^{10})}(x) \cdot \int_{-\infty}^{x} \phi_{(k-1+m\theta_{r+1},(2(k-1)+4m\theta_{r+1})/2^{10})}(u)du\ dx \geq 1 - \frac{1}{20(2^e-1)},$$

then $Ps_{rc6p,e}(n) \geq 0.95$. ■

Table 5. Theoretical and estimated #texts for $Ps_{rc6p,4}(n) \geq 0.95$ or $Ps \geq 0.95$.

	Theoretical (Th.6)		Estimated (Th.4)	
round	# texts	Work ‡	# texts	Work ‡
4	$2^{20.69}$	$2^{24.69}$	†$2^{21.60}$	†$2^{25.60}$
6	$2^{36.73}$	$2^{39.67}$	$2^{37.64}$	$2^{41.64}$
8	$2^{52.76}$	$2^{56.76}$	$2^{53.68}$	$2^{57.68}$
10	$2^{68.79}$	$2^{72.79}$	$2^{69.72}$	$2^{73.72}$
12	$2^{84.81}$	$2^{88.81}$	$2^{85.76}$	$2^{89.76}$
14	$2^{100.82}$	$2^{104.82}$	$2^{101.80}$	$2^{105.80}$
16	$2^{116.83}$	$2^{119.77}$	$2^{117.84}$	$2^{121.84}$
18	$2^{132.85}$	$2^{136.85}$	$2^{133.88}$	$2^{137.88}$

† : experimental result [5]
‡ : the number of incrementing *cnt*.

Here we set $e = 4$. Table 5 shows theoretical and experimental number of texts necessary for $Ps_{rc6p,e}(n) \geq 0.95$ in each r round. From Table 5, Algorithm 2 is faster than exhaustive search for 128-bit-key RC6P with up to 16 rounds. It corresponds with the previous experimental result [5]. Our theorem estimates the number of texts necessary for recovering r-round RC6P with the success probability of more than 95% to

$$log_2(\#\text{texts}) = 8.01r - 11.63. \tag{10}$$

On the other hand, it is estimated in [5] heuristically as

$$log_2(\#\text{texts}) = 8.02r - 10.48. \tag{11}$$

We see that both estimations are pretty close each other.

4.4 Success Probability of Algorithm 2 on RC6P-8

We also demonstrate our theorem on 4-round RC6P-8 whose word size is 8-bit. Table 6 shows the theoretical and experimental results of Algorithm 2 on RC6P-8. In the same way as 4-round RC6P, we see that theoretical estimation gives the upper bound of experimental results.

Table 6. Theoretical and experimental success probability of 4-round RC6P-8 by using Algorithm 2.

# texts	Theoretical	Experimental
2^{12}	0.742	0.228
2^{13}	1.000	0.481
2^{14}	1.000	0.888
2^{15}	1.000	1.000

5 χ^2 Attack Against RC6

This section improves Algorithm 2 to a key recovery attack against RC6, Algorithm 3, and then gives the theorem that computes the success probability. We also implement Algorithm 3 on 4-round RC6-8 and demonstrate the accuracy of the theorem. Furthermore we also discuss the difference between Theorem 5 and 7 in view of accuracy.

5.1 Key Recovery Algorithm and Theoretical Success Probability

The primitive extension of Algorithm 2 to a key recovery attack on RC6 is to decrypt $y_a||y_d$ for each key candidate of s, S_{2r+2} and S_{2r+3}, which is shown in [5]. Apparently it is rather straightforward since it means that it decrypts each ciphertext by each 2^{68} key. So we improve Algorithm 2 such that it does not have to decrypt each ciphertext. Before showing the algorithm, let us use the following notation:
$\mathcal{U} = \{u \in \{0,1\}^{32} | \mathrm{msb}_5(u \times (2u+1)) = 0\}, (u_a, u_c) \in \mathcal{U} \times \mathcal{U}, t_a = A_{r+2} - u_a, t_c = C_{r+2} - u_a,$
$v = \mathrm{lsb}_5(B_0)||\mathrm{lsb}_5 D_0, z = \mathrm{lsb}_3(B_{r+2})||\mathrm{lsb}_3(D_{r+2})$.

Algorithm 3

1. Choose a plaintext (A_0, B_0, C_0, D_0) and encrypt it to $(A_{r+2}, B_{r+2}, C_{r+2}, D_{r+2})$.
3. For each (u_a, u_c), compute both t_a and t_c and update each array by incrementing count$[t_a][t_c][v][z]$.
4. For each t_a, t_c and v, compute the χ^2-value $\chi^2[t_a][t_c][v]$.
5. Compute the average $ave[t_a][t_c]$ of $\{\chi^2[t_a][t_c][v]\}_v$ for each t_a, t_c and output t_a, t_c with the highest $ave[t_a][t_c]$ as S_{2r+2}, S_{2r+3}.

Algorithm 3 computes the χ^2-value on 6-bit z, which follows the idea of Algorithm 2. Compared with [8], in which the χ^2-value is computed on 10-bit data, Algorithm 3 seems to recover a correct key efficiently.

We may note that Algorithm 3 calculates the χ^2-value on $z = \mathrm{lsb}_3(B_{r+2})||\mathrm{lsb}_3(D_{r+2})$ by using such plaintexts that make the final-round-notation 0 for each key candidate. For a correct key, this is exactly equivalent to compute the χ^2-value on $\mathrm{lsb}_3(A_r)||\mathrm{lsb}_3(C_r)$, which is output of $(r-1)$-round RC6P because

the addition keeps the χ^2-value. Thus, we succeed to skip the post-whitening and get that the probability density function of distribution of χ^2-value with a correct key in r-round RC6 is equal to $f_{c[r,n]}$ defined in Theorem 4. On the other hand, in the case of wrong keys, this is exactly equivalent to compute the χ^2-value on $lsb_3(A_{r+2})|\ lsb_3(C_{r+2})$, which is output of $(r+1)$-round RC6P. Thus, we get that the probability density function of distribution of χ^2-value with a wrong key in r-round RC6 is equal to $f_{w[r,n]}$ defined in Theorem 4. From the above discussion, we've proved the following theorem.

Theorem 7. *The success probability of Algorithm 3 on r-round RC6 is given theoretically as*

$$Ps_{rc6}(n) = \int_{-\infty}^{\infty} \phi_{(2^6-1+m\theta_{r-1},(2(2^6-1)+4m\theta_{r-1})/2^{10})}(x) \cdot \left(\int_{-\infty}^{x} \phi_{(2^6-1+m\theta_{r+1},(2(2^6-1)+4m\theta_{r+1})/2^{10})}(u)du\right)^{2^{64}-1} dx, \quad (12)$$

where 2^n is the number of texts, $m = 2^{n-20}$ and $m\theta_r$ is r-round non-central parameter.

Table 7. #texts necessary for $Ps_{rc6}(n) \geq 0.95$ (From Th.8)

r	4	6	8	10	12	14	16	18
# texts	$2^{31.06}$	$2^{47.10}$	$2^{63.13}$	$2^{79.15}$	$2^{95.17}$	$2^{111.19}$	$2^{127.20}$	$2^{143.21}$
work†	$2^{85.06}$	$2^{101.10}$	$2^{117.13}$	$2^{133.15}$	$2^{149.17}$	$2^{165.19}$	$2^{181.20}$	$2^{197.21}$

†: a time to increment of *cnt*.

We approximate Equation (12) to reduce the computation amount to get (12) in the same way as Theorem 5. Theorem 8 is pretty effective to compute n with $Ps_{rc6}(n) \geq 0.95$ since the computation of exponentiation $2^{64}-1$ on an integral part in (12) is eliminated.

Theorem 8. *The sufficient condition for $Ps_{rc6}(n) \geq 0.95$ is*

$$\widetilde{Ps}_{rc6}(n) \geq 1 - \frac{1}{20(2^{64}-1)},$$

where

$$\widetilde{Ps}_{rc6}(n) = \int_{-\infty}^{\infty} \phi_{(2^6-1+m\theta_{r-1},(2(2^6-1)+4m\theta_{r-1})/2^{10})}(x) \cdot \int_{-\infty}^{x} \phi_{(2^6-1+m\theta_{r+1},(2(2^6-1)+4m\theta_{r+1})/2^{10})}(u)du\ dx,$$

$m = 2^{n-20}$ and $m\theta_r$ is r-round non-central parameter.

Table 8. Theoretical and experimental success probability of 4-round RC6-8 (Alg. 3)

# texts	2^{17}	2^{18}	2^{19}	2^{20}
Theoretical	0.00	0.05	0.73	1.00
Experimental	0.00	0.04	0.76	1.00

Table 7 shows the necessary number of texts and work which make success probability of Algorithm 3 on RC6 95% or more. The necessary number of texts is computed by Theorem 8. "Work" means the time to increment of counter *cnt*. Note that the number of available texts is bounded by 2^{128} in Algorithm 3. Therefore, we see from Table 7 that Algorithm 3 is applicable to 192-bit-key and 256-bit-key RC6 with up to 16 rounds. Thus, our results can answer the open question of [8], that is whether or not the χ^2 attack works on RC6 with 16 rounds or more.

In [8], they estimated heuristically that 192-bit-key or 256-bit-key RC6 are broken up to 14 or 15 rounds by their key recovery algorithm, respectively. We've now proved theoretically that 192-bit-key and 256-bit-key RC6 can be broken in up to 16 rounds. In Algorithm 3, we recover both post-whitening keys at once. As a result, the number of work is #texts $\times\ 2^{27\times 2}$, and thus it works on an 128-bit-key RC6 with up to 8 rounds. But we can reduce the amount of work by recovering either post-whitening key at once to #texts $\times\ 2^{27}$. Then it works on 128-bit-key RC6 with up to 12 rounds, which will be shown in the final paper.

5.2 Success Probability of Algorithm 3 on RC6-8

We also demonstrate Theorem 7 on 4-round RC6-8. Table 8 shows the theoretical and experimental results. We see that theoretical estimation gives a pretty good approximation compared with Table 6. Let us discussion the reason. In Algorithm 2, we assume that the χ^2-values of wrong keys in r-round RC6P equals that in $(r+1)$-round RC6P to estimate $Ps_{rc6p,e}(n)$. However, this is exactly upper bound of χ^2-values of wrong keys. In the case of Algorithm 3, the χ^2-values of wrong keys in r-round RC6 are equal to that in $(r+1)$-round RC6P. Thus, we see that theoretical estimation of Theorem 7 is much better than that of Theorem 5.

6 Concluding Remarks

In this paper, we have improved the χ^2-attack on RC6P to the χ^2-attack on RC6 and proved the theorems that evaluate the success probability in both χ^2-attacks. The derived formulae can be computed efficiently and provide a theoretical analysis of the success probability in the χ^2-attack. We have also demonstrated that our theorems can estimate success probability in χ^2-attacks against 4-round RC6P, RC6P-8, and RC6-8. Furthermore we have shown theoretically that our

χ^2-attack is applicable to 192-bit-key and 256-bit-key RC6 with up to 16 rounds by using $2^{127.20}$ plaintexts.

References

1. S. Contini, R. Rivest, M. Robshaw, and Y. Yin, "The Security of the RC6 Block Cipher. v 1.0," August 20, 1998.
   Available at http://www.rsasecurity.com/rsalabs/rc6/.
2. R.J. Freund and W.J. Wilson, *Statistical Method*, Academic Press, San Diego, 1993.
3. H. Gilbert, H. Handschuh, A. Joux, and S. Vaudenay, "A Statistical Attack on RC6", *FSE 2000*, LNCS **1978**(2000), Springer-Verlag, 64–74.
4. H. Handschuh and H. Gilbert, "χ^2 Cryptanalysis of the SEAL Encryption Algorithm", *FSE '97*, LNCS **1267**(1997), Springer-Verlag, 1–12.
5. N. Isogai, T. Matsunaka, and A. Miyaji, "Optimized χ^2-attack against RC6", *ANCS 2003*, LNCS **2846**(2003), Springer-Verlag, .
6. P. Junod, "On the Complexity of Matsui's Attack", *SAC 2001*, LNCS **2259**(2001), Springer-Verlag, 199–211.
7. J. Kelsey, B. Schneier, and D. Wagner, "Mod n Cryptanalysis, with applications against RC5P and M6", *FSE '99*, LNCS **1636**(1999), Springer-Verlag, 139–155.
8. L. Knudsen and W. Meier, "Correlations in RC6 with a reduced number of rounds", *FSE 2000*, LNCS **1978**(2000), Springer-Verlag, 94–108.
9. D. Knuth, *The art of computer programming*, vol.2, Seminumerical Algorithms, 2nd ed., Addison-Wesley, Reading, Mass. 1981.
10. T. Matsunaka, A. Miyaji, and Y. Takano, "Success probability in χ^2-attacks", *ACNS 2004*, LNCS **3089**(2004), Springer-Verlag, 310-325.
11. A. Miyaji and M. Nonaka, "Cryptanalysis of the Reduced-Round RC6", *ICICS 2002*, LNCS **2513**(2002), Springer-Verlag, 480–494.
12. R. Rivest, M. Robshaw, R. Sidney, and Y. Yin, "The RC6 Block Cipher. v1.1," August 20, 1998. Available at http://www.rsasecurity.com/rsalabs/rc6/.
13. A. A. Selcuk and A. Bicak, "On probability of success in differential and linear cryptanalysis", *SCN 2002*, LNCS **2576**(2003), Springer-Verlag, 1751–185.
14. S. Vaudenay, "An Experiment on DES Statistical Cryptanalysis", *ACM-CCS '96*, ACM Press(1996), 139–147.
15. T. Shimoyama, M. Takenaka, and T. Koshiba, "Multiple linear cryptanalysis of a reduced round RC6," *FSE 2002*, LNCS **2365** (2002), Springer-Verlag, 76–88.
16. M. Takenaka, T. Shimoyama, T. Koshiba, "Theoretical Analysis of χ^2 Attack on RC6", IEICE Trans., VOL.**E87-A**, NO.1(2004), 28–35.
17. B. Ryabko, "Adaptive chi-square test and its application to some cryptographic problems", *Cryptology ePrint Archive, Report 2002/030* (2003), http://eprint.iacr.org/.

Solving Systems of Differential Equations of Addition*

(Extended Abstract)

Souradyuti Paul and Bart Preneel

Katholieke Universiteit Leuven, Dept. ESAT/COSIC,
Kasteelpark Arenberg 10,
B–3001, Leuven-Heverlee, Belgium
{Souradyuti.Paul, Bart.Preneel}@esat.kuleuven.ac.be

Abstract. Mixing addition modulo 2^n $(+)$ and exclusive-or $(\oplus)$ have a host of applications in symmetric cryptography as the operations are fast and nonlinear over GF(2). We deal with a frequently encountered equation $(x + y) \oplus ((x \oplus \alpha) + (y \oplus \beta)) = \gamma$. The difficulty of solving an arbitrary system of such equations – named *differential equations of addition* (DEA) – is an important consideration in the evaluation of the security of many ciphers against *differential attacks*. This paper shows that the satisfiability of an arbitrary set of DEA – which has so far been assumed *hard* for large n – is in the complexity class P. We also design an efficient algorithm to obtain all solutions to an arbitrary system of DEA with running time linear in the number of solutions.
Our second contribution is solving DEA in an *adaptive query model* where an equation is formed by a query (α, β) and oracle output γ. The challenge is to optimize the number of queries to solve $(x+y)\oplus((x\oplus\alpha)+(y\oplus\beta)) = \gamma$. Our algorithm solves this equation with only 3 queries in the worst case. Another algorithm solves the equation $(x+y)\oplus(x+(y\oplus\beta)) = \gamma$ with $(n-t-1)$ queries in the worst case (t is the position of the least significant '1' of x), and thus, outperforms the previous best known algorithm by Muller – presented at FSE '04 – which required $3(n-1)$ queries. Most importantly, we show that the upper bounds, for our algorithms, on the number of queries match worst case lower bounds. This, essentially, closes further research in this direction as our lower bounds are *optimal*. Finally we describe applications of our results in *differential cryptanalysis*.

1 Introduction

Addition modulo 2^n. Mixing *addition modulo* 2^n $(+)$ with other Boolean operations such as *exclusive-or* $(\oplus)$, *or* $(\vee)$ and/or *and* $(\wedge)$ is extensively used

* This work was supported in part by the Concerted Research Action (GOA) Mefisto 2000/06 and Ambiorix 2005/11 of the Flemish Government and in part by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 75–88, 2005.

in symmetric cryptography. The main motivation for including *addition mod* 2^n in cryptographic primitives is that it is a nonlinear transformation over GF(2) and the operation is extremely fast on all present day architectures. Nonlinear transformations are of paramount importance in the design of ciphers as they make functions hard to invert. Helix [9], IDEA [14], Mars [4], RC6 [20], and Twofish [21] which mix modular *addition* with *exclusive-or* are a few examples of the application of *addition*. Very recently Klimov and Shamir also used an update function for internal state, known as a T-function, where *addition* is mixed with *multiplication* and *or* in a certain fashion to achieve many useful properties of a secure stream cipher [11], [12].

Keeping with the trend of widespread use of *addition* in symmetric ciphers, there is a large body of literature that studies equations involving *addition* from many different angles. Staffelbach and Meier investigated the probability distribution of the carry for integer addition [22]. Wallén explained the linear approximations of modular addition [24]. Lipmaa and Moriai [15] investigated the equation $(x + y) \oplus ((x \oplus \alpha) + (y \oplus \beta)) = \gamma$, where α, β are the input differences and γ is the output difference, to compute many differential properties. The dual of the above equation $(x \oplus y) + ((x + \alpha) \oplus (y + \beta)) = \gamma$ was investigated for differential properties by Lipmaa *et al.* [16].

Differential Cryptanalysis (DC). Differential Cryptanalysis, introduced by Biham and Shamir [5], is one of the most powerful attacks against symmetric ciphers. There are broadly two lines of attacks based on DC. One is guessing input or output differences of a cipher with nontrivial probability. In a cipher that is secure against DC, input and output differences should behave 'pseudorandomly', so that none of them can be guessed from any known values with a nontrivial probability. This line of attack usually results in distinguishing attacks [23]. A second line of attack is much stronger but more difficult to implement than the other. It recovers secret information from known input and output differences, akin to the algebraic attacks [17]. Note that this second line of attack implies the first but the converse is not true. Therefore, provable security against DC – introduced by Lai *et al.* [14] and first implemented by Nyberg and Knudsen [18] – remained a key factor in the evaluation of the security of a cipher. However, security of many complex modern ciphers against DC is hard to evaluate because of lack of theory to evaluate the security of its components. Our target is to mount a second line of attack (i.e., to recover secret information) on the much used symmetric cipher component *addition modulo* 2^n.

Results of the Paper. There are two basic *addition* equations under DC where differences of inputs and outputs are expressed as *exclusive-or*.

$$(x + y) \oplus (x + (y \oplus \beta)) = \gamma\,, \tag{1}$$

$$(x + y) \oplus ((x \oplus \alpha) + (y \oplus \beta)) = \gamma\,. \tag{2}$$

These equations are named *differential equations addition* (DEA). While engaged in cryptanalysis of MD5, Berson noted in 1992 that, for large n, it is *hard* to analyze modular addition when differences are expressed as XOR [3]. This may

have motived the use of *addition* in conjunction with XOR in many symmetric ciphers to increase resistance against DC.

In this paper we show that the satisfiability of a randomly generated set of DEA is in the complexity class P. In other words, a *Turing machine* can show in $\mathcal{O}(n^k)$ time whether there exists a solution to an arbitrary set of DEA (n denotes the bit-length of x, y and $k > 0$ is an integer-valued constant computed from the degree of the polynomial (in n) by which the number of equations to be solved is bounded above). This result, on one hand, gives deeper insight into the behavior of addition under DC. On the other hand this leaves a cautionary note for the cryptographers to be more careful about using addition in the design. Outside cryptography, satisfiability of a system of equations has a natural appeal to many areas such as computational complexity, combinatorics, circuit optimization and computer algebra (remember the most famous NP-Complete satisfiability problem: Boolean formula satisfiability [6]). For example, if a large system of DEA is NOT satisfiable then the whole circuit representing the system of DEA can be safely removed to optimize the circuit complexity. Going beyond the satisfiability problem, we also give an efficient algorithm to compute all the solutions to a randomly generated system of DEA with running time linear in the number of solutions. Another subtle but a noteworthy aspect of our work is the departure from the traditional technique for solving multivariate polynomial equations over GF(2) [2]. We heavily benefit from certain properties of DEA and solve such systems combinatorially.

Next, we extend our work to solve DEA in a crypto-friendly *adaptive query model*. The aim is to *minimize* the search space for the secret (x, y) using a *minimum* number of adaptive queries (α, β). Such an optimization problem – typically used to reduce data complexity of chosen plaintext attacks – for (1) has already been tackled by Muller [17]. But an optimal solution has been elusive until now. We achieve optimal solutions for both of the equations. We show that a worst case lower bound on the number of queries $(0, \beta)$ to solve (1) is $(n-t-1)$ where $(n-t) > 1$ with t being the bit-position of the least significant '1' of x. A worst case lower bound on the number of queries (α, β) to solve (2) is 3 for $n > 2$. Most importantly, for solving the above equations we also design algorithms whose upper bounds on the number of queries match worst case lower bounds. Note that our algorithm outperforms the previous best known algorithm by Muller to solve (1) – presented at FSE '04 – which required $3(n-1)$ queries [17]. Over and above, our results essentially close further investigation in this particular direction as the equations are solved with an *optimal* number of queries in the worst case. It is particularly interesting to note that, for (2), although the number of all queries grows exponentially with the input size n, an optimal lower bound to solve (2) is 3 for all $n > 2$, i.e., constant asymptotically.

Our results on *modular addition* have the potential to be used either directly in the cryptanalysis of ciphers that include this special component or to facilitate cryptanalysis of several modern ciphers (e.g., mixing addition with multiplication). We show that, with a maximum of only 3 adaptively chosen queries, the search space of the secret of modular addition against DC can be reduced from

2^{2n} to only 4 for all $n \geq 1$. We used our results to cryptanalyze a recently proposed cipher Helix [9] which was a candidate for consideration in the 802.11i standard. We are successful in reducing the data complexity of a DC attack on the cipher by a factor of 3 in the worst case (a factor of 46.5 in the best case) [17]. In addition, using our algorithm to solve DEA, as discussed above we are able to compute all the *differential properties of addition* by investigating a single equation (note that the *differential properties of addition* have been independently found by Lipmaa and Moriai using a different technique [15]).

1.1 Notation and Model of Computation

The purpose of the paper is to solve (1) and (2) for (x, y) using triples (α, β, γ) where $x, y, \alpha, \beta, \gamma \in \mathbb{Z}_2^n$. The ith bit of an n-bit integer l is denoted by l_i (l_0 denotes the least significant bit or the 0th bit of l). The operation *addition modulo* 2^n over $\mathbb{Z}_{2^n}$ can be viewed as a binary operation over $\mathbb{Z}_2^n$ (we denote this operation by '+') using the bijection that maps $(l_{n-1}, \cdots, l_0) \in \mathbb{Z}_2^n$ to $l_{n-1}2^{n-1} + \cdots + l_0 2^0 \in \mathbb{Z}_{2^n}$. Therefore '+' is a function '+': $\mathbb{Z}_2^n \times \mathbb{Z}_2^n \rightarrow \mathbb{Z}_2^n$. The symbols '$\oplus$' and '$\wedge$' denote the operations *bit-wise exclusive-or* and *bit-wise and* of two n-bit integers respectively. We will denote $a \wedge b$ by ab. Throughout the paper, $[p, q]$ denotes a set containing all integers between the integers p and q including both of them. Unless otherwise stated, n denotes a positive integer. The size of a set S is denoted by $|S|$.

The algorithms, described in this paper, can be implemented on a generic one-processor *Random Access Machine* (RAM) (i.e., instructions are executed sequentially) whose memory is composed of an unbounded sequence of registers each capable of containing an integer. Typically, *RAM* instructions consist of simple arithmetic operations (addition and bitwise XOR in our case), storing, addressing (direct and indirect) and branching, each of which is a constant time operation. However, the choice of *RAM* instructions is relatively less important because algorithms based on two reasonable sets of instructions will have the same asymptotic complexity. A detailed analysis of *RAM* can be found in [1], [8]. It can be shown that a polynomial-time solvable problem on a *RAM* is also polynomial-time solvable on a *Turing Machine* and vice versa.

2 Solving an Arbitrary System of DEA

Our aim is to solve for (x, y) from the following set of differential equations of addition over $\mathbb{Z}_2^n$,

$$(x + y) \oplus ((x \oplus \alpha[k]) + (y \oplus \beta[k])) = \gamma[k]\,, \qquad k = 1, 2 \cdots m\,. \tag{3}$$

We notice that the ith bit of $\gamma[k]$ is a function of the least $(i + 1)$ bits of x, y, $\alpha[k]$ and $\beta[k]$. More formally,

$$\gamma[k]_i = F_i(x_0, \cdots, x_i, y_0, \cdots, y_i, \alpha[k]_0, \cdots, \alpha[k]_i, \beta[k]_0, \cdots, \beta[k]_i)\,. \tag{4}$$

Note that, from a system of m differential equations of addition, a total of mn multivariate polynomial equations over GF(2) can be formed by ranging (k, i) through all values in (4).

Plenty of research has been undertaken to design efficient ways to solve randomly generated multivariate polynomial equations. The classical Buchberger's Algorithm for generating Gröbner bases [2] and its variants [7] are some of them. This problem is NP-complete (NPC) over GF(2). Many other techniques such as relinearization [13] have been proposed to solve a special case of overdefined systems of multivariate polynomial equations. Note that, in our case, the number of unknowns and equations are $2n$ and mn respectively (if $m > 2$ then the system of equations is overdefined). However, taking full advantage of the specific nature of the *differential equations of addition*, we shall use a combinatorial technique to prove that, although the satisfiability of an arbitrary multivariate polynomial equation over GF(2) is NP-complete, this special cryptographically important subclass of equations is in the complexity class P (see [6], [10] for definitions of NP, P, NPC). Finally, we also derive all the solutions to a system of such equations.

2.1 Computing the *Character Set* and the *Useful Set*

From (3) we construct $A = \{(\alpha[k],\, \beta[k],\, \gamma[k]) \,|\, k \in [1,\, m]\}$ assuming $(\alpha[k],\ \beta[k],\ \gamma[k])$'s are all distinct.[1] We call A the *character set*. Our first step is to transform the system of equations defined in (3) into a new set of equations over $\mathbb{Z}_2^n$ as defined below,

$$(x + y) \oplus ((x \oplus \alpha[k]) + (y \oplus \beta[k])) \oplus \alpha[k] \oplus \beta[k] = \tilde{\gamma}[k]\,, \qquad k = 1,\, 2 \cdots m\,; \tag{5}$$

where $\tilde{\gamma}[k] = \gamma[k] \oplus \alpha[k] \oplus \beta[k]$. Now, we construct $\tilde{A}$,

$$\tilde{A} = \{(\alpha,\, \beta,\, \tilde{\gamma} = \alpha \oplus \beta \oplus \gamma) \,|\, (\alpha,\, \beta,\, \gamma) \in A\}\,. \tag{6}$$

We call $\tilde{A}$ the *useful set*. Let all the solutions for (3) and (5) be contained in the sets A-satisfiable and $\tilde{A}$-consistent respectively. It is trivial to show that

$$A\text{-satisfiable} = \tilde{A}\text{-consistent}\,. \tag{7}$$

Our aim is to compute $\tilde{A}$-consistent from $\tilde{A}$.

2.2 Precomputation

Take an arbitrary element $(\alpha,\, \beta,\, \tilde{\gamma}) \in \tilde{A}$ $(n > 1)$. Observe that $\tilde{\gamma}_{i+1}$ can be computed using $x_i,\, y_i,\, c_i,\, \alpha_i,\, \beta_i, \tilde{\gamma}_i,\ \forall\, i \in [0,\, n-2]$, from the following three equations

$$\tilde{\gamma}_{i+1} = c_{i+1} \oplus \tilde{c}_{i+1},\ c_{i+1} = x_i y_i \oplus x_i c_i \oplus y_i c_i,\ \tilde{c}_{i+1} = \tilde{x_i}\tilde{y_i} \oplus \tilde{x_i}\tilde{c_i} \oplus \tilde{y_i}\tilde{c_i}$$

where c_i is the carry at the ith position of $(x + y)$, $\tilde{x_i} = x_i \oplus \alpha_i$, $\tilde{y_i} = y_i \oplus \beta_i$ and $\tilde{c_i} = c_i \oplus \tilde{\gamma}_i$. Table 1 lists the values of $\tilde{\gamma}_{i+1}$ as computed from all values of $x_i,\, y_i,\, c_i,\, \alpha_i,\, \beta_i,\, \tilde{\gamma}_i$.

[1] This can be obtained by taking one of the identical equations in (3).

Table 1. The values of $\tilde{\gamma}_{i+1}$ corresponding to the values of x_i, y_i, c_i, α_i, β_i, $\tilde{\gamma}_i$. A row and a column are denoted by R(l) and Col(k)

(x_i, y_i, c_i)	$(\alpha_i, \beta_i, \tilde{\gamma}_i)$								
	(0,0,0)	(0,0,1)	(0,1,0)	(0,1,1)	(1,0,0)	(1,0,1)	(1,1,0)	(1,1,1)	R(0)
(0,0,0)	0	0	0	1	0	1	1	1	R(1)
(1,1,1)									
(0,0,1)	0	0	1	0	1	0	1	1	R(2)
(1,1,0)									
(0,1,0)	0	1	0	0	1	1	0	1	R(3)
(1,0,1)									
(1,0,0)	0	1	1	1	0	0	0	1	R(4)
(0,1,1)									
Col(0)	Col(1)	Col(2)	Col(3)	Col(4)	Col(5)	Col(6)	Col(7)	Col(8)	

2.3 Computing G_i, $S_{i,0}$ and $S_{i,1}$ from the *Useful Set* $\tilde{A}$

We now determine an important quantity, denoted by G_i, for a nonempty *useful set* $\tilde{A}$. In G_i, we store the ith and $(i+1)$th bits of $\tilde{\gamma}$ and the ith bit of α and β for all $(\alpha, \beta, \tilde{\gamma}) \in \tilde{A}$. We call G_i the ith core of the *useful set* $\tilde{A}$. More formally (suppose $n > 1$),

$$G_i = \{(\alpha_i, \beta_i, \tilde{\gamma}_i, \tilde{\gamma}_{i+1}) \,|\, (\alpha, \beta, \tilde{\gamma}) \in \tilde{A}\}, \quad i \in [0, n-2]\,. \tag{8}$$

In the subsequent discussion we will often use the expression "$G_i \Rightarrow (x_i, y_i, c_i)$". Let $|G_i| = g$. Take an element $(\alpha_i, \beta_i, \tilde{\gamma}_i, \tilde{\gamma}_{i+1}) \in G_i$. In Table 1, find the row(s) of the fourth coordinate $\tilde{\gamma}_{i+1}$ in the column specified by the first three coordinates $(\alpha_i, \beta_i, \tilde{\gamma}_i)$ in R(0) and put them in set F_{i1}. Find $F_{i1}, \cdots F_{ig}$ for all g elements of G_i. Let $F_i = \bigcap_j F_{ij}$ and R(x)$\in F_i$. If (x_i, y_i, c_i) is in Col(0)$\times$R(x) then we say $G_i \Rightarrow (x_i, y_i, c_i)$. If $F_i = \phi$ then no such (x_i, y_i, c_i) exists. We compute $S_{i,j} = \{(x_i, y_i) \,|\, G_i \Rightarrow (x_i, y_i, c_i = j)\}$. See [19] for an example.

We assume $m = |\tilde{A}| = \mathcal{O}(n^l)$ for some nonnegative integer l. Then the time and memory to compute all G_i's and $S_{i,j}$'s are $\mathcal{O}(n^k)$ each because the size of Table 1 and $|G_i|$ are $\mathcal{O}(1)$ each ($k = l + 1$). Now, we show a relation between $S_{i,0}$ and $S_{i,1}$ that will be used to obtain several results.

Proposition 1. *For all nonempty useful set* $\tilde{A}$ *and all* $n > 1$, $|S_{i,0}| = |S_{i,1}|$ $\forall\, i \in [0, n-2]$.

Proof. The proof follows from Table 1. □

We set,

$$|S_{i,0}| = |S_{i,1}| = S_i \quad \forall\, i \in [0, n-2]. \tag{9}$$

2.4 Satisfiability of DEA is in P

In this section, we deal with a decision problem: does there exist a solution for an arbitrary set of differential equations of addition, i.e., is a system of DEA satisfiable?

We have already seen how to compute the *character set* A, the *useful set* $\tilde{A}$, the core G_i's and S_i's from a system of DEA in $\mathcal{O}(n^k)$ (see Sect. 2.3). We now formulate $|\tilde{A}$-consistent$|$ (i.e., the number of solutions) in the following proposition which will later answer our satisfiability question.

Proposition 2. *Let the useful set* $\tilde{A} \neq \phi$ *and* S *denote* $|\tilde{A}$*-consistent*$|$*. Then,*

$$S = \begin{cases} 0 & \text{if } \tilde{\gamma}_0 = 1 \text{ for some } (\alpha, \beta, \tilde{\gamma}) \in \tilde{A}, \\ 4 \cdot \prod_{i=0}^{n-2} S_i & \text{if } \tilde{\gamma}_0 = 0, \forall (\alpha, \beta, \tilde{\gamma}) \in \tilde{A} \text{ and } n > 1, \\ 4 & \text{if } \tilde{\gamma}_0 = 0, \forall (\alpha, \beta, \tilde{\gamma}) \in \tilde{A} \text{ and } n = 1. \end{cases}$$

The S_i*'s are defined in (9).*

Proof. A detailed proof is provided in the full version of the paper [19]. □

Using Proposition 2, we now answer the question of satisfiability of DEA in the following claim.

Claim. *(i) If* $\tilde{\gamma}_0 = 1$ *for some* $(\alpha, \beta, \tilde{\gamma}) \in \tilde{A}$*, then the set of DEA is NOT satisfiable. (ii) If* $\tilde{\gamma}_0 = 0, \forall (\alpha, \beta, \tilde{\gamma}) \in \tilde{A}$ *and* $n > 1$*, then the set of DEA is satisfiable if and only if* $S_i \neq 0 \ \forall i \in [0, n-2]$*. (iii) If* $\tilde{\gamma}_0 = 0, \forall (\alpha, \beta, \tilde{\gamma}) \in \tilde{A}$ *and* $n = 1$*, then the set of DEA is satisfiable.*

Verification of *(i)*, *(ii)* and *(iii)* take time $\mathcal{O}(1)$, $\Theta(n)$ and $\mathcal{O}(1)$ respectively. Therefore, the overall time to decide whether a system of DEA is satisfiable is $\mathcal{O}(n^k) + \mathcal{O}(1) + \Theta(n) + \mathcal{O}(1) = \mathcal{O}(n^k)$. Thus the satisfiability of DEA is in P.

2.5 Computing All the Solutions to a System of DEA

Now the only part left unanswered is how to actually compute $\tilde{A}$-consistent, i.e., to extract all the solutions of a system of DEA which is satisfiable. Note, if $n = 1$ then $\tilde{A}$-consistent comprises all 4 values of (x, y). The G_i's can be computed from the *useful set* $\tilde{A}$ in $\mathcal{O}(n^k)$ (see Sect. 2.3). Now we compute an intermediate parameter $L_i = \{(x_i, y_i, c_i) \,|\, G_i \Rightarrow (x_i, y_i, c_i)\}$ for all $i \in [0, n-2]$ (note that the L_i's are different from the F_i's which have been computed in Sect. 2.3). Computation of the L_i's takes time and memory each $\Theta(n)$. We call L_i the ith bit solution. Algorithm 1 computes $\tilde{A}$-consistent from the L_i's ($n > 1$).

The idea of the algorithm is to collect in M all $(x, y) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n$ such that $G_i \Rightarrow (x_i, y_i, c_i), \forall i \in [0, n-2]$. The following theorem is the heart of the argument to prove that M is essentially $\tilde{A}$-consistent.

Algorithm 1 Compute all the solutions to a system of satisfiable DEA

Input: $L_i, \forall i \in [0,\, n-2]$
Output: $\tilde{A}$-consistent
1: Find $M = \{((x_{n-1},\, x_{n-2},\, \cdots,\, x_0),\, (y_{n-1},\, y_{n-2},\, \cdots,\, y_0)) \,|\, (x_{n-1},\, y_{n-1}) \in \mathbb{Z}_2^2,$
$(x_i,\, y_i,\, c_i) \in L_i,\, i \in [0,\, n-2],\, c_0 = 0, c_{i+1} = x_i y_i \oplus x_i c_i \oplus y_i c_i\}$.
2: Return (M).

Theorem 1. *Let the useful set $\tilde{A} \neq \phi$ and $n > 1$. The following two statements are equivalent.*
1. $(x,\, y) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n$ *is such that* $G_i \Rightarrow (x_i,\, y_i,\, c_i),\, \forall\, i \in [0,\, n-2]$.
2. $(x,\, y)$ *is* $\tilde{A}$*-consistent.*

Proof. From the construction of G_i, it can be shown that $1 \Leftrightarrow 2$. □

Time and Memory. Algorithm 1 takes time $\Theta(S)$ and memory $\Theta(n \cdot S)$ where S is the number of solutions (an explicit construction of M from the L_i's and its complexity analysis are presented in [19]).

3 Solving DEA in the Adaptive Query Model

In this section, we solve the equations

$$(x + y) \oplus ((x \oplus \alpha) + (y \oplus \beta)) = \gamma, \tag{10}$$

$$(x + y) \oplus (x + (y \oplus \beta)) = \gamma \tag{11}$$

separately in an *adaptive query model*. Solving (10) in *adaptive query model* means solving a set of 2^{2n} equations generated by ranging $(\alpha,\, \beta)$ with the corresponding γ. The number of solutions satisfying 2^{2n} equations is less than that of any subset of the equations. Therefore, solving these 2^{2n} equations reduces the search space of the secret $(x,\, y)$ to the minimum. This fact is the major motivation for dealing with this problem. The task of a computationally unbounded adversary is to select a subset A of all equations such that the solutions to the chosen subset A are the same as that of the entire 2^{2n} equations. The target of the adversary is to minimize $|A|$. A similar optimization problem can be asked of (11) where the number of equations is 2^n. Such an optimization problem for (11) has already been tackled by Muller [17] in cryptanalysis of the Helix cipher but an optimal solution has still been elusive. We reach optimal solutions for both equations.

3.1 The Power of the Adversary

The power of an adversary that solves (10) is defined as follows.

1. An adversary has unrestricted computational power and an infinite amount of memory.

2. An adversary can *only* submit queries $(\alpha, \beta) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n$ to an honest oracle[2] which computes γ using fixed unknown $(x, y) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n$ in (10) and returns the value to the adversary. We will often refer to that fixed (x, y) as the *seed* of the oracle.

Such an oracle with seed (x, y) is viewed as a mapping $O_{xy} : \mathbb{Z}_2^n \times \mathbb{Z}_2^n \to \mathbb{Z}_2^n$ and defined by

$$O_{xy} = \{(\alpha, \beta, \gamma) \,|\, (\alpha, \beta) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n, \gamma = (x + y) \oplus ((x \oplus \alpha) + (y \oplus \beta))\}\,. \quad (12)$$

An adversarial model, similar to the one described above for (10), can be constructed for (11) by setting $(\alpha, \beta) \in \{0\}^n \times \mathbb{Z}_2^n$ and the mapping $O_{xy} : \{0\}^n \times \mathbb{Z}_2^n \to \mathbb{Z}_2^n$.

The model described above represents a practical adaptively chosen message attack scenario where the adversary makes adaptive queries to an oracle. Based on the replies from the oracle, the adversary computes one or more unknown parameters.

3.2 The Task

O_{xy}, defined in (12), generates a family of mappings $\mathcal{F} = \{O_{xy} \,|\, (x, y) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n\}$. Note that, if $D \in \mathcal{F}$ then $|D| = 2^{2n}$ for (10). Therefore, $D \in \mathcal{F}$ is the *character set* with the number of equations $m = 2^{2n}$ (see Sect. 2.1). Our aim is to find all (x, y) satisfying these 2^{2n} equations, i.e., to compute D-satisfiable from a subset of the *character set* D. If we deal with (11) then $|D| = 2^n$.

An Equivalent Task. From the *character set* D one can compute the *useful set* $\tilde{D}$ using (6). Therefore, the task is equivalent to the determination of $\tilde{D}$-consistent from a subset of the *useful set* $\tilde{D}$. We call D and $\tilde{D}$ the *total character set* and the *total useful set* as their sizes are maximal and they are generated from a satisfiable set 2^{2n} DEA (because we assumed the oracle to be honest). Note that there is a bijection between D and $\tilde{D}$.

Adjusting the Oracle Output. If the oracle outputs γ on query (α, β), we shall consider the oracle output to be $\tilde{\gamma} = \alpha \oplus \beta \oplus \gamma$ for the sake of simplicity in the subsequent discussions.

Rules of the Game. Now we lay down the rules followed by the adversary to determine the set $\tilde{D}$-consistent that, in turn, gives the essence of the whole problem.

1. The adversary starts with no information about x and y except their size n.

2. The adversary settles on a strategy (i.e., a deterministic algorithm) which is publicly known. Using the strategy, the adversary computes queries adaptively, i.e., based on the previous queries and the corresponding oracle outputs the next query is determined.

[2] An honest oracle correctly computes γ and returns it to the adversary.

3. The game stops the moment the adversary constructs $\tilde{D}$-consistent. The adversary fails if she is unable to compute $\tilde{D}$-consistent for some $(x, y) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n$.

We search for an algorithm that determines $\tilde{D}$-consistent for all $(x, y) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n$. Furthermore, there is an additional requirement that, in the worst case of (x, y), the number of queries required by the algorithm is the minimum . We shall elaborate on the meaning of *worst case* in Sect. 3.4 which focuses on worst case lower bounds on the number of queries.

3.3 Number of Solutions

In this section we are interested to determine the number of solutions of (10) and (11) in the adaptive query model. We have already developed a framework where the set of all solutions in the adaptive query model is denoted by $\tilde{D}$-consistent where $\tilde{D}$ is the *total useful set*. Therefore, formally, our effort will be directed to formulate $|\tilde{D}$-consistent$|$. We will see in Theorem 4 that, for (11), $|\tilde{D}$-consistent$|$ depends on the least significant '1' of x. However, for (10), $|\tilde{D}$-consistent$| = 4, \forall (x, y) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n$. We shall use these results in Theorem 4 and 5 of Sect. 3.4, to obtain lower bounds on the number of queries to compute $\tilde{D}$-consistent and in Sect. 3.5, to prove the correctness of our optimal algorithms. The proofs of the following two theorems are provided in [19].

Theorem 2. *Let the position of the least significant '1' of x in the equation*

$$(x + y) \oplus (x + (y \oplus \beta)) = \gamma$$

be t and $x, y, \beta, \gamma \in \mathbb{Z}_2^n$. Let the total useful set $\tilde{D}$ *be given. Then $|\tilde{D}$-consistent$|$ is (i) 2^{t+3} if $n - 2 \geq t \geq 0$, (ii) 2^{n+1} otherwise.*

Theorem 3. *Let the* total useful set $\tilde{D}$ *be given for the equation*

$$(x + y) \oplus ((x \oplus \alpha) + (y \oplus \beta)) = \gamma$$

with $x, y, \alpha, \beta, \gamma \in \mathbb{Z}_2^n$. Then $|\tilde{D}$-consistent$|$=4.

3.4 Worst Case Lower Bounds on the Number of Queries

Our target is to design an algorithm (for (10) or (11)) which computes $\tilde{D}$-consistent for all *seeds* $(x, y) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n$ with adaptive queries. For such an algorithm, the number of required queries may vary with the choice of (x, y). In this section we concentrate on a lower bound on the number of queries in the worst case of (x, y) under the "rules of the game" stated in Sect. 3.2. The significance of the lower bound is that there exists no algorithm that requires less queries in the worst case than the obtained lower bound.

We already noticed that more queries tend to reduce the search space of the secret (x, y). In our formal framework, if $A \subseteq B \subseteq \tilde{D}$ then $\tilde{D}$-consistent $\subseteq B$-consistent $\subseteq A$-consistent. This implies that $|\tilde{D}\text{-consistent}| \leq |B\text{-consistent}| \leq |A\text{-consistent}|$. Note that our algorithm constructs $A \subseteq \tilde{D}, \forall(x, y) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n$, using the submitted queries and the corresponding outputs such that $|\tilde{D}\text{-consistent}| = |A\text{-consistent}|$. The algorithm fails if $|\tilde{D}\text{-consistent}| < |A\text{-consistent}|$, for some $(x, y) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n$. In the following theorems, we will use the condition – $|A$-consistent$|$ cannot be *strictly* greater than $|\tilde{D}$-consistent$|$ – to compute worst case lower bounds on the number of queries (note that we have already obtained formulas for $|\tilde{D}$-consistent$|$ in the previous section).

In the following theorem, we partition the entire seed space $\mathbb{Z}_2^n \times \mathbb{Z}_2^n$ and compute a worst case lower bound for each partition. Note that a lower bound (say, l) for any partition shows that, for any algorithm that computes $\tilde{D}$-consistent $\forall(x, y) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n$, there exists at least one seed in that particular partition which requires at least l queries.

Theorem 4. *A lower bound on the number of queries* $(0, \beta)$ *to solve*

$$(x + y) \oplus (x + (y \oplus \beta)) = \gamma$$

in the worst case of (x, y) *is (i) 0 if* $n = 1$*, (ii) 1 if* $x = 0$ *and* $n > 1$*, (iii) 1 if* $n = 1 + t$ *with* $t > 0$*, (iv)* $(n - t - 1)$ *if* $n - 2 \geq t \geq 0$*, where* n *is the bit-length of* x*,* y *and* t *is the position of the least significant '1' of* x*.*

Proof. The basic idea of the proof is to show that, in each partition (note that the partition is according to t and in each partition there are many possible $(x,\ y)$'s), there exists at least one pair of $(x,\ y)$ such that, if the number of queries for that $(x,\ y)$ is less than the lower bound then we reach an impossible condition where the size of the solution set is greater than $|\tilde{D}$-consistent$|$. This is shown using Theorem 2 and Table 1. The detailed and formal proof is in the full version of the paper [19]. □

Theorem 5. *A worst case lower bound on the number of queries* (α, β) *to solve*

$$(x + y) \oplus ((x \oplus \alpha) + (y \oplus \beta)) = \gamma$$

is (i) 3 if $n > 2$*, (ii) 2 if* $n = 2$*, (iii) 0 if* $n = 1$*.*

Proof. To prove *(i)*, we consider all possible *adaptively chosen* two queries by all algorithms and find that, in each case, an oracle can always select a seed such that the number of solutions obtained from the outputs exceeds 4 which is impossible by Theorem 3. A detailed analysis is given in [19]. The proofs for *(ii)* and *(iii)* are trivial. □

3.5 Optimal Algorithms

In this section, we concentrate on designing algorithms that solve (11) and (10) in the adaptive query model. In the formal framework, if the oracle is seeded with

an unknown (x, y) then there is always a *total useful set* $\tilde{D}$ that the unknown seed (x, y) generates. For a particular *total useful set* $\tilde{D}$, there exists a set $\tilde{D}$-consistent containing all values of (x, y). Our algorithm makes adaptive queries (α, β) to the oracle – which is already seeded with a fixed (x, y) – and the oracle returns $\tilde{\gamma}$. The task of the algorithm is to compute $\tilde{D}$-consistent using oracle outputs $\tilde{\gamma}$, for all seeds $(x, y) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n$. The algorithm is *optimal* if the number of queries in the worst case (i.e., the upper bound) matches the lower bound derived in the relevant theorem (Theorem 4 or 5). We refer the readers to the full version of the paper [19] for optimal algorithms and their complexity analysis. For each of the algorithms (one for each equation), the memory and the time are $\Theta(n)$ and $\mathcal{O}(n)$ respectively.

4 Cryptographic Applications

Weakness of modular addition under DC. The fact that an arbitrary system of DEA is in the complexity class P, shows a major differential weakness of modular addition which is one of the most used block cipher components. The weakness is more alarming because, with a maximum of only 3 adaptively chosen queries the search space of the secret can be reduced from 2^{2n} to only 4 for all $n \geq 1$. It is, however, not known at this moment, how much influence this weakness has on many modern symmetric ciphers that mix other nonlinear operations with addition. Still, a recommendation for a deeper analysis of ciphers mixing XOR and modular addition in the light of our investigation is well justified.

Cryptanalysis of Helix. Helix, proposed by Ferguson *et al.* [9], is a stream cipher with a combined MAC functionality. The primitive uses combination of addition and XOR to generate pseudorandom bits. Recently a differential attack was found against Helix by Muller [17]. He solved the equation $(x + y) \oplus (x + (y \oplus \beta)) = \gamma$ many times to recover secret information (x, y) using β and the corresponding γ. Every time β corresponds to a chosen plaintext. His algorithm uses $3(n - 1)$ queries every time. Therefore, the most natural challenge, from an algorithmic point of view, is to reduce the number of queries and if possible to attain an optimality. For the Helix output word $n = 32$ bits, 93 queries were needed whereas our optimal algorithm (described in [19]) takes *at most* 31 queries if the position of the least significant '1' of x (denoted by t) is zero. Note that, if $t > 0$ then the number of queries is less. However, the most important fact is that the number of queries cannot be further reduced in the worst case as our algorithm is worst case optimal. This fact can be straightaway used to reduce the data complexity of that particular attack on Helix cipher by, *at least*, a factor of 3. However, in the best case, there exists seed (x, y), $\forall\, t \in [0, n - 3]$, for which (11) can be solved by our optimal algorithm with only 2 queries and the improvement in such a case is a factor of 46.5.

Computing Differential Properties of Addition. It is easy to show that our algorithm to solve a randomly generated set DEA is robust enough to com-

pute all differential properties of addition (e.g., maximal differential, impossible differential, etc.) In such a case we need to consider a single differential equation of addition instead of many and then compute the solutions to it. For example, if the single equation is NOT satisfiable then this is an impossible differential. However, the differential properties of addition has been independently studied in depth by Lipmaa and Moriai using a different approach [15].

5 Conclusion and Further Research

The results of the paper have impacts on both theory and practice. We have shown the importance of solving DEA from both a theoretical and a practical point of view. The paper seals any further search to improve lower bounds on the number of queries for solving DEA with adaptive queries. Although the total number of queries grows exponentially, an optimal lower bound is linear for one of them and constant for the other. Moreover, our algorithm reduces the number of queries of the previous best known algorithm and our results improve the data complexity of an attack on Helix cipher. Our work leaves room for further research, particularly, in the direction of solving DEA with nonadaptive queries. Last but not the least, our solution techniques motivate further research to solve more complex equations that mix *modular addition*, *exclusive-or*, *modular multiplication* and *T-functions*.

6 Acknowledgments

We are grateful to Mridul Nandi of the Indian Statistical Institute for encouraging us to do this work and for his critical remarks on many technical results. We also thank Taizo Shirai of Sony Incorporation, Japan, Sankardas Roy of George Mason University, USA and Jongsung Kim of Katholieke Universiteit Leuven, Belgium for numerous useful discussions. Thanks are also due to the anonymous reviewers of ACISP 2005 for their constructive comments.

References

1. A. Aho, J. Hopcroft, J. Ullman, "The Design and Analysis of Computer Algorithms," Addison-Wesley, 1974.
2. I. A. Ajwa, Z. Liu, P. S. Wang, "Gröbner Bases Algorithm," *ICM Technical Report*, February 1995, Available Online at `http://icm.mcs.kent.edu/reports/1995/gb.pdf`.
3. T. A. Berson, "Differential Cryptanalysis Mod 2^{32} with Applications to MD5," *Eurocrypt 1992* (R. A. Rueppel, ed.), vol. 658 of *LNCS*, pp. 71-80, Springer-Verlag, 1993.
4. C. Burwick, D. Coppersmith, E. D'Avignon, Y. Gennaro, S. Halevi, C. Jutla, S. M. Matyas Jr., L. O'Connor, M. Peyravian, D. Safford and N. Zunic, "MARS – A Candidate Cipher for AES," Available Online at `http://www.research.ibm.com/security/mars.html`, June 1998.
5. E. Biham, A. Shamir, "Differential Cryptanalysis of DES-like Cryptosystems," *Crypto '90* (A. Menezes, S. A. Vanstone, eds.), vol. 537 of *LNCS*, pp. 2-21, Springer-Verlag, 1991.

6. T. H. Cormen, C. E. Leiserson, R. L. Rivest, *"Introduction to Algorithms,"* MIT Press.
7. J. Faugère, "A new effecient algorithm for computing Gröbner bases (F_4)," *Journal of Pure and Applied Algebra*, vol. 139, pp. 61-88, 1999, Available Online at `http://www.elsevier.com/locate/jpaa`.
8. R. Floyd, R. Beigel, "The Language of Machines,"W. H. Freeman, 1994.
9. N. Ferguson, D. Whiting, B. Schneier, J. Kelsey, S. Lucks, T. Kohno, "Helix: Fast Encryption and Authentication in a Single Cryptographic Primitive," *Fast Software Encryption 2003* (T. Johansson, ed.), vol. 2887 of *LNCS*, pp. 330-346, Springer-Verlag, 2003.
10. J. E. Hopcroft, R. Motwani, J. D. Ullman, *''Introduction to Automata Theory, Languages and Computation,"* Second Edition, Pearson Education, 2004.
11. A. Klimov, A. Shamir, "Cryptographic Applications of T-Functions," *Selected Areas in Cryptography 2003* (M. Matsui, R. J. Zuccherato, eds.), vol. 3006 of *LNCS*, pp. 248-261, Springer-Verlag, 2004.
12. A. Klimov, A. Shamir, "New Cryptographic Primitives Based on Multiword T-Functions," *Fast Software Encryption 2004* (B. Roy, W. Meier, eds.), vol. 3017 of *LNCS*, pp. 1-15, Springer-Verlag, 2004.
13. A. Kipnis, A. Shamir, "Cryptanalysis of the HFE Public Key Cryptosystems by Relinearization," *Crypto 1999* (M. Wiener, ed.), vol. 1666 of *LNCS*, pp. 19-30, Springer-Verlag, 1999.
14. X. Lai, J. L. Massey, S. Murphy, "Markov Ciphers and Differential Cryptanalysis," *Eurocrypt '91* (W. Davis, ed.), vol. 547 of *LNCS*, pp. 17-38, Springer-Verlag, 1991.
15. H. Lipmaa, S. Moriai, "Efficient Algorithms for Computing Differential Properties of Addition," *FSE 2001* (M. Matsui, ed.), vol. 2355 of *LNCS*, pp. 336-350, Springer-Verlag, 2002.
16. L. Lipmaa, J. Wallén, P. Dumas, "On the Additive Differential Probability of Exclusive-Or," *Fast Software Encryption 2004* (B. Roy, W. Meier, eds.), vol. 3017 of *LNCS*, pp. 317-331, Springer-Verlag, 2004.
17. F. Muller, "Differential Attacks against the Helix Stream Cipher," *Fast Software Encryption 2004* (B. Roy, W. Meier, eds.), vol. 3017 of *LNCS*, pp. 94-108, Springer-Verlag, 2004.
18. K. Nyberg, L. Knudsen, "Provable Security Against a Differential Attack," *Journal of Cryptology*, 8(1):27-37, 1991.
19. S. Paul, B. Preneel, "Solving Systems of Differential Equations of Addition," Cryptology ePrint Archive, Report 2004/294, Available Online at `http://eprint.iacr.org/2004/294`, 2004.
20. R. L. Rivest, M. Robshaw, R. Sidney, Y. L. Yin, "The RC6 Block Cipher," Available Online at `http://theory.lcs.mit.edu/ rivest/rc6.ps`, June 1998.
21. B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C.Hall, N. Ferguson, "The Twofish Encryption Algorithm: A 128-Bit Block Cipher," John Wiley & Sons, April 1999, ISBN: 0471353817.
22. O. Staffelbach, W. Meier, "Cryptographic Significance of the Carry for Ciphers Based on Integer Addition," *Crypto '90* (A. Menezes, S. A. Vanstone, eds.), vol. 537 of *LNCS*, pp. 601-614, Springer-Verlag, 1991.
23. H. Yoshida, A. Biryukov, Christophe De Cannière, J. Lano, B. Preneel, "Non-randomness of the Full 4 and 5-Pass HAVAL," *SCN 2004* (Carlo Blundo and Stelvio Cimato, eds.), vol. 3352 of *LNCS*, pp. 324-336, Springer-Verlag, 2004.
24. J. Wallén, "Linear Approximations of Addition Modulo 2^n," *Fast Software Encryption 2003* (T. Johansson, ed.), vol. 2887 of *LNCS*, pp. 261-273, Springer-Verlag, 2003.

A Tree Based One-Key Broadcast Encryption Scheme with Low Computational Overhead

Tomoyuki Asano and Kazuya Kamio

Sony Corporation
6-7-35 Kitashinagawa, Shinagawa-ku, Tokyo 141-0001, Japan
tomo@arch.sony.co.jp, Kazuya.Kamio@jp.sony.com

Abstract. In this paper, we propose a new broadcast encryption method which is a modification of the Complete Subtree method and it reduces the number of keys a receiver stores to one. There have been proposed some methods which minimize the number of keys for a receiver to one. The most efficient one among them uses RSA cryptosystem in order to reduce the number of keys, while the proposed method is based on Rabin cryptosystem. The computational overhead at receivers in our method is around $1/\log_2 e$ compared with the most efficient method proposed previously, where e is a public exponent of RSA. We examine this result by experiments. Therefore, the proposed method is the most efficient among tree based one-key methods with respect to the computational overhead at receivers. This reduction in the computational overhead is achieved in exchange for an increase in the size of nonsecret memory by [log N * few (e. g. eight)] bits, where N is the total number of receivers. The security of the proposed method is equivalent to Rabin cryptosystem in the sense of key-intractability in the random oracle model.

1 Introduction

We deal with *broadcast encryption schemes* or *revocation schemes*, in which a sender can distribute secret information securely to a group of receivers excluding specified receivers (called *revoked receivers*) over a broadcast channel. One of the main applications of the broadcast encryption technology is digital rights management (DRM) of copyrighted contents. For example, in a content protection scheme with recordable media, a session key to encrypt or decrypt the contents stored on a medium can be retrieved only by authorized receivers (i. e. players or recorders) having collect receiver keys. If these receiver keys are stolen or exposed, these keys are revoked from the system, and as a result the receiver having only the exposed keys will not be able to retrieve session keys from media which are produced after the revocation.

We have two simple methods for broadcast encryption. *The one key method* requires each receiver to store only one unique key, however the sender must broadcast $N - r$ ciphertexts encrypted under each of the keys possessed by unrevoked receivers, where N and r denote the total number of receivers in the system and the number of revoked receivers, respectively.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 89–100, 2005.

The power set method defines a power set of N receivers, i. e. $\{S_{b_1 \cdots b_i \cdots b_N}\}$, where $b_i \in \{0, 1\}$ indicates whether or not receiver i belongs to subset $S_{b_1 \cdots b_i \cdots b_N}$. The sender broadcasts only one ciphertext of secret information which is encrypted with a key corresponding to an appropriate subset. On the other hand, each receiver must store 2^{N-1} keys since it belongs to 2^{N-1} subsets.

Both methods are impractical for a large N with respect to the upper bound of the number of ciphertexts to be broadcast (the communication overhead) or the number of keys each receiver stores (the storage overhead). We should also consider another criterion: the computational overhead at each receiver. It should be noted that usually administrators and broadcasters are assumed to have much greater memory and computing resources than receivers.

1.1 Related Work

Berkovits [4] and Fiat et al. [6] independently introduced the notion of broadcast encryption. Berkovits used a secret sharing scheme to construct a broadcast encryption method. Fiat et al. combined their 1-resilient broadcast encryption methods hierarchically in order to construct an r-resilient method which is resistant to any collusion of at most r revoked receivers.

Wallner et al. [17] and Wong et al. [19] independently proposed efficient methods for key distribution, using a logical key-tree structure. Their methods assign a receiver to a leaf of a tree, and give the receiver $\log N + 1$ node keys corresponding to nodes on the path from the leaf to the root. Note that logarithms are base 2, throughout this paper. To revoke a receiver, unrevoked receivers update all node keys which have been shared by the revoked one. Then the sender transmits secret information encrypted with the renewed root key.

These methods assume that receivers can change their keys. However, there are challenging problems for use of such receivers: synchronization and secure update of those keys, and their solutions might increase the production cost of these receivers. Therefore broadcast encryption methods which allow receivers without the ability to update their key are preferred for many applications. Such receivers are called *stateless receivers*.

The notion of stateless receivers was introduced by Naor et al. [12], who also proposed two efficient methods suitable for such receivers. The Complete Subtree (CS) method is a direct application of the binary key-tree structure proposed in [17,19] for stateless receivers. The communication, storage and computational overhead in CS are $r \log (N/r)$, $\log N+1$ and $O(\log \log N)$, respectively. The Subset Difference (SD) method improves the algorithm to divide the set of receivers into subsets and the key assignment mechanism of CS using a pseudo-random sequence generator. Its communication, storage and computational overhead are $2r-1$, $\frac{1}{2}\log^2 N + \frac{1}{2}\log N + 1$ and $O(\log N)$, respectively. Similar to CS, numbers of modifications of SD such as [2,3,7,8,9,18] have been proposed.

Asano [1] modified CS using an a-ary tree and the master-key technique proposed by Chick et al. [5], where a is an arbitrary integer satisfying $a > 1$. Method 1 proposed in [1], which we call CS-MKT, reduces the storage overhead

Table 1. The properties of the original Complete Subtree method [12] and its modifications proposed in [1], [13], [15] and in this paper. N, r, M and e denote the total number of receivers, the number of revoked receivers, the modulus of RSA and Rabin, and the public exponent of RSA, respectively. †The computational overhead in the original CS is $O(\log\log N)$ lookups

	# ciphertexts	# keys	Comp. overhead
CS [12]	$r\log(N/r)$	$\log N+1$	$O(\log\log N)^{\dagger}$
CS-MKT [1]	$r\log(N/r)$	1	$O\left(\max\{\log^5 N, \log^2 N\log^2 M\}\right)$
CS-TOPT [13][15]	$r\log(N/r)$	1	$O\left(\log e\log N\log^2 M\right)$
CS-MRT (This paper)	$r\log(N/r)$	1	$O\left(\log N\log^2 M\right)$

and the communication overhead to one key and to $\frac{r\log(N/r)}{\log a}+r$, respectively, in exchange for an increase in the computational overhead to $O\left(\frac{2^a\log^5 N}{\log a}\right)$.

Nojima et al. [13] (its improved version is [14]) and Ogata et al. [15] independently modified CS using trapdoor one-way permutations based on RSA cryptosystem. Their methods reduce the number of keys a receiver stores to one, while keeping the communication overhead as equal to CS. We call their methods CS-TOPT. Note that the computational overhead at receivers in CS-TOPT is smaller than that in CS-MKT, and CS-TOPT is the most efficient tree based one-key method, with respect to the computational overhead.

Gentry and Ramazan [7] used the master-key technique [5] to modify SD. Their method, which we call GR, divides subsets of receivers in SD into some portions, and each unrevoked receiver derives a key corresponding to a portion from its unique master key. The storage and communication overhead in GR are one key and $\frac{r\log(N/r)}{\log a}+r$, respectively, where a is an integer satisfying $2\le a\le N$. Note that its computational overhead is greater than CS-TOPT. Namely, receivers in GR and CS-TOPT must perform $(2a-\log a-1)\frac{\log N}{\log a}$ and $\log N$ modular exponentiations, respectively. Moreover, the computational cost of each modular exponentiation in GR is larger than that in CS-TOPT.

1.2 Our Contribution

In this paper we focus on CS and its modifications. We propose a new key-tree structure, which we call Modified Rabin Tree (MRT), based on Rabin cryptosystem [16]. Then we modify CS using MRT. The construction of our modification, called CS-MRT, is similar to CS-TOPT based on RSA. However, our method uses Rabin cryptosystem in order to reduce the number of keys a receiver stores, and hence the computational overhead at a receiver is much smaller than CS-TOPT. We also show that the security of our method is equivalent to breaking Rabin, in the sense of key-intractability in the random oracle model.

Table 1 summarizes the properties (the number of broadcast ciphertexts, the number of keys a receiver stores, and the computational overhead at a receiver)

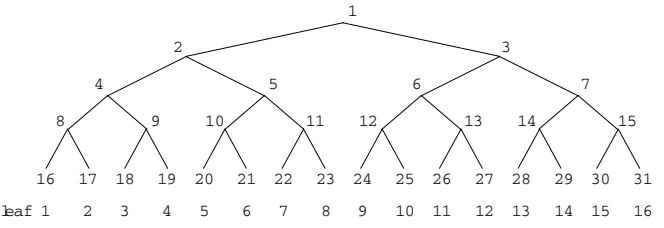


Fig. 1. A binary tree with 16 leaves

of the original CS and its modifications CS-MKT [1] (assuming its parameter a is set as $a = 2$), CS-TOPT [13,15] and CS-MRT proposed in this paper. Similar to other modifications, CS-MRT reduces the number of keys for a receiver to one, while keeping the upper bound of the number of ciphertexts to be broadcast as the same as in CS, i. e. $r \log (N/r)$. CS-MRT is more efficient with respect to the computational overhead at receivers than any previously proposed modification of CS which reduces the number of keys a receiver stores to one. This reduction in the computational overhead is achieved in exchange for an increase in the nonsecret storage at receivers. In CS-MRT, receivers must store $\log N$ nonsecret values with the size of small number of (e. g. eight) bits.

2 Underlying Tree Structure

This section presents the basic tree structure and two cryptographic tree structures: *Rabin Tree* (RT) introduced by Kikuchi [10] and our proposal *Modified Rabin Tree* (MRT), which is used to construct our broadcast encryption method.

2.1 The Basic Tree Structure

For all methods discussed below we assume that the total number of receivers N is a power of 2. These methods use a complete binary tree with N leaves. Each node in the tree is numbered l ($l = 1, \ldots, 2N - 1$) where the root is 1 and other nodes are numbered with level order from left to right. Note that child nodes of node l are numbered $2l$ and $2l + 1$, respectively. Each node has *node value* NV_l ($l = 1, \ldots, 2N-1$). Each leaf is also represented by leaf number $leaf_m$ ($m = 1, \ldots, N$), where the left most leaf is $leaf_1$. Let $path_m$ be the path from the root to $leaf_m$. Let $l \in T$ denote that node l is included in tree T, and let $P(l)$ denote the parent node of node l. Moreover, let $l_1 \prec l_2$ denote that node l_1 is an ancestor of node l_2. Figure 1 illustrates a binary tree with 16 leaves.

2.2 Rabin Tree

Kikuchi [10] proposed a broadcast encryption scheme using *Rabin Tree* (RT), which is a structure based on Rabin cryptosystem. Let $M_L = p_L q_L$ and $M_R =$

$p_R q_R$ be products of two large primes chosen by TC, satisfying $M_L < M_R$. Node values NV_l $(l = 1, \ldots, 2N-1)$ in RT satisfy the following (1) and (2):

$$NV_l \in QR_{M_L} \cap QR_{M_R} \tag{1}$$

$$NV_l = NV_{2l}^2 \bmod M_L = NV_{2l+1}^2 \bmod M_R \tag{2}$$

where QR_M denotes a set of quadratic residues mod M. RT is constructed as follows.

1. Let Trusted Center (TC) be the administrator and also the sender of the broadcast encryption schemes discussed in this paper. TC randomly chooses an element $NV_1 \in \mathbb{Z}_{M_L}$ satisfying (1). TC sets counter $l = 1$.
2. For each M_L and M_R, TC regards NV_l as a ciphertext of Rabin cryptosystem and performs decryption. Then TC sets one of four solutions with respect to M_L (respectively, M_R) as NV_{2l} (resp., NV_{2l+1}).
3. TC checks whether NV_{2l} (resp., NV_{2l+1}) satisfies (1). If not, TC sets another solution of the decryption as NV_{2l} (resp., NV_{2l+1}) and checks it.
4. If all four solutions do not satisfy (1), then TC changes node value of their parent node with another solution. If necessary, TC repeats this backtrack until it assigns node values for all $2N-1$ nodes in the tree.

Kikuchi [10] has reported that the probability that TC successfully assigns all node values from a fixed value NV_1 is $\frac{1}{16}\left(1 - \left(\frac{15}{16}\right)^4\right)^{N-1}$. Since this probability decreases significantly as N grows, it is difficult to construct RT for a large N.

2.3 Modified Rabin Tree

We use a similar but slightly different tree structure called Modified Rabin Tree (MRT). In addition to node value NV_l, all nodes except the root in MRT has its own "salt" value $salt_l$ $(l = 2, \ldots, 2N-1)$. MRT is constructed as follows.

1. Given input 1^λ where λ is a security parameter, TC chooses two large primes p and q such that the length of their product $M = pq$ is λ (namely, $|M| = \lambda$). TC also selects hash function H_M which maps elements of arbitrary length into random elements in $\mathbb{Z}_M$. Then it uniformly chooses an element in $\mathbb{Z}_M^*$ and sets it as node value NV_1 for the root.
2. For $l = 2$ to $2N-1$ TC determines NV_l and $salt_l$ as follows.
   (a) TC finds the smallest positive integer $salt_l$ such that $tmp_l \in QR_M$, where $tmp_l \triangleq \left(NV_{\lfloor l/2 \rfloor} - H_M\left(l || salt_l\right)\right) \bmod M$ and $||$ denotes a bitwise concatenation.
   (b) Regarding tmp_l as a ciphertext, TC performs Rabin decryption to compute square roots of tmp_l mod M. TC sets one of the solutions as NV_l.

Intuitively, MRT alleviates the need for an exponential sized search for a suitable node value for the root in RT. By assigning a salt value to each node, it changes a global search in the entire tree into a small search at each node.

As we see later in Section 5, the probability that a randomly chosen element $x \in \mathbb{Z}_M$ satisfies $x \in QR_M$ is about 1/4. Therefore, TC must try four integers in average as $salt_l$ in order to find an appropriate one for each node. In total, TC needs $O(N)$ trials to fix node values and salt values for all $2N-2$ nodes except the root, and hence MRT is significantly efficient to construct than RT.

We have actually constructed MRT with $N = 2^{25}$, where $|p| = |q| = 512$ bits and $|M| = 1024$ bits, using Xeon 2.80GHz, 512KB cache, 2GB RAM machine with gcc-3.3, ntl-5.3.2 and gmp-4.1.3. The processing time was 482846 seconds (5.6 days). This experimental result shows that construction of MRT is practical.

Note that for the purpose of broadcast encryption as we see in Section 3.2, MRT is constructed only once by TC before it starts up the system. It may use numbers of high performance machines and may spend long time (e. g. a couple of months). Therefore, even when we need MRT with a larger N (for example $N = 2^{30}$), it can be practically constructed with suitable resources and time.

3 The Proposed Broadcast Encryption Method

In this section we give brief explanations about CS and our proposed broadcast encryption method, which is a modification of CS using MRT.

3.1 The Complete Subtree (CS) Method [12]

TC assigns receiver u_m to leaf $leaf_m$ $(m = 1, \ldots, N)$. Node values NV_l $(l = 1, \ldots, 2N-1)$ are randomly chosen from $\{0,1\}^C$, where C denotes the key length of symmetric encryption algorithm E. These node values are directly used as *node keys* NK_l, i. e. $NK_l = NV_l$. Receiver u_m is given a set of node keys corresponding to the nodes on $path_m$. Therefore each receiver must store $\log N + 1$ keys in a secret manner. In the example depicted in Fig. 1, receiver u_4 assigned to $leaf_4$ stores five node keys: NK_1, NK_2, NK_4, NK_9 and NK_{19}.

In order to broadcast secret information I, TC finds the Steiner Tree $ST(R)$ which is the minimal subtree of the original tree containing the root and all leaves in R, where R is a set of leaves such that receivers assigned to them are revoked. Then TC broadcasts ciphertexts $\{E_{NK_l}(I) \mid l \notin ST(R) \cap P(l) \in ST(R)\}$. It has been reported in [12] that the number of broadcast ciphertexts for a transmission of I including revocation of r $(= |R|)$ receivers is at most $r \log(N/r)$.

3.2 The Proposed Method: CS-MRT

TC generates MRT and publishes M, H_M and $salt_l$ $(l = 2, \ldots, 2N-1)$. It also defines node keys NK_l $(l = 1, \ldots, 2N-1)$ as $NK_l = NV_l$. Receiver u_m assigned to leaf $leaf_m$ is given node value NV_l of $leaf_m$ and $\log N$ salt values $salt_l$ for the nodes located on $path_m$ except the root. It enables u_m to derive any NV_l for node l on $path_m$, one by one from the leaf to the root as

$$NV_{\lfloor l/2 \rfloor} = \left(NV_l^2 + H_M(l||salt_l)\right) \bmod M \tag{3}$$

In the example depicted in Fig. 1, receiver u_4 stores only one node value NV_{19} and four salt values $salt_2$, $salt_4$, $salt_9$ and $salt_{19}$. Using these values u_4 can derive any node values of nodes on $path_4$. Note that salt values are public values and do not require secrecy. We call this method CS-MRT. The way for transmission of secret information in CS-MRT is the same as in CS.

4 Security of the Proposed Method

The discussion on security of CS-MRT in this section mainly follows the analysis of a key management scheme by Nojima and Kaji [14], in which the number of receivers N is assumed to be represented by a polynomial of a security parameter λ, namely $N = w(\lambda)$. First of all, we clarify the assumption regarding Rabin function.

Definition 1 (Rabin function). *A Rabin function is a 2-tuple of polynomial-time algorithms* (Keygen, Forward)*, where:*

Rabin.Keygen(1^λ)*: Takes as input a security parameter* 1^λ*. It returns two primes* p *and* q *and their product* M*, such that the size of* M *is* λ*.*

Rabin.Forward(M, x)*: Takes as input modulus* M *and element* $x \in \mathbb{Z}_M$*. It returns* $y = x^2 \bmod M$*.*

Definition 2 (Intractability of inverse of Rabin function). *We assume that Rabin function is intractable to inverse. Namely, for all polynomial-time algorithm* $\mathcal{A}$ *the probability*

$$\mathsf{AdvRabin}_{\mathcal{A}}(\lambda) = Pr\ [\ x'^2 \equiv x^2 \pmod{M} \mid \begin{array}{l}(p, q, M) \leftarrow \mathsf{Rabin.Keygen}\left(1^\lambda\right); \\ x \leftarrow \mathbb{Z}_M; \\ y = \mathsf{Rabin.Forward}(M, x); \\ x' \leftarrow \mathcal{A}(M, y)\ ]\end{array}$$

is a negligible function in λ*, where* $x \leftarrow \mathbb{Z}_M$ *denotes that* x *is chosen uniformly from* $\mathbb{Z}_M$*.*

Note that Rabin [16] proved that inverting Rabin.Forward function is as intractable as factoring M. We consider CS-MRT as an Access Control (AC) scheme as follows.

Definition 3 (Access Control scheme from CS-MRT). *An Access Control scheme* AC[CS-MRT] *is a 2-tuple of polynomial-time algorithms* (Keygen, Derive)*, where:*

AC[CS-MRT].Keygen(1^λ)*: Takes as input a security parameter* 1^λ*. It returns* p*,* q*,* M*,* H_M*,* NV_l $(l = 1, \ldots, 2N-1)$*,* $salt_l$ $(l = 2, \ldots, 2N-1)$*, where* H_M*,* NV_l *and* $salt_l$ *are generated in the way mentioned in Section 2.3. Let* pub *denote public information such that* M*,* H_M *and* $salt_l$*.*

AC[CS-MRT].Derive$(l, NV_l, n, \mathsf{pub})$*: Takes as input node number* l *of a leaf to which a receiver is assigned, its node value* NV_l*, node number* n *in the tree, and* pub*. It returns* NV_n *if* $n \preceq l$*, otherwise returns special symbol* $\perp$*.*

The scenario we consider is that all of the revoked receivers collude and try to compute any node value of a node which is not an ancestor of any of the leaves to which the revoked receivers are assigned. For node n in MRT, let $\overline{NV_n} \triangleq \{NV_l \mid l \text{ is a leaf such that } n \not\preceq l\}$.

We consider an adversary in AC[CS-MRT] as a pair of probabilistic polynomial-time algorithms $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$. $\mathcal{B}_1$ takes as input a security parameter 1^λ and pub, and outputs node number n and auxiliary information aux which is helpful for $\mathcal{B}_2$. $\mathcal{B}_2$ takes as input n, $\overline{NV_n}$, pub and aux, and outputs NV_n.

Here we define the security notion, then state the security theorem on AC[CS-MRT].

Definition 4 (Key-Intractability of AC.[CS-MRT]). *We say that* AC.[CS-MRT] *is secure in the sense of Key-Intractability if for all probabilistic polynomial-time algorithms $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ the probability*

$$\mathsf{AdvAC[CS\text{-}MRT]}_{\mathcal{B}}(\lambda) =$$

$$Pr\left[x = NV_n \mid (p, q, NV_l\ (l = 1, \ldots 2N-1), \mathsf{pub}) \leftarrow \mathsf{AC[CS\text{-}MRT].Keygen}\left(1^\lambda\right);\right.$$
$$(n \in [1, 2N-1], \mathsf{aux}) \leftarrow \mathcal{B}_1\left(1^\lambda, \mathsf{pub}\right);$$
$$\left. x \leftarrow \mathcal{B}_2\left(n, \overline{NV_n}, \mathsf{pub}, \mathsf{aux}\right)\right]$$

is a negligible function in λ.

Theorem 1. *If p, q are randomly chosen from a collection of appropriate prime numbers and H_M is randomly chosen from the random oracles, then* AC[CS-MRT] *is secure in the sense of Key-Intractability, in the random oracle model.*

Proof. We construct probabilistic polynomial-time algorithm $\mathcal{A}$ which inverts Rabin.Forward function (namely, given randomly chosen value $y \in QR_M$, $\mathcal{A}$ finds x satisfying $x^2 \equiv y \pmod M$), by using an adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ to AC[CS-MRT] which outputs node value NV_n of node n, given $\overline{NV_n}$ and public information, with oracle access with querying to $\mathcal{A}$ on execution of H_M. Note that since $\mathcal{B}$ views H_M as a random oracle, let $\mathcal{B}_1^{H_M}\left(1^\lambda, \mathsf{pub'}\right)$ and $\mathcal{B}_2^{H_M}\left(\overline{NV_n}, \mathsf{pub'}, \mathsf{aux}\right)$ represent $\mathcal{B}_1\left(1^\lambda, \mathsf{pub}\right)$ and $\mathcal{B}_2\left(\overline{NV_n}, \mathsf{pub}, \mathsf{aux}\right)$, respectively, where pub' denotes public information pub excluding H_M. The inverter $\mathcal{A}$ of Rabin.Forward function is constructed as follows.

1. $\mathcal{A}$ constructs a binary tree with N leaves.
2. $\mathcal{A}$ randomly picks up target node $c \in [1, 2N-1]$.
3. $\mathcal{A}$ determines node values NV_l for nodes $c \not\preceq l$ and hash values $L_l \triangleq H_M(l||salt_l)$ for all nodes in the tree as follows.
   (a) For node l such that $c \preceq l$, let L_l be a randomly chosen element in $\mathbb{Z}_M$.
   (b) For leaf l such that $c \not\preceq l$, let NV_l be a randomly chosen element in $\mathbb{Z}_M$.
   (c) If two nodes l and m are sibling nodes in the tree such that both of node values NV_l and NV_m have been already fixed, then choose one of L_l and L_m randomly from $\mathbb{Z}_M$ and determine the other so that $\left(NV_l^2 + L_l\right) \bmod M = \left(NV_m^2 + L_m\right) \bmod M$ is satisfied. In addition, let $NV_{P(l)}$ be $\left(NV_l^2 + L_l\right) \bmod M$.

(d) If node l is the sibling node of c and NV_l has been already fixed, then choose L_l so that $(y + L_c) \bmod M = \left(NV_l^2 + L_l\right) \bmod M$ is satisfied. In addition, let $NV_{P(c)}$ be $(y + L_c) \bmod M$.

Note that this construction makes NV_c be one of four values x such that $x^2 \equiv y \pmod{M}$. $\mathcal{A}$ uses values generated above in order to give $\overline{NV_n}$ as input to $\mathcal{B}$ and to answer the queries from $\mathcal{B}$ on L_l.

4. $\mathcal{A}$ runs $\mathcal{B}_1^{H_M}\left(1^\lambda, \mathsf{pub}'\right)$ and obtains its output (n, aux).
5. If $n \neq c$ then $\mathcal{A}$ halts. Otherwise, $\mathcal{A}$ proceeds to the next step.
6. $\mathcal{A}$ runs $\mathcal{B}_2^{H_M}\left(\overline{NV_n}, \mathsf{pub}', \mathsf{aux}\right)$ and obtains its output x.
7. $\mathcal{A}$ outputs x.

The above process completes only if $\mathcal{B}_1$'s output n is equal to c. Recall that the number of receivers N is represented by a polynomial of the security parameter λ, namely $N = w(\lambda)$. Since the tree has $2N-1 = 2w(\lambda)-1$ nodes, the probability that it happens is $1/(2w(\lambda)-1)$. Hence, $\mathcal{A}$ outputs x such that $x^2 \equiv y \pmod{M}$ with probability $\mathsf{AdvRabin}_{\mathcal{A}}(\lambda) = \frac{1}{2w(\lambda)-1}\mathsf{AdvAC[CS\text{-}MRT]}_{\mathcal{B}}(\lambda)$. This means that if $\mathsf{AdvAC[CS\text{-}MRT]}_{\mathcal{B}}(\lambda)$ is a nonnegligible function in λ then so is $\mathsf{AdvRabin}_{\mathcal{A}}(\lambda)$. However, this contradicts the assumption that Rabin function is intractable to invert given in Definition 2. Hence, $\mathsf{AdvAC[CS\text{-}MRT]}_{\mathcal{B}}(\lambda)$ must be a negligible function in λ. This proves the theorem. □

Moreover, using $\mathcal{B}$, we can construct probabilistic polynomial-time algorithm $\mathcal{A}'$ which factorizes M. $\mathcal{A}'$ randomly chooses $x_1 \in \mathbb{Z}_M^*$ and uses $x_1^2 \bmod M$ as y when it constructs the tree. If output x of $\mathcal{B}_2$ satisfies $x \neq \pm x_1 \bmod M$, then $\gcd(x_1 - x, M)$ gives a factor of M. When $\mathcal{B}_2$ outputs x, the probability that x satisfies $x \neq \pm x_1 \bmod M$ is $1/2$. Therefore, the probability that $\mathcal{A}'$ succeeds to factorize M is $\frac{1}{4w(\lambda)-2}\mathsf{AdvAC[CS\text{-}MRT]}_{\mathcal{B}}(\lambda)$, which is nonnegligible if $\mathsf{AdvAC[CS\text{-}MRT]}_{\mathcal{B}}(\lambda)$ is nonnegligible.

5 Efficiency of the Proposed Method

In this section we analyze efficiency of CS-MRT compared with CS and its previously proposed modifications. The following analysis is summarized in Table 1.

5.1 Communication Overhead

Since CS-MRT adopts the same way for sending secret information as CS, the communication overhead is also the same. Namely, the upper bound of the number of ciphertexts is $r \log(N/r)$. CS-MKT (assuming its parameter a is set as $a = 2$) proposed in [1], and CS-TOPT in [13,15] have the same property.

5.2 Storage Overhead

First, we consider the size of secure memory where each receiver stores its keys. A receiver in CS stores $\log N + 1$ node keys. On the other hand, a receiver in

CS-MRT stores only one node value. Since MRT is based on Rabin cryptosystem, the size of each node value is equal to the size of a secure Rabin modulus. As an example, if we set parameters as the total number of receivers $N = 2^{25}$, and the size of a node key in CS and a node value in CS-MRT as 128 and 1024 bits, respectively, then the size of the secure memory of the receiver in CS-MRT is about 70% smaller than in CS. This reduction rate becomes larger as N increases. CS-MKT and CS-TOPT also have the same reduction rate, assuming that the size a secure modulus for RSA is equal to that for Rabin.

Second, the size of memory which does not need secrecy is considered. In total, CS-MRT uses $2N - 2$ salt values $salt_l$ assigned to nodes except the root, and each receiver needs $\log N$ of them. Recall that $salt_l$ is the minimum positive integer such that $tmp_l = \left(NV_{\lfloor l/2 \rfloor} - H_M\left(l||salt_l\right)\right) \bmod M \in QR_M$. For a randomly chosen element $y \in \mathbb{Z}_M$ where M is the product of two primes p and q, the probability that $y \in QR_M$ is $\frac{(p-1)/2+1}{p}\frac{(q-1)/2+1}{q} \approx \frac{1}{4}$, since $y \in QR_M \Leftrightarrow \left(\frac{y}{p}\right) = \left(\frac{y}{q}\right) = 1$ where $\left(\frac{y}{p}\right)$ denotes Legendre symbol. Therefore, under the assumption that the output of H_M is uniformly chosen from $\mathbb{Z}_M$, the expected number of integers to try as $salt_l$ until we find an appropriate one is four, and hence each $salt_l$ can be represented as a $\log 4 = 2$ bit number in average.

On the other hand, if we fix in advance the maximum length of each $salt_l$ as len, the probability that all 2^{len} integers generate $tmp_l \notin QR_M$ is approximately $\left(\frac{3}{4}\right)^{2^{len}}$. If we set $len = 8$, this probability becomes $1.0{*}10^{-32}$, and it is considered enough small even if N is a large number such as $2^{25} \approx 3.4 * 10^7$. Actually, in the experimental results where we constructed MRT with $N = 2^{25}$ leaves, the biggest value of $salt_l$ was 65, and their average was 4.0003. Hence, receivers in CS-MRT need only [$\log N$ * few (e. g. eight)] bits of nonsecret storage for storing the salt values.

Note that this increase in the size of nonsecret but unique storage for a receiver does not require the receiver to equip a new storage apparatus. A receiver in any tree based method must store its own unique information such as address or leaf number which requires $O(\log N)$ bits for representation. Therefore, it must equip $O(\log N)$ bits of memory for storing its unique data anyway. CS-MRT merely requires an increase in the size of such memory by a small factor.

A receiver in CS-MKT uses $\log N + 1$ public primes, which may be stored directly, or be generated in an on-the-fly manner. In the former case, the receiver needs $O\left(\log^2 N\right)$ bits of nonsecret storage, and in the latter case it must perform $O\left(\log^5 N\right)$ bit operation when it uses Miller-Rabin primality testing algorithm [11]. In contrast, CS and CS-TOPT use $O(1)$ nonsecret information.

5.3 Computational Overhead

Here we consider computational overhead at receivers. In CS, each receiver needs $O(\log \log N)$ lookups. To derive node value NV_l from its master key MV_m, receiver u_m in CS-MKT must perform operation of $MV_m^{w_m/p_l} \bmod M$, where p_l is a prime assigned to node l and w_m is a product of p_l's which are assigned to

the nodes on $path_m$. This operation requires $O\left(\max\{\log^4 N, \log^2 N \log^2 M\}\right)$ of computational overhead, assuming u_m stores $\log N + 1$ primes directly.

A receiver in CS-TOPT must perform one modular exponentiation with public exponent e, and one execution of hash function H_M, in order to derive node value $NV_{\lfloor l/2 \rfloor}$ from its child's node value NV_l, i. e.

$$NV_{\lfloor l/2 \rfloor} = \left(NV_l^e + H_M\left(l\right)\right) \bmod M \tag{4}$$

Similarly, a receiver in CS-MRT must perform one modular squaring and one execution of a hash function to do it (see formula (3)). Since the computational cost of a hash function is considered as much smaller than modular exponentiation or modular squaring, we ignore it and focus on modular arithmetic.

The computational overhead for $x^e \bmod M$ is asymptotically represented as $O\left(\log e \log^2 M\right)$. More precisely, if we use one of "repeated square-and-multiply" algorithms [11] to compute $x^e \bmod M$, we must perform $\lfloor \log e \rfloor$ squaring and $wt\left(e\right) - 1$ multiplications, where $wt\left(e\right)$ denotes Hamming weight of e. By definition, e must be an odd number satisfying $e > 2$ and hence $wt\left(e\right) \geq 2$. Note that since squaring is a special case of multiplication, the computational cost of squaring is smaller than multiplication. Therefore, the computational cost for $x^e \bmod M$ is greater than the cost of $\lfloor \log e \rfloor + 1$ squaring.

On the other hand, in CS-MRT, the computation for deriving a node value from its child's node value requires only one squaring, and its cost is asymptotically represented as $O\left(\log^2 M\right)$. This means that the computational overhead at receivers in CS-MRT is at least $\lfloor \log e \rfloor + 1$ times smaller than receivers in CS-TOPT. It should be noted that by definition the minimum value of e is 3, in this case $x^e \bmod M$ requires one squaring and one multiplication. Even this minimum e is used in CS-TOPT, the computational overhead at receivers in CS-MRT is at least two times smaller than CS-TOPT. Therefore, the proposed CS-MRT is more efficient with respect to the computational overhead at receivers than other tree based one-key broadcast encryption methods.

Our experimental results support the above analysis. Using Xeon 2.80GHz, 512KB cache, 2GB RAM machine with gcc-3.3, ntl-5.3.2 and gmp-4.1.3, we compared processing time of formulae (3) in CS-MRT and (4) in CS-TOPT where $|p| = |q| = 512$ bits, $|M| = 1024$ bits, $e = 2^{16} + 1$ (i. e. $\lfloor \log e \rfloor + 1 = 17$). These average time (of 10^6 trials) are 5.867 and 112.4 μs, respectively, which are approximately $1 : 19$[1].

References

1. T. Asano, "A Revocation Scheme with Minimal Storage at Receivers," Advances in Cryptology - Asiacrypt 2002, Lecture Notes in Computer Science 2501, pp. 433-450, Springer, 2002.

[1] Note that this results do not contain processing time of hash function H_M. We used SHA-1 as substitution for H_M in these experiments. If we count processing time of the hash function, these average of processing time become 8.493 and 114.2 μs, respectively, which are approximately $1 : 13$.

2. T. Asano, "Reducing Storage at Receivers in SD and LSD Broadcast Encryption Schemes," Information Security Applications, 4th International Workshop, WISA 2003, Lecture Notes in Computer Science 2908, pp. 317-332, Springer, 2004.
3. N. Attrapadung, K. Kobara, and H. Imai, "Sequential Key Derivation Patterns for Broadcast Encryption and Key Predistribution Schemes," Advances in Cryptology - Asiacrypt 2003, Lecture Notes in Computer Science 2894, pp. 374-391, Springer, 2003.
4. S. Berkovits, "How to Broadcast a Secret," Advances in Cryptology - Eurocrypt '91, Lecture Notes in Computer Science 547, pp. 535-541, Springer, 1991.
5. G. C. Chick and S. E. Tavares, "Flexible Access Control with Master Keys," Advances in Cryptology - Crypto '89, Lecture Notes in Computer Science 435, pp. 316-322, Springer, 1990.
6. A. Fiat and M. Naor, "Broadcast Encryption," Advances in Cryptology - Crypto '93, Lecture Notes in Computer Science 773, pp. 480-491, Springer, 1994.
7. C. Gentry and Z. Ramzan, "RSA Accumulator Based Broadcast Encryption," Information Security, 7th International Conference, ISC 2004, Lecture Notes in Computer Science 3225, pp. 73-86, Springer, 2004.
8. M. T. Goodrich, J. Z. Sun and R. Tamassia, "Efficient Tree-Based Revocation in Groups of Low-State Devices," Advances in Cryptology - Crypto 2004, Lecture Notes in Computer Science 3152, pp. 511-527, Springer, 2004.
9. D. Halevy and A. Shamir, "The LSD Broadcast Encryption Scheme," Advances in Cryptology - Crypto 2002, Lecture Notes in Computer Science 2442, pp. 47-60, Springer, 2002.
10. H. Kikuchi, "Rabin Tree and its Application to Broadcast Encryption," (in Japanese), IEICE Technical Report ISEC 2003-13, pp. 9-12, 2003.
11. A. J. Menezes, P. C. van Oorschot and S. A. Vanstone, "Handbook of Applied Cryptography," CRC Press, 1997.
12. D. Naor, M. Naor and J. Lotspiech, "Revocation and Tracing Schemes for Stateless Receivers," Advances in Cryptology - Crypto 2001, Lecture Notes in Computer Science 2139, pp. 41-62, Springer, 2001.
13. R. Nojima and Y. Kaji, "Efficient Tree-based Key Management Using One-way Functions," (in Japanese), Proceedings of the 2004 Symposium on Cryptography and Information Security, pp. 189-194, 2004.
14. R. Nojima and Y. Kaji, "Secure, Efficient and Practical Key Management Scheme in the Complete-Subtree Method," IEICE Trans. Fundamentals, vol. E88-A, no. 1, pp. 189-194, 2001.
15. W. Ogata, T. Hiza and D. V. Quang, "Efficient Tree Based Key management based on RSA function," (in Japanese), Proceedings of the 2004 Symposium on Cryptography and Information Security, pp. 195-199, 2004.
16. M. O. Rabin, "Digitalized Signatures and Public-Key Functions as Intractable as Factorization," MIT Technical Report, MIT/LCS/TR-212, 1979.
17. D. Wallner, E. Harder and R. Agee, "Key Management for Multicast: Issues and Architectures," IETF Network Working Group, Request for Comments: 2627, available from `ftp://ftp.ietf.org/rfc/rfc2627.txt`, 1999.
18. P. Wang, P. Ning and D. S. Reeves, "Storage-Efficient Stateless Group Key Revocation," Information Security, 7th International Conference, ISC 2004, Lecture Notes in Computer Science 3225, pp. 25-38, Springer, 2004.
19. C. K. Wong, M. Gouda and S. S. Lam, "Secure Group Communications Using Key Graphs," Proceedings of ACM SIGCOMM '98, 1998.

Dynamic Group Key Agreement in Tree-Based Setting
(Extended Abstract)

Ratna Dutta and Rana Barua

Indian Statistical Institute
{ratna_r,rana}@isical.ac.in

Abstract. We present a provably secure tree based authenticated group key agreement protocol in dynamic scenario. Bilinear pairing and multi-signature are at the heart of our protocol. We prove that our protocol is provably secure in the standard security model of Bresson *et al.* An appropriate modification of Katz-Yung approach to tree based setting is adopted while proving its security against active adversaries. The protocol has an in-built hierarchical structure that makes it desirable for certain applications.

1 Introduction

A group key agreement protocol allows a group of users to exchange information over public network to agree upon a common secret key from which a session key can be derived. This common session key can later be used to achieve desirable security goals, such as authentication, confidentiality and data integrity.

Tree based group key agreement protocols are typically essential while the users are grouped into a hierarchical structure. The leaves of the tree denote individual users and each internal node corresponds to a user that represents the set of users in the subtree rooted at that node. The representative users have more computational resources than other users in the subtree. In a tree based group key agreement protocol, the set of all users in each subtree agree upon a common secret key. Besides, making optimal use of precomputed values in the previous session, a group of users can save computation and communication in subsequent sessions in which users join or leave the group. Moreover, some subclass of users agree upon multiple common keys in a single session which facilitates a typical subclass of users of the group to securely communicate among themselves. These features make tree based key agreement protocols desirable for certain applications.

In this work, we present a provably secure tree based authenticated group key agreement in the dynamic scenario where a user can join or leave the group as his desire with updating sets of keys. We can combine constant round protocols and tree based protocols to get hybrid group key agreement which are efficient in terms of both computation and communication. Consider the situation where

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 101–112, 2005.

there are collection of user sets, each having a common key agreed upon by executing an efficient constant round protocol among the users in that user set. Now executing the constant round protocol among these user sets may not always be desirable and executing the protocol among all the users may be expensive from computation or communication point of view when number of users in each subgroup is large. For instance, if the number of groups is about 20 and each group is large, then a constant round key agreement using the protocol of [11] will involve a large number of computations as well as communications. In contrast, the tree based protocol (with the representatives of each group) will compute the common key in 3 rounds with lesser number of communications and verifications. Thus, a tree based scheme can be incorporated among these user sets to get an efficient multi-party key agreement protocol.

A ternary tree based protocol was proposed by Barua *et al.* [2] that extends the basic Joux [8] protocol to multi-party setting. They have shown that the protocol is secure against passive adversaries. Dutta *et al.* [6] authenticate this unauthenticated protocol using multi-signature and provide a concrete security analysis against active adversaries in the standard model as formalized by Bresson *et al.* [5]. This security was achieved by modifying the Katz and Yung [9] technique to tree based setting. The present work further extends this static authenticated protocol [6] to dynamic authenticated protocol and provides a proof of security in the above security model. Our protocol is designed to ensure minimum modification to the computation already precomputed when a user leaves or joins the group. Besides, if the tree structure is not maintained after a join or leave operation, then the subsequent join or leave in the group can not be performed anymore. Thus retaining the tree structure is another important issue of our protocol.

2 Preliminaries

2.1 Cryptographic Bilinear Maps

Let G_1, G_2 be two groups of the same prime order q. A mapping $e : G_1 \times G_1 \rightarrow G_2$ satisfying the following properties is called a cryptographic bilinear map: (*Bilinearity*) $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in G_1$ and $a, b \in Z_q^*$; (*Non-degeneracy*) if P is a generator of G_1, then $e(P, P)$ is a generator of G_2; and (*Computablity*) there exists an efficient algorithm to compute $e(P, Q)$. Modified Weil Pairing [3] and Tate Pairing [1] are examples of such bilinear maps.

2.2 Security Model

We assume that the reader is familier with the model of Bresson *et al.* [5], which is the model in which we prove security of our dynamic key aggreement protocol. For completeness, we review their definitions (see [5]).

Let $\mathcal{P} = \{U_1, \ldots, U_n\}$ be a set of n (fixed) users or participants. A user can execute the protocol for group key agreement several times with different

partners, can join or leave the group at its desire by executing the protocols for Insert or Delete. We assume that users do not deviate from the protocol and adversary never participates as a user in the protocol. This adversarial model allows concurrent execution of the protocol. The interaction between the adversary $\mathcal{A}$ and the protocol participants occur only via oracle queries, which model the adversary's capabilities in a real attack. These queries are as follows, where Π_U^i denotes the i-th instance of user U and sk_U^i denotes the session key after execution of the protocol by Π_U^i.

- $\mathsf{Send}(U, i, m)$: This query models an active attack, in which the adversary may intercept a message and then either modify it, create a new one or simply forward it to the intended participant. The output of the query is the reply (if any) generated by the instance Π_U^i upon receipt of message m. The adversary is allowed to prompt the unused instance Π_U^i to initiate the protocol with partners $U_2, \ldots, U_l, l \leq n$, by invoking $\mathsf{Send}(U, i, \langle U_2, \ldots, U_l \rangle)$.
- $\mathsf{Execute}(\{(V_1, i_1), \ldots, (V_l, i_l)\})$: Here $\{V_1, \ldots, V_l\}$ is a non empty subset of $\mathcal{P}$. This query models passive attacks in which the attacker evesdrops on honest execution the protocol among unused instances $\Pi_{V_1}^{i_1}, \ldots, \Pi_{V_l}^{i_l}$ and outputs the transcript of the execution. A transcript consists of the messages that were exchanged during the honest execution of the protocol.
- $\mathsf{Join}(\{(V_1, i_1), \ldots, (V_l, i_l)\}, (U, i))$: This query models the insertion of a user instance Π_U^i in the group $(V_1, \ldots, V_l) \subseteq \mathcal{P}$ for which Execute have already been queried. The output of this query is the transcript generated by the invocation of algorithm Insert. If $\mathsf{Execute}(\{(V_1, i_1), \ldots (V_l, i_l)\})$ has not taken place, then the adversary is given no output.
- $\mathsf{Leave}(\{(V_1, i_1), \ldots, (V_l, i_l)\}, (U, i))$: This query models the removal of a user instance Π_U^i from the group $(V_1, \ldots V_l) \subseteq \mathcal{P}$. If $\mathsf{Execute}(\{(V_1, i_1), \ldots (V_l, i_l)\})$ has not taken place, then the adversary is given no output. Otherwise, algorithm Delete is invoked. The adversary is given the transcript generated by the honest execution of procedure Delete.
- $\mathsf{Reveal}(U, i)$: This outputs session key sk_U^i. This query models the misuse of the session keys, *i.e* known session key attack.
- $\mathsf{Corrupt}(U)$: This outputs the long-term secret key (if any) of player U. The adversarial model that we adopt is a weak-corruption model in the sense that only the long-term secret keys are compromised, but the ephemeral keys or the internal data of the protocol participants are not corrupted. This query models (perfect) forward secrecy.
- $\mathsf{Test}(U, i)$: This query is allowed only once, at any time during the adversary's execution. A bit $b \in \{0, 1\}$ is chosen uniformly at random. The adversary is given sk_U^i if $b = 1$, and a random session key if $b = 0$. This oracle computes the adversary's ability to distinguish a real session key from a random one.

An adversary which has access to the Execute, Join, Leave, Reveal, Corrupt and Test oracles, is considered to be passive while an active adversary is given access to the Send oracle in addition. We also use notations sid_U^i, the session identity for instance Π_U^i, (we set $\mathsf{sid}_U^i = S = \{(U_1, i_1), \ldots, (U_k, i_k)\}$ such that

$(U, i) \in S$ and $\Pi_{U_1}^{i_1}, \ldots, \Pi_{U_k}^{i_k}$ wish to agree upon a common key), pid_U^i, the partner identity for instance Π_U^i (we defined $\mathsf{pid}_U^i = \{U_1, \ldots, U_k\}$ such that $(U_j, i_j) \in \mathsf{sid}_U^i$ for all $1 \leq j \leq k$) and acc_U^i, a 0/1-valued variable (set to be 1 by Π_U^i upon normal termination of the session and 0 otherwise).

The adversary can ask Send, Execute, Join, Leave, Reveal and Corrupt queries several times, but Test query is asked only once and on a fresh instance. We say that an instance Π_U^i is *fresh* unless either the adversary, at some point, queried $\mathsf{Reveal}(U, i)$ or $\mathsf{Reveal}(U', j)$ with $U' \in \mathsf{pid}_U^i$ or the adversary queried $\mathsf{Corrupt}(V)$ (with $V \in \mathsf{pid}_U^i$) before a query of the form $\mathsf{Send}(U, i, *)$ or $\mathsf{Send}(U', j, *)$ where $U' \in \mathsf{pid}_U^i$. Finally adversary outputs a guess bit b'. Such an adversary is said to win the game if $b = b'$ where b is the hidden bit used by the Test oracle. Let Succ denote the event that the adversary $\mathcal{A}$ wins the game for a protocol XP. We define $\mathsf{Adv}_{\mathcal{A},\mathsf{XP}} := |2\,\mathsf{Prob}[\mathsf{Succ}] - 1|$ to be the advantage of the adversary $\mathcal{A}$ in attacking the protocol XP. The protocol XP is said to be a *secure unauthenticated group key agreement* (KA) protocol if there is no polynomial time *passive* adversary with non-negligible advantage. We say that protocol XP is a *secure authenticated group key agreement* (AKA) protocol if there is no polynomial time *active* adversary with non-negligible advantage. Next we define the advantage functions: $\mathsf{Adv}_{\mathsf{XP}}^{\mathsf{KA}}(t, q_E)$ to be the maximum advantage of any passive adversary attacking protocol XP running in time t and making q_E calls to the Execute oracle and $\mathsf{Adv}_{\mathsf{XP}}^{\mathsf{AKA}}(t, q_E, q_J, q_L, q_S)$ to be the maximum advantage of any active adversary attacking protocol XP, running in time t and making q_E calls to the Execute oracle, q_J calls to Join oracle, q_L calls to the Leave oracle and q_S calls to the Send oracle.

2.3 DHBDH Problem

Let (G_1, G_2, e) be as in Section 2.1. We define the following problem. Given an instance (P, aP, bP, cP, r) for some $a, b, c, r \in_R Z_q^*$ and a one way hash function $H : G_2 \rightarrow Z_q^*$, decide whether $r = H(e(P, P)^{abc}) \bmod q$. This problem is termed Decision Hash Bilinear Diffie-Hellman (DHBDH) problem in [2] and is a combination of the bilinear Diffie-Hellman(BDH) problem and a variation of the hash Diffie-Hellman(HDH) problem. The DHBDH assumption is that there exists no probabilistic, polynomial time, 0/1-valued algorithm which can solve the DHBDH problem with non-negligible probability of success.

2.4 Multi-signatures

Multi-signatures allow a group of users to sign a message, such that a verifier can verify that all users indeed signed the message. We use the multi-signatures presented by Boldyreva in [4] which is based on the Boneh-Lynn-Shacham [3] (BLS) pairing based short signature. Formally, a multi-signature scheme consists of three algorithms $\mathsf{MSig} = (\mathcal{MK}, \mathcal{MS}, \mathcal{MV})$, where $\mathcal{MK}$ is the key generation algorithm; $\mathcal{MS}$ is the signature generation algorithm and $\mathcal{MV}$ is the signature verification algorithm. We denote by $\mathsf{Succ}_{\mathsf{DSig}}(t)$ the maximum success probability of any adversary running in time t to forge signatures for a standard digital

signature scheme $\mathsf{DSig} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$. Similarly, by $\mathsf{Succ}_{\mathsf{MSig}}(t)$ the maximum success probability of any adversary running in time t to break the multi-signature scheme MSig based on DSig.

3 Dynamic Group Key Agreement Protocol

Our protocol extends the tree-based multi-party group key agreement protocols of [2], [6] to dynamic case where a user can leave or join the group.

Suppose a set of n users $\mathcal{P} = \{U_1, U_2, \ldots, U_n\}$ wish to agree upon a secret key. Let US be a subset of users. Quite often, we identify a user with its instance during the execution of a protocol. In case US is a singleton set, we will identify US with the instance it contains. Each user set US has a representative $\mathsf{Rep}(\mathsf{US})$ and for the sake of concreteness we take $\mathsf{Rep}(\mathsf{US}) = U_j$ where $j = \min\{k : \Pi_{U_k}^{d_k} \in \mathsf{US}\}$. We use the notation $A[1, \ldots, n]$ for an array of n elements $A_1, \ldots, A_n$ and write $A[i]$ or A_i to denote the ith element of array $A[\,]$. Let $G_1 = \langle P \rangle, G_2$ (groups of prime order q) and $e(,)$ be as described in Section 2.1. We choose a hash function $H : G_2 \rightarrow Z_q^*$. The public parameters are $\mathsf{params} = (G_1, G_2, e, q, P, H)$. Each user $U_i \in \mathcal{P}$ chooses $s_i \in Z_q^*$ at random which it uses as its ephemeral key. These keys are session specific and determine the final common key for a session.

3.1 Unauthenticated Key Agreement Protocol of [2]

We present an informal description of the unauthennticated protocol of [2]. Security of our dynamic key agreement protocol relies on the security of this scheme.

Let $p = \lfloor \frac{n}{3} \rfloor$ and $r = n \bmod 3$. The set of users participating in a session is partitioned into three user sets $\mathsf{US}_1, \mathsf{US}_2, \mathsf{US}_3$ with respective cardinalities being p, p, p if $r = 0$; $p, p, p + 1$ if $r = 1$; and $p, p + 1, p + 1$ if $r = 2$. This top down recursive procedure $\mathsf{KeyAgreement}$ is invoked for further partitioning to obtain a ternary tree structure. The lowest level 0 consists of singleton users having a secret key. We invoke $\mathsf{CombineTwo}$, a key agreement protocol for two user sets and $\mathsf{CombineThree}$, a key agreement protocol for three user sets in the key tree thus obtained. For more details, see [2], [7].

All communications are done by representatives and users in each user set have a common agreed key. In $\mathsf{CombineThree}$, a, b, c respectively are the common agreed key of user sets A, B, C. Representative of user set A sends aP to both the user sets B, C. Similarly, representative of B sends bP to both A, C and representative of C sends cP to both A, B. After these communications, each user can compute the common agreed key $H(e(P, P)^{abc})$. In $\mathsf{CombineTwo}$, users in user set A has common agreed key a, users in user set B has common agreed key b. Representative of A sends aP to user set B and representative of B sends bP to user set A. Besides representative of user set A generates a random key $\hat{a} \in Z_q^*$ and sends $\hat{a}P$ to all the users in both A, B. After these communications, each user can compute the common agreed key $H(e(P, P)^{a\hat{a}b})$.

3.2 Authenticated Key Agreement Protocol of [6]

This protocol incorporates secure signature based authentication mechanism into the unauthenticated protocol of Barua, Dutta, Sarkar [6]. Each user U_i chooses a signing and a verification key sk_i (or sk_{U_i}) and pk_i (or pk_{U_i}) respectively as part of the basic signature scheme DSig and multi-signature scheme MSig.

Besides, this authentication mechanism uses a variable, partial session identity $\mathsf{psid}_{U_{i_j}}^{d_j}$ for each instance $\Pi_{U_{i_j}}^{d_j}$ which is initially set to be $\{(U_{i_j}, d_j)\}$ and after completion of the session, $\mathsf{psid}_{U_{i_j}}^{d_j} = \mathsf{sid}_{U_{i_j}}^{d_j} = \{(U_{i_1}, d_1), \ldots, (U_{i_k}, d_k)\}$ where instances $\Pi_{U_{i_1}}^{d_1} \ldots, \Pi_{U_{i_k}}^{d_k}$ are involved in this session. The session-identity $\mathsf{sid}_{U_{i_j}}^{d_j}$ uniquely identifies the session and is same for all instances participating in this session. The authenticated protocol consists of an algorithm for 2 party authenticated key agreement $\mathsf{AuthCombTwo}$, a 3 party authenticated key agreement $\mathsf{AuthCombThree\text{-}A}$ and an authenticated key agreement $\mathsf{AuthCombThree\text{-}B}$ among three user sets. These procedures are invoked instead of $\mathsf{CombineTwo}$ and $\mathsf{CombineThree}$ in the key tree obtained by the procedure $\mathsf{KeyAgreement}$ in the unauthenticated protocol described above. We discuss $\mathsf{AuthCombTwo}$, $\mathsf{AuthCombThree\text{-}A}$, $\mathsf{AuthCombThree\text{-}B}$ informally and refer the reader to [6], [7] for details.

In $\mathsf{AuthCombTwo}$, user instance $\Pi_{U_1}^{d_1}$ has a secret key s_1 with partial session-identity $\mathsf{psid}_{U_1}^{d_1} = \{(U_1, d_1)\}$. Besides it generates a random key $s \in Z_q^*$, computes a signature σ_1 using basic signature scheme DSig on $m_1 = (s_1 P, sP)$ and sends $U_1|1|m_1|\sigma_1$ to user instance $\Pi_{U_2}^{d_2}$. Similarly, user instance $\Pi_{U_2}^{d_2}$ with secret key s_2 and partial session-identity $\mathsf{psid}_{U_2}^{d_2} = \{(U_2, d_2)\}$ computes a basic signature σ_2 on $m_2 = s_2 P$ and sends $U_2|1|m_2|\sigma_2$ to user instance $\Pi_{U_1}^{d_1}$. On receiving the message $U_2|1|m_2|\sigma_2$, user U_1 verifies σ_2 on $U_2|1|m_2$ according to the verification algorithm $\mathcal{V}$ of DSig. If verification fails, it sets $\mathsf{acc}_{U_1}^{d_1} = 0$, $\mathsf{sk}_{U_1}^{d_1} = \mathsf{NULL}$ annd aborts. Otherwise it sets $\mathsf{psid}_{U_1}^{d_1} = \mathsf{psid}_{U_1}^{d_1} \cup \{(U_2, d_2)\}$ and computes the key $H(e(P,P)^{ss_1s_2})$. In a similar way, user U_2 performs verification σ_1 on $U_1|1|m_1$ and computes the key only if verification succeeds. Note that at the end, user instances $\Pi_{U_1}^{d_1}, \Pi_{U_2}^{d_2}$ have the same partial session-identity : $\mathsf{psid}_{U_1}^{d_1} = \mathsf{psid}_{U_1}^{d_1} \cup \mathsf{psid}_{U_2}^{d_2} = \mathsf{psid}_{U_2}^{d_2}$. An analogus description holds for $\mathsf{AuthCombThree\text{-}A}$. The algorithm $\mathsf{AuthCombThree\text{-}B}$ performs key agreement among three user sets $\mathsf{US}_1, \mathsf{US}_2, \mathsf{US}_3$ as follows: Suppose $\Pi_{U_1}^{d_1}$ is the representative of user set US_1 and users in this set have a common agreed key s_1. Similarly, $\Pi_{U_2}^{d_2}$, s_2 are those for user set US_2 and $\Pi_{U_3}^{d_3}$, s_3 for user set US_3. Let $m_1 = \mathsf{psid}_{U_1}^{d_1}|t_1|s_1 P$, where t_1 is the next expected message number to be sent by $\Pi_{U_1}^{d_1}$. For each user V in user set US_1, $\mathsf{psid}_V^{d_V}|t_1|s_1 P$ is same as m_1. Each user $V \in \mathsf{US}_1$ computes a basic signature σ_V on m_1 using the scheme DSig and sends (V, σ_V) to $\Pi_{U_1}^{d_1}$. After accumulating these basic signatures, the representative $\Pi_{U_1}^{d_1}$ constructs the multi-signature msig_1 on message m_1 using the scheme MSig and sends $m_1|\mathsf{msig}_1$ to $\mathsf{US}_2 \cup \mathsf{US}_3$. Similarly, representative $\Pi_{U_2}^{d_2}$ of user set US_2 sends $m_2|\mathsf{msig}_2$ to $\mathsf{US}_1 \cup \mathsf{US}_3$ and representative $\Pi_{U_3}^{d_3}$ of user set US_3 sends $m_3|\mathsf{msig}_3$ to $\mathsf{US}_1 \cup \mathsf{US}_2$ where $m_2 = \mathsf{psid}_{U_2}^{d_2}|t_2|s_2 P$ and

$m_3 = \mathsf{psid}_{U_3}^{d_3}|t_3|s_3P$, t_2, t_3 being the next expected message number to be sent by $\Pi_{U_2}^{d_2}$, $\Pi_{U_3}^{d_3}$ respectively.

For the ease of discussion, we define a variable $\mathsf{First}(\mathsf{psid}_U^i)$ to be the set $\{U_{i_1}, \ldots, U_{i_k}\}$ where $\mathsf{psid}_U^i = \{(U_{i_1}, d_{i_1}), \ldots, (U_{i_k}, d_{i_k})\}$. Now on receipt of messages $m_2|\mathsf{msig}_2$ and $m_3|\mathsf{msig}_3$, each user instance $\Pi_V^{d_V} \in \mathsf{US}_1$ (1) checks $\mathsf{First}(\mathsf{psid}_{U_2}^{d_2}) \subseteq \mathsf{pid}_V^{d_V}$ and $\mathsf{First}(\mathsf{psid}_{U_3}^{d_3}) \subseteq \mathsf{pid}_V^{d_V}$; (2) verifies t_2, t_3 are the next expected message number to be sent by $\Pi_{U_2}^{d_2}$, $\Pi_{U_3}^{d_3}$ respectively; (3) verifies $\mathsf{msig}_2, \mathsf{msig}_3$ are multi–signatures on m_2, m_3 respectively. If any of these verification fails, $\Pi_V^{d_V}$ sets $\mathsf{acc}_V^{d_V} = 0$ and $\mathsf{sk}_V^{d_V} = \mathsf{NULL}$ and aborts. Otherwise computes the common key $H(e(P,P)^{s_1 s_2 s_3})$ and sets $\mathsf{psid}_V^{d_V} = \mathsf{psid}_V^{d_V} \cup \mathsf{psid}_{U_2}^{d_2} \cup \mathsf{psid}_{U_3}^{d_3}$.

Similar verifications are done by each user in user sets US_2 and US_3. Common key $H(e(P,P)^{s_1 s_2 s_3})$ is computed only if verification holds. Note that at the end of an honest execution of this protocol, each user in the group $\mathsf{US}_1 \cup \mathsf{US}_2 \cup \mathsf{US}_3$ has a common partial session-identity.

3.3 Proposed Dynamic Key Agreement Protocol

Dynamic key agreement consists of a key agreement protocol together with two additional algorithms, Insert and Delete. The procedure Insert enables a user to join a group. A user can leave a group by invoking the procedure Delete. Now, we describe protocols for insertion and deletion for the above static tree-based authenticated protocol. Our protocol design makes an optimal use of the data precomputed in the procedure KeyAgreement. When a user joins or leaves a group, the structure of the key tree is disturbed and requires to be updated for any subsequent join or leave operation. Maintaining the tree structure of the key agreement protocol is a crucial part of our scheme. We refer this as the preservation of the structure of the procedure KeyAgreement.

Suppose we have a keytree T of n users $\{1, 2, \ldots, n\}$ according to the key agreement of [2] with $k = R(n)$ rounds. For the sake of easy description, we take the user set $\mathcal{P} = \{1, \ldots, n\}$ instead of the set $\{U_1, \ldots, U_n\}$ and introduce some more notations. For $1 \leq l \leq k$, let $U_i^{(l)}$ be the i-th user set at level l and $s_i^{(l)}$ be the common agreed key of users in the user set $U_i^{(l)}$ at level l. Initially, $U_i^{(0)} = \{i\}$, $s_i^{(0)}$ is the private key randomly chosen by user i from Z_q^*.

Insertion Let a new user $\{u\}$ with private key x joins the group $\{1, \ldots, n\}$. He joins the tree in such a way that the structure of KeyAgreement is still preserved and updation of key path is optimal in the sense that minimum updation or recomputation is required. For instance, consider a group key agreement with $n = 10$ members. In this case, the root node will have 3 subtrees, with the left and middle subtrees having 3 leaves each and the right subtree having 4 leaves. Now suppose a new user wants to join this group. He cannot join the first subgroup (of subusers) since this is contrary to the way we partition the user sets. So the entire group of users will have to be repartitioned. Similarly, he cannot join the third subgroup (of 4 users) without causing repartitioning.

But if he joins the second subtree, then there is no need of repartitioning and so key updation is minimal and is only along a single path. This is illustrated in Figure 1. The following Lemma (proof in the full version [7]) determines uniquely such a path that we call the optimal path of joining of the new user.

Lemma 31 *For $1 \leq l \leq k$, $k = R(n)$, define the following:*
$i_l :=$ *the index of the node at level l whose subtree will contain the new user $\{u\}$ as a leaf,*
$\eta_l :=$ *the number of leaf nodes in the subtree at i_l,*
$N_l :=$ *number of leaf nodes to the left of node i_l and*
$r_l = \eta_l \bmod 3$.
Clearly $i_k = 1; \eta_k = n; r_k = n \bmod 3; N_k = 0$. Then, for $2 \leq l \leq k$, we have $i_{l-1} = 3i_l - r_l$; $\eta_{l-1} = \lfloor \frac{\eta_l}{3} \rfloor$; and $N_{l-1} = N_l + (2 - r_l)\eta_{l-1}$. Thus user u joins the subgroup at i_1 and $i_k, i_{k-1}, \ldots, i_1$ determine the optimal path along which the subgroup keys needs to be updated or recomputed.

We now describe our protocol Insert. This algorithm invokes a procedure FindKeyPath to find the optimal key path path of joining of a new member and updates path according to the algorithm UpdateKeyPath. The new user who has permission to join the group for a group key agreement computes it's optimal key path of joining $i_k, \ldots, i_1; N_k, \ldots, N_1$ by using algorithm FindKeyPath, communicate this to all other members of the group and invokes algorithm UpdateKeyPath to update this path. The formal description of the procedures FindKeyPath and UpdateKeyPath are in the full version [7].

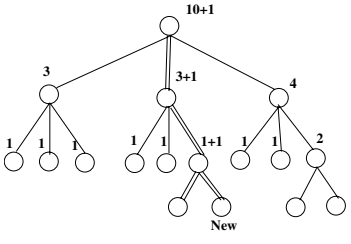


Fig. 1. procedure Insert

The algorithm UpdateKeyPath works as follows to update keys in level 1 on joining of the new user. In the key tree T with n users, the number of children of the node i_1 (node at level 1) in the optimal key path is either 1 or 2 or 3. If i_1 has 1 leaf node, then the user corresponding to this leaf node chooses a new private key, agree upon a common key with the new user by invoking algorithm AuthCombTwo and the corresponding user set for level 1 is modified to a set that includes these two users. In case i_1 has 2 leaf nodes, the users corresponding to these leaves choose new private keys, a new user set for level 1 is constructed that contains these two users and the new user and algorithm AuthCombThree-A is invoked to agree upon a common key among them. If i_1 has

3 leaves, then the users corresponding to these leaves choose new private keys. The rightmost user agree upon a common key with the new user by invoking algorithm AuthCombTwo and constructs a new user set that consists of the new user and itself. Then AuthCombThree-B is invoked for this new user set and the the other two leaves of i_1 to agree upon a common key. Finally the corresponding user set for level 1 is modified to a set that includes these users.

The subsequent user sets are accordingly changed by algorithm UpdateKey-Path and key updates in level $l+1$ ($1 \leq l \leq k-1$) are done by invoking algorithm AuthCombThree-B among the three user sets which are subtrees of node i_{l+1}. The modified user set corresponding to the node i_l invokes AuthCombThree-B to agree upon a common key with the user sets corresponding to the other two subtrees (siblings of i_l) of node i_{l+1} and a new user set for level $l+1$ is constructed that is the union of these three user sets. We proceed in this way and finally a common key is agreed among all the $n+1$ users. At the end, we newly index the members (leaves) as $\{1, 2, \ldots, n+1\}$.

Deletion Suppose key tree T with n leaf nodes $\{1, 2, \ldots, n\}$ has the tree structure used in the procedure KeyAgreement and suppose a member j_0, $1 \leq j_0 \leq n$, wants to leave the group. For this we first introduce a procedure Extract which outputs the identity or index of a leaf node in T such that the structure of KeyAgreement is preserved in the tree after removal of this node. We take a designated user, called group controller (GC) to initiate the operation Delete. To be specific, we take one sibling of the node leaving the group as the GC which is trusted only for this purpose.

The procedure Extract works as follows (for formal description, see the full paper [7]). If all the three subtrees T_L, T_M, T_R of the tree T have equal number of leaf nodes, then removal of a leaf node from the leftmost subtree T_L will not disturb the tree structure and so we can extract a leaf node from T_L. Similarly, if the number of leaves in both T_L, T_M are same, say p, and that of right subtree T_R is $p+1$, then we can extract a leaf from T_R without disturbing the tree structure. If the number of leaves in T_M, T_R are same, say, $p+1$ and that of T_L is p, then we can extract a leaf from T_M retaining the tree structure. We recursively apply this procedure on T_L, T_R or T_M chosen in this manner and finally reach a leaf node. The index of the user corresponding to this leaf node is outputed to the GC.

Now if a user corresponding to leaf j_0 leaves the group, the tree structure is disturbed. We first find the highest level, say i, for which the subtree rooted at the internal node at, say j_i, lacks the tree structure. Consequently, j_i is the root of the highest level subtree with disturbed tree structure on removal of j_0. To retain the tree structure, we may need to extract suitably a leaf node l_0 from the tree T in such a way that the key updates required are minimal. We do this by using an algorithm FindExtractNode that uses the procedure Extract as a subroutine and removes the leaving member j_0, and replaces it with the extracted node, still preserving the structure of KeyAgreement in the resulting key tree. This algorithm outputs the index of the extracted leaf node l_0 to GC

and also the index of the internal node j_i. The procedure FindExtractNode is formally described in the full version [7].

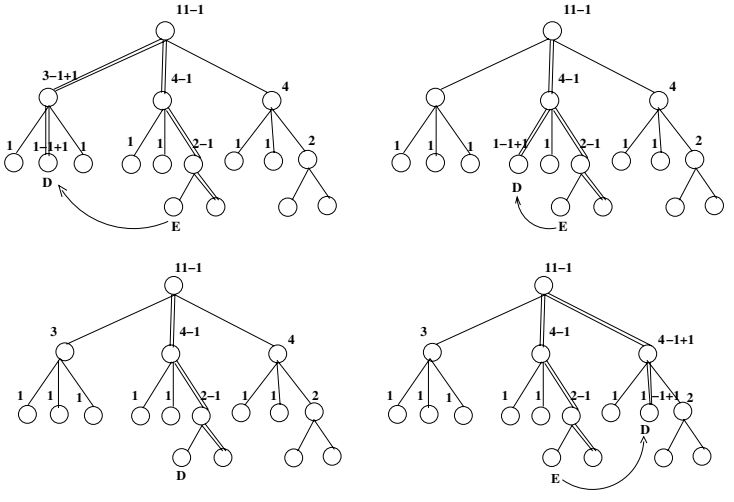


Fig. 2. Different cases of **procedure** Delete with $n = 11$ (D denotes the node to be deleted and E denote the node to be extracted to maintain the key structure)

Next we describe the algorithm Delete below (formal description is in the full version [7]). Our algorithm Delete invokes FindExtractNode to obtain the index l_0 of the node to be extracted (if required), finds from the tree T the path from root to the parent of the leaving leaf node j_0 and also the path from root to the parent of the extracted leaf node l_0. Two new user sets are constructed in level 1: one user set includes the user corresponding to l_0 together with the users corresponding to brothers of j_0 and another user set includes only the users corresponding to brothers of l_0. All the users in these two new user sets choose new private keys. The subsequent higher level user sets are modified accordingly and appropriate algorithms AuthCombTwo, AuthCombThree-A or AuthCombThree-B are invoked for successive key agreements in the key tree. At the end of the procedure Delete, we newly index the users (leaves) as $\{1, 2, \ldots, n-1\}$. Some particular cases are shown in Figure 2.

4 Security Analysis

We will show that our dynamic authenticated key agreement protocol DAP is secure in the model as described in Subsection 2.2. In fact, we can convert any active adversary attacking the protocol DAP into a passive adversary attacking the unauthenticated protocol UP assuming that both DSig and MSig are secure and DHBDH problem is hard. No Corrupt query appears since long term secret keys are not used. So our protocol trivially achieves forward secrecy.

Theorem 4.1 *[2] The group key agreement protocol* UP *described in Section 3.1 is secure against passive adversaries provided DHBDH problem is hard.*

Theorem 4.2 *[6] The group key agreement protocol* AP *described in Section 3.2 satisfies the following:* $\mathsf{Adv}^{\mathsf{AKA}}_{\mathsf{AP}}(t, q_E, q_S) \leq \mathsf{Adv}^{\mathsf{KA}}_{\mathsf{UP}}(t', q_E+q_S/2)+|\mathcal{P}|\ \mathsf{Succ}_{\mathsf{DSig}}(t')+ |\mathcal{P}|\ \mathsf{Succ}_{\mathsf{MSig}}(t')$ *where* $t' \leq t + (|\mathcal{P}|q_E + q_S)t_{\mathsf{AP}}$*, where* t_{AP} *is the time required for execution of* AP *by any one of the users.*

Theorem 4.3 *The dynamic group key agreement protocol* DAP *described in Section 3.3 satifies the following:* $\mathsf{Adv}^{\mathsf{AKA}}_{\mathsf{DAP}}(t, q_E, q_J, q_L, q_S) \leq \mathsf{Adv}^{\mathsf{KA}}_{\mathsf{UP}}(t', q_E+(q_J+q_L+ q_S)/2)+|\mathcal{P}|\ \mathsf{Succ}_{\mathsf{DSig}}(t')+|\mathcal{P}|\ \mathsf{Succ}_{\mathsf{MSig}}(t')$ *where* $t' \leq t+(|\mathcal{P}|q_E+q_J+q_L+q_S)t_{\mathsf{DAP}}$*, where* t_{DAP} *is the time required for execution of* DAP *by any one of the users,* q_E, q_J, q_L *and* q_S *are respectively the maximum number of execute, join, leave and send queries that an adversary can make.*

Proof (Sketch) : Let $\mathcal{A}'$ be an adversary which attacks the dynamic authenticated protocol DAP. Using this we construct an adversary $\mathcal{A}$ which attacks the unauthenticated protocol UP. As in [6], we have the following claim.

Claim : Let Forge be the event that a signature (either of DSig or of MSig) is forged by $\mathcal{A}'$. Then $\mathsf{Prob}[\mathsf{Forge}] \leq |\mathcal{P}|\ \mathsf{Succ}_{\mathsf{MSig}}(t') + |\mathcal{P}|\ \mathsf{Succ}_{\mathsf{DSig}}(t')$.

Adversary $\mathcal{A}$ maintains a list Tlist to store pairs of session IDs and transcripts. It also uses two lists Jlist and Llist to be specified later. Adversary $\mathcal{A}$ generates the verification/signing keys pk_U, sk_U for each user $U \in \mathcal{P}$ and gives the verification keys to $\mathcal{A}'$. If ever the event Forge occurs, adversary $\mathcal{A}$ aborts and outputs a random bit. Otherwise, $\mathcal{A}$ outputs whatever bit is eventually output by $\mathcal{A}'$. Note that since the signing and verification keys are generated by $\mathcal{A}$, it can detect occurrence of the event Forge. $\mathcal{A}$ simulates the oracle queries of $\mathcal{A}'$ using its own queries to the Execute oracle. We provide details below.

Execute and Send queries: These queries are simulated as in [6]. Apart from the usual send queries, there are two special type of send queries, Send_J and Send_L. If an unused instance Π^d_U wants to join the group $\Pi^{d_1}_{U_{i_1}}, \ldots, \Pi^{d_k}_{U_{i_k}}$, then $\mathcal{A}'$ will make $\mathsf{Send}_J(U, d, \langle U_{i_1}, \ldots, U_{i_k}\rangle)$ query. This query initiates $\mathsf{Join}(\{(U_{i_1}, d_1), \ldots, (U_{i_k}, d_k)\}, (U, d))$ query . $\mathcal{A}$ first finds a unique entry of the form (S, T) in Tlist with $S = \{(U_{i_1}, d_1), \ldots, (U_{i_k}, d_k)\}$. If no such entry, $\mathcal{A}$ makes an execute query to its own execute oracle on S and gets a transcript T. $\mathcal{A}$ then stores $(S, U|d, T)$ in Jlist. Similarly, when $\mathsf{Send}_L(U, d, \langle U_{i_1}, \ldots, U_{i_k}\rangle)$ query is made, $\mathcal{A}$ stores $(S, U|d, T)$ in Llist.

Join queries : Suppose $\mathcal{A}'$ makes a query $\mathsf{Join}(\{(U_{i_1}, d_1), \ldots, (U_{i_k}, d_k)\}, (U, d))$. $\mathcal{A}$ finds an entry of the form $(S, U|d, T)$ in Jlist where $S = \{(U_{i_1}, d_1), \ldots, (U_{i_k}, d_k)\}$. If no such entry, then the adversary $\mathcal{A}'$ is given no output. Otherwise, $\mathcal{A}$ modifies T as follows: $\mathcal{A}$ can find the path of joining of U in the key tree with leafs $U_{i_1}, \ldots, U_{i_k}$ and detect the positions in T where the new messages are to be injected or where the old messages are to be replaced by new messages. $\mathcal{A}$ does these modifications in T according to the unauthenticated version of the algorithm Insert described in Section 3.3 and gets a modified transcript T_M. It then patches appropriate basic signatures and multi-signatures with each message in T_M according to the modifications described in Section 3.2. Thus $\mathcal{A}$ expands the transcript T_M into a transcript T' for DAP. It returns T' to $\mathcal{A}'$.

Leave queries : These queries are simulated as Join queries with modified transcript T_M obtained from unauthenticated transcript T according to the algorithm Delete in Section 3.3.
Reveal/Test queries : Suppose $\mathcal{A}'$ makes the query $\mathsf{Reveal}(U,i)$ or $\mathsf{Test}(U,i)$ for an instance Π_U^i for which $\mathsf{acc}_U^i = 1$. At this point the transcript T' in which Π_U^i participates has already been defined. If T' corresponds to the transcript of the authenticated protocol, then $\mathcal{A}$ finds the unique pair (S,T) in Tlist such that $(U,i) \in S$. Assuming that the event Forge does not occur, T is the unique unauthenticated transcript which corresponds to the transcript T'. Then $\mathcal{A}$ makes the appropriate Reveal or Test query to one of the instances involved in T and returns the result to $\mathcal{A}'$. Otherwise, T' is the transcript for Join or Leave, as the case may be. Since T' has been simulated by $\mathcal{A}$, $\mathcal{A}$ is able to compute the modified session key and hence send an appropriate reply to $\mathcal{A}'$.

As long as Forge does not occur, the above simulation for $\mathcal{A}'$ is perfect. Whenever Forge occurs, adversary $\mathcal{A}$ aborts and outputs a random bit. So $\mathsf{Prob}_{\mathcal{A}',\mathsf{AP}}[\mathsf{Succ}|\mathsf{Forge}] = \frac{1}{2}$. Using this, one can obtain $\mathsf{Adv}_{\mathcal{A},\mathsf{UP}} \geq \mathsf{Adv}_{\mathcal{A}',\mathsf{DAP}} - \mathsf{Prob}[\mathsf{Forge}]$ and finally show that (see the full version [7]) $\mathsf{Adv}^{\mathsf{AKA}}_{\mathsf{DAP}} \leq \mathsf{Adv}^{\mathsf{KA}}_{\mathsf{UP}}(t', q_E + (q_J + q_L + q_S)/2) + \mathsf{Prob}[\mathsf{Forge}]$. This yields the statement of the theorem. □

References

1. P. S. L. M. Barreto, H. Y. Kim and M. Scott. *Efficient algorithms for pairing-based cryptosystems*. Advances in Cryptology - Crypto'02, LNCS 2442, pp. 354-368, Springer 2002.
2. R.Barua, R.Dutta, P.Sarkar. *Extending Joux Protocol to MultiParty Key Agreement*. Proc. of Indocrypt'03, LNCS 2905, pp. 205-217, Springer 2001.
3. D. Boneh, B. Lynn, and H. Shacham. *Short Signature from Weil Pairing*. Proc. of Asiacrypt'01, LNCS 2248, pp. 514-532, Springer 2001.
4. A. Boldyreva. *Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme*. Proc. of PKC'03, LNCS 2567, pp. 31-46, Springer 2003.
5. E. Bresson, O. Chevassut, and D. Pointcheval. *Dynamic Group Diffie-Hellman Key Exchange under Standard Assumptions*. Advances in Cryptology - Eurocrypt '02, LNCS 2332, pp. 321-336, Springer 2002.
6. R. Dutta, R. Barua and P. Sarkar. *Provably Secure Authenticated Tree Based Group Key Agreement*. Proc. of ICICS'04, LNCS 3269, pp. 92-104, Springer 2004.
7. R. Dutta and R. Barua. *Dynamic Group Key Agreement in Tree-Based setting*. Available at http://www.cse.iitk.ac.in/users/iriss05/r_dutta.pdf.
8. A. Joux. *A One Round Protocol for Tripartite Diffie-Hellman*. Proc. of ANTS IV, LNCS 1838, pp. 385-394, Springer 2000.
9. J. Katz and M. Yung. *Scalable Protocols for Authenticated Group Key Exchange*. Advances in Cryptology - Crypto'03, LNCS 2729, pp. 110-125, Springer 2003.
10. Y. Kim, A. Perrig, and G. Tsudik. *Tree based Group Key Agreement*. Available at http://eprint.iacr.org/2002/009.
11. H. J. Kim, S. M. Lee and D. H. Lee. *Constant-Round Authenticated Group Key Exchange for Dynamic Groups*. Proc. of Asiacrypt'04, LNCS 3329, pp. 245-259, Springer 2004.

Immediate Data Authentication for Multicast in Resource Constrained Network

C.K. Wong and Agnes Chan

Northeastern University, Boston MA 02115, USA
{ckwong,ahchan}@ccs.neu.edu

Abstract. In this paper, we consider the problem of authentication of multicast data. The TESLA scheme was introduced to provide data authentication for multicast communication over lossy channels. Later, TESLA was further improved to offer immediate authentication of packets and fortifications against denial-of-service attacks. The improved TESLA scheme is efficient and applicable to mobile resource-constrained receivers for authentication of multicast data. The resource limitation of mobile resource-constrained receivers gives additional challenges to multicast authentication. In this paper, a denial-of-service attack called the Random-Substitution attack is presented. We present a new scheme that can provide immediate packet authentication and deter the Random-Substitution attack. It is also robust against packet losses. In addition, the new scheme allows a receiver to immediately authenticate all packets upon arrival, when the receiver joins the multicast communication. Hence, the new scheme offers a practical multicast authentication solution for resource-constrained receivers.

1 Introduction

When a sender wants to simultaneously transmit a sequence of data packets to multiple receivers, multicast communication can be used. Multicast is an efficient means of communication and particularly well suited to applications such as audio and video streaming. Over the past decade, multicast communication has gained substantial attention among researchers. In an open environment such as the Internet, the communication channels generally cannot be assumed secure, and data received through insecure channels can be fabricated and altered. In unicast communication where sender and receiver share a unique secret key, message authentication code (MAC) provides an efficient means for verifying the authenticity and integrity of data. However, in multicast communication, the same common key is shared by all receivers. Any compromised receiver knowing the secret key can compute the MAC and impersonate the sender. Thus, the same approach cannot be directly applied to multicast communication.

A scheme called TESLA [3] has been proposed to provide data authentication for multicast communication over lossy channels. It is computationally efficient and is robust against packet losses. However, a drawback of TESLA is that packets cannot be immediately authenticated. Moreover, delayed packet authentication forces receivers to buffer incoming packets until they can be authenticated.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 113–121, 2005.

The buffering requirement introduces a vulnerability to denial-of-service attacks, as an adversary can flood bogus packets to buffer-limited receivers. An improved TESLA scheme [4] is proposed later, offering immediate packet authentication and fortifications against denial-of-service attacks.

The resource limitation of mobile resource-constrained receivers gives additional challenges to multicast authentication. In this paper, we present the Random-Substitution attack that wastes computation and storage resources of receivers. To guard against the Random-Substitution attack, we propose a new scheme that can provide immediate packet authentication and is robust against packet losses. Our new scheme is built on the improved TESLA scheme, using a different construction of hash values in order to deter the Random-Substitution attack. The design of our scheme assumes that data are stored and known to the sender in advance. Examples of such applications can be found in multimedia services for mobile users.

To provide a receiver the ability to immediately authenticate packets when the receiver joins the multicast communication, we introduce an additional hash value per packet. This additional field also provides a better sustainability of immediate authentication against packet losses.

The paper is organized as follows. In section 2, we give a review of the improved TESLA scheme. The Random-Substitution attack is presented in section 3. Next, we present the proposed scheme in section 4. The immunity of the proposed scheme to the Random-Substitution attack is discussed in section 5. Finally, we conclude the paper in section 6.

2 Review of the Improved TESLA Scheme

The original TESLA scheme [3] has drawbacks including delayed authentication of packets and vulnerabilities to denial-of-service attacks. Later, improvements to TESLA are proposed [4], offering immediate authentication of packets and fortifications against denial-of-service attacks. In this section, we review the improved TESLA scheme.

2.1 Overview

In the multicast communication model, there is a sender who multicasts messages M_i to multiple receivers [4,3]. Each message M_i is carried by a packet P_i. In each time interval, the sender may send zero or multiple packets. When a receiver receives a new packet P_i, the receiver will verify the authenticity and integrity of P_i. The first packet is authenticated via a digital scheme such as RSA [5] or DSA [6]. Each packet carries the hash value of the message of a future packet. Thus, if the current packet is authentic, then the hash value it carries is authentic and can be used to provide immediate authentication of a future packet.

The improved TESLA scheme can be described in four stages: sender setup, bootstrapping a new receiver, sending authenticated packets, and verifying received packets.

2.2 Sender Setup

The improved TESLA scheme makes use of a key chain analogous to the one-way chain introduced by Lamport [2] and the S/KEY authentication scheme [1]. It also assumes that time is split into equal intervals I_i, with starting time denoted by T_i. Before sending the first message, the sender determines the sending duration (possibly infinite), the interval duration T_{int}, and the number N of keys on the key chain. Next, the sender generates a random key K_N as the last key of the key chain, and computes the entire key chain using a pseudo-random function F. Each element of the key chain is defined as $K_i = F(K_{i+1})$, where $0 \leq i < N$. Each key K_i of the key chain corresponds to one interval I_i, and K_i is used during the interval I_i. A second pseudo-random function F' is applied to each K_i to derive the key which is used to compute the MAC of messages in each interval. Hence, $K'_i = F'(K_i)$ for $0 \leq i \leq N$. Finally, a cryptographic hash function H and a message authentication code scheme are selected, where $MAC(k, m)$ denotes the message authentication code of message m under key k.

2.3 Bootstrapping a New Receiver

The initial packet sent to a dynamically added new receiver is authenticated via a digital signature scheme, such as RSA [5] or DSA [6]. It also contains the following information about the time intervals and key chain:

- The beginning time of a specific interval T_j, along with its id I_j
- The interval duration T_{int}
- The key disclosure delay d (in interval unit)
- A commitment to the key chain K_i ($i < j - d$ where j is the current interval index)

2.4 Sending Authenticated Packets

In each interval, the sender may send zero or multiple packets, and the corresponding key is used to compute the MAC of all these packets. Thus, the sender can send packets at any rate and adapt the sending rate dynamically. Furthermore, the key is kept secret for $d - 1$ future intervals. Packets sent in interval I_j disclose the key K_{j-d}, which allows receivers to verify the authenticity of the packets sent in interval I_{j-d}.

For simplicity, it is assumed that the sender sends out a constant number v of packets per time interval. To support immediate authentication of packets, the sender constructs and sends the packet for the message M_j in time interval T_i as follows:

1. construct $D_j = M_j || H(M_{j+vd})$, where $||$ denotes message concatenation,
2. determine from the current time t the index of the current time interval $i = \lfloor \frac{t-T_0}{T_{int}} \rfloor$,
3. multicast the packet $P_j = (D_j, MAC(K'_i, D_j), K_{i-d}, i)$.

Remark: The time interval index i was omitted in P_j in [4]. We believe that i has to be explicit in P_j as originally proposed in [3].

2.5 Verifying Received Packets

The initial packet is verified according to the signature scheme of the sender. Since signature verification is usually expensive, it should be avoided for verification of other packets.

A received packet can be immediately authenticated if its hash value is contained in a previously received packet. If a received packet cannot be immediately authenticated, it needs to be buffered and verified later. Since each key K_i will be disclosed eventually, a receiver must verify the security condition stated below for each packet it receives.

Security condition: A packet arrived *safely*, if the receiver is assured that the sender cannot yet be in the time interval in which the corresponding key is disclosed.

Assume that the packet is sent in interval I_i and received at receiver's local time t, where t denotes the time after adjustment for time synchronization error is made [4]. The security condition is satisfied if $\lfloor \frac{t-T_0}{T_{int}} \rfloor < i + d$, where i is the index of the interval I_i and d is the key disclosure delay.

The receiver verifies a packet $P_j = (D_j, MAC(K'_i, D_j), K_{i-d}, i)$ that arrives at time t as follows:

1. If P_j contains a disclosed key K_{i-d}, the receiver checks whether K_{i-d} has been previously received. If not, then the receiver will compute keys from K_{i-d} to authenticate packets that are buffered to be verified. Let K_v denote the key in the key chain that the receiver has most recently received. Then $v < i - d$. The receiver verifies that $K_v = F^{i-d-v}(K_{i-d})$. If the condition is satisfied, the receiver updates the key chain. For each new key K_w in the key chain, the receiver computes $K'_w = F'(K_w)$, which can be used to authenticate the buffered packets.
2. The receiver verifies the security condition and rejects P_j if the condition is not satisfied.
3. If P_j satisfies the security condition, then there are two cases:
   Case (1), the receiver has previously received the packet P_{j-vd}. It means that if P_{j-vd} is authentic, then the $H(M_j)$ of P_{j-vd} is also authentic and therefore the data M_j is immediately authenticated by using the authentic P_{j-vd}.
   Case (2), the packet P_{j-vd} is lost or previously dropped by the receiver. Then, it means that P_j cannot be immediately authenticated. Therefore, it needs to be buffered until its MAC value can be verified later in a time interval.

3 A Denial-of-Service Attack

In this section, a denial-of-service attack called the Random-Substitution attack is presented. We show that the Random-Substitution attack can waste the computation and storage resources of a victim receiver.

For our study of denial-of-service attacks, we assume that an adversary is capable of :

1. eavesdropping packets sent from a legitimate sender
2. creating and injecting packets to receivers

but cannot block packets.

3.1 The Random-Substitution Attack

Although a receiver can immediately authenticate the message M_j of an incoming packet P_j in the improved TESLA scheme, the receiver cannot immediately verify the authenticity of the hash value $H(M_{j+vd})$ of future message M_{j+vd}, and the $MAC(K'_i, D_j)$. Then, the adversary can create valid bogus packets and flood receivers as shown in the Random-Substitution attack.

Random-Substitution attack: Since K_{i-d} and M_j are plaintexts and known to the adversary, the adversary can generate a bogus packet $\tilde{P}_j = (\tilde{D}_j, r_2, K_{i-d}, i)$, where $\tilde{D}_j = M_j||r_1$, r_1 and r_2 are random values chosen by the adversary. The adversary can flood a victim receiver with these bogus packets. Note that when the victim receives a bogus packet $\tilde{P}_j$, the receiver will verify M_j and $H(M_j)$, and thus it is accepted. Also, note that the receiver cannot distinguish a bogus packet $\tilde{P}_j$ from an authentic packet P_j, because at the current moment the receiver does not know the correct key and cannot distinguish r_2 from a valid MAC code for $\tilde{P}_j$. Therefore, the receiver cannot decide whether to store r_1 from $\tilde{P}_j$ or $H(M_{j+vd})$ from P_j for the immediate authentication of packet P_{j+vd} in the future. So, the receiver must store both r_1 and $H(M_{j+vd})$. Similarly, the receiver must store both r_2 of the packet $\tilde{P}_j$, and the $MAC(K'_i, D_j)$ value of the packet P_j.

Thus, an adversary can force a receiver to assign new buffer storage for the bogus packets received. Other than wasting storage, the Random-Substitution attack also wastes the computational resources of the victim receiver. A receiver must extract and buffer all the r_1 and r_2 values of bogus packets. For each bogus pair (r_1, r_2) buffered, the receiver must verify its validity by checking if r_2 is a valid MAC code when a future packet such as P_{j+vd} arrives. Until a stored (r_1, r_2) pair is found to pass such a test, the receiver is unable to immediately authenticate P_{j+vd}. So, the average number of MAC operations that a victim receiver must compute is equal to half of the total number of the bogus packets flooded to the victim receiver.

The Random-Substitution attack forces the victim receiver to assign a substantial amount of computation and storage. So, mobile resource-constrained receivers are most affected by the Random-Substitution attack.

4 The Proposed Scheme

In this section, we present a new scheme that is immune to the Random-Substitution attack. The proposed scheme also allows a receiver to immediately authenticate all packets (including the initial sequence of received packets) upon arrival, when the receiver joins the system.

The proposed scheme is built by extending the improved TESLA scheme. We describe the proposed scheme in four stages: sender setup, bootstrapping a new receiver, sending authenticated packets, and verifying received packets.

4.1 Sender Setup

The proposed scheme uses a key chain analogous to the one-way chain introduced by Lamport [2] and the S/KEY authentication scheme [1]. It assumes that time is split into equal intervals I_i, with starting time denoted by T_i. Before sending the first message, the sender determines the sending duration, the interval duration T_{int}, and the number N of keys on the key chain. Next, the sender generates a random key K_N as the last key of the key chain, and computes the entire key chain using a pseudo-random function F. Each element of the key chain is defined as $K_i = F(K_{i+1})$, where $0 \leq i < N$. Each key K_i of the key chain corresponds to one interval I_i, and K_i is used during the interval I_i. A second pseudo-random function F' is applied to each K_i to derive the key which is used to compute the MAC of messages in each interval. Hence, $K'_i = F'(K_i)$ for $0 \leq i \leq N$. Finally, a cryptographic hash function H and a message authentication code scheme are selected, where $MAC(k, m)$ denotes the message authentication code of message m under key k.

4.2 Bootstrapping a New Receiver

The initial packet sent to a new receiver is authenticated via a digital signature scheme. It also contains the following information about the time intervals and key chain:

- The beginning time of a specific interval T_j, along with its id I_j
- The interval duration T_{int}
- The key disclosure delay d (in interval unit)
- A commitment to the key chain K_i ($i < j - d$ where j is the current interval index)

4.3 Sending Authenticated Packets

The proposed scheme is built by extending the improved TESLA scheme. Similar to the improved TESLA scheme, a packet P_j in the proposed scheme carries a hash value β_j. However, different from TESLA, the proposed scheme uses a different construction for the hash value β_j in order to deter the Random-Substitution attack.

Moreover, an additional hash value α_j is introduced to enable a receiver to immediately authenticate all received packets after the initial one, regardless when the receiver joins the system.

We assume that the sender sends out a constant number v of packets per time interval. We also assume that all messages $M_0, M_1, ..., M_r$ are known to the sender in advance. The sender computes the followings for multicasting the messages:

- $\alpha_{r-1} = H(M_r, \beta_r, \alpha_r)$,
- $\alpha_j = H(M_{j+1}, \beta_{j+1}, \alpha_{j+1})$ for $0 \leq j \leq r-2$,
- $\beta_{r-vd-((r-i) \bmod vd)} = H(M_{r-((r-i) \bmod vd)}, \alpha_{r-((r-i) \bmod vd)}, \beta_{r-((r-i) \bmod vd)})$ for $0 \leq i \leq vd-1$,
- $\beta_j = H(M_{j+vd}, \alpha_{j+vd}, \beta_{j+vd})$ for $0 \leq j \leq r-2vd$,
- $\alpha_r = \beta_r = \beta_y = \gamma''$ for $r - vd + 1 \leq y \leq r-1$, where γ'' can be any string, for example, the empty string.

($a \bmod b$ denotes the non-negative integer c such that $0 \leq c \leq b-1$ and b divides $a - c$).

To support immediate authentication of packets, the sender constructs and sends the packet P_j for the message M_j in time interval T_i as follows:

1. construct $D_j = M_j||\alpha_j||\beta_j$, where $||$ denotes message concatenation,
2. determine from the current time t the index of the current time interval $i = \lfloor \frac{t-T_0}{T_{int}} \rfloor$,
3. multicast the packet $P_j = (D_j, MAC(K'_i, D_j), K_{i-d}, i)$.

4.4 Verifying Received Packets

The initial packet is verified according to the signature scheme of the sender. The receiver verifies a packet $P_j = (D_j, MAC(K'_i, D_j), K_{i-d}, i)$ that arrives at time t as follows:

1. If P_j contains a disclosed key K_{i-d}, the receiver checks whether K_{i-d} has been previously received. If not, then the receiver will compute keys from K_{i-d} to authenticate packets that are buffered to be verified. Let K_v denote the key in the key chain that the receiver has most recently received. Then $v < i - d$. The receiver verifies that $K_v = F^{i-d-v}(K_{i-d})$. If the condition is satisfied, the receiver updates the key chain. For each new key K_w in the key chain, the receiver computes $K'_w = F'(K_w)$, which can be used to authenticate the buffered packets.
2. The receiver verifies the security condition and rejects P_j if the condition is not satisfied.
3. If P_j satisfies the security condition, then there are three cases :
   Case (1), the receiver has previously received the packet P_{j-vd}. Let $D_j = M_j||\alpha_j||\beta_j$. It means that if P_{j-vd} is authentic, then the β_{j-vd} of P_{j-vd} is also authentic and therefore the packet P_j is immediately authenticated by using the authentic P_{j-vd}.

Case (2), the receiver has previously received the packet P_{j-1}. Let $D_j = M_j||\alpha_j||\beta_j$. It means that if P_{j-1} is authentic, then the α_{j-1} of P_{j-1} is also authentic and therefore the packet P_j is immediately authenticated by using the authentic P_{j-1}.
Case (3), both packets P_{j-vd} and P_{j-1} are lost or previously dropped by the receiver. It means that P_j cannot be immediately authenticated. Therefore, it needs to be buffered until its MAC value can be verified later in a time interval.

4. Finally, if all verifications pass, the receiver accepts P_j and stores α_j, β_j for future immediate authentication. Moreover, once the receiver has accepted a packet P_j, then any additional incoming packet claiming to be P_j can be immediately discarded and no further inspection is required. The ability to discard all these additional packets prevents the Random-Substitution attack.

A received packet P_{j+vd} can be immediately authenticated unless both P_j and P_{j+vd-1} are lost. Let p be the probability that a packet will be lost, P_{proposed} and P_{TESLA} denote the probability that a received packet can be immediately authenticated in the proposed scheme and the improved TESLA scheme respectively. Assuming independent packet loss in different time intervals,

$$P_{\text{proposed}} = 1 - p^2$$

On the other hand, since a received packet can be immediately authenticated under TESLA if the packet containing its hash value is not lost,

$$P_{\text{TESLA}} = 1 - p$$

5 Denial-of-Service Attacks on the Proposed Scheme

In this section, we show that the proposed scheme can deter the Random-Substitution attack.

Let P_y be the first packet received by the receiver, $y \geq 0$. Since P_y is signed, and assuming that the signature scheme is secure, the received P_y must be authentic if it passes the signature verification.

Next, consider a receiver has an authentic packet P_{j-vd}, and has used it to immediately authenticate a packet P_j. After P_j is accepted, the receiver stores α_j, β_j. To launch the Random-Substitution attack, the adversary sends the receiver bogus packets $\tilde{P}_j = (\tilde{D}_j, r_2, K_{i-d}, i)$, where $\tilde{D}_j = \tilde{M}||r_\alpha||r_\beta$ and r_2 are values chosen by the adversary. If the receiver wanted to verify $\tilde{P}_j$, the receiver would use $\tilde{M}$, r_α , r_β to compute the hash value $\tilde{\gamma} = H(\tilde{M}, r_\alpha, r_\beta)$ and verify $\tilde{\gamma}$. Assuming the adversary cannot break the cryptographic hash function H, then $\tilde{P}_j$ will not pass the verification unless $\tilde{M} = M_j$, $r_\alpha = \alpha_j$ and $r_\beta = \beta_j$. Let us first consider the case where $\tilde{M} = M_j$, $r_\alpha = \alpha_j$ and $r_\beta = \beta_j$. If the receiver wanted to verify $\tilde{P}_j$, then the receiver would know that $\tilde{P}_j$ is valid and contains

the same M_j, α_j, β_j which were previously received from P_j, and so $\tilde{P}_j$ can be discarded. In the other case, $\tilde{M} \neq M_j$ or $r_\alpha \neq \alpha_j$ or $r_\beta \neq \beta_j$. If the receiver wanted to verify $\tilde{P}_j$, then the receiver would know that $\tilde{P}_j$ is invalid and should be discarded. Thus, in both cases, the bogus packet $\tilde{P}_j$ would have no effect on the receiver and so the receiver can immediately discard the packet $\tilde{P}_j$.

Therefore, the receiver is assured that M_j, α_j and β_j are authentic for any accepted packet P_j. So, the receiver can immediately discard any additional packet claiming to be P_j. On the other hand, in the improved TESLA scheme [4], only the message in P_j is authenticated by the receiver. In order to ensure that future packet P_{j+vd} can be authenticated upon arrival, the receiver needs to buffer all packets claiming to be P_j. The Random-Substitution attack can waste the storage and computation resources of a victim receiver, but it does not affect the proposed scheme.

6 Conclusion

We have presented modifications on the improved TESLA scheme to provide a new scheme that allows immediate data authentication even though keys are disclosed at a later time, and is resistant to the Random-Substitution attack. The proposed scheme also allows users to check data authenticity immediately upon joining the communication system. In addition, the hash value added in the proposed scheme improves the probability of immediate data authentication under lossy channel.

7 Acknowledgements

We would like to thank the anonymous referees for their useful comments on the paper.

References

1. N. Haller. The S/KEY one-time password system. Request for Comments (Informational) 1760, Internet Engineering Task Force, Feb. 1995.
2. L. Lamport. Password authentication with insecure communication. *Commun. ACM*, 24(11), Nov. 1981.
3. Adrian Perrig, Ran Canetti, J.D. Tygar, and Dawn Song. Efficient Authentication and Signing of Multicast Streams over Lossy Channels. In *Proc. of IEEE Symposium on Security and Privacy*, 2000.
4. Adrian Perrig, Ran Canetti, Dawn Song, and J.D. Tygar. Efficient and Secure Source Authentication for Multicast. In *Proc. of NDSS 2001*, 2001.
5. R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120-126,1978.
6. U. S. National Institute of Standards and Technology (NIST). Digital Signature Standard (DSS), Federal Register 56. FIPS PUB 186, Aug. 1991.

Redundant Trinomials for Finite Fields of Characteristic 2

Christophe Doche

Division of ICS, Department of Computing
Macquarie University, NSW 2109 Australia
doche@ics.mq.edu.au

Abstract. In this article we introduce *redundant trinomials* to represent elements of finite fields of characteristic 2. This paper develops applications to cryptography, especially based on elliptic and hyperelliptic curves. After recalling well-known techniques to perform efficient arithmetic in extensions of $\mathbb{F}_2$, we describe redundant trinomial bases and discuss how to implement them efficiently. They are well suited to build $\mathbb{F}_{2^n}$ when no irreducible trinomial of degree n exists. Depending on $n \in [2, 10000]$ tests with `NTL` show that, in this case, improvements for squaring and exponentiation are respectively up to 45% and 25%. More attention is given to extension degrees relevant for curve-based cryptography. For this range, a scalar multiplication can be sped up by a factor up to 15%.

1 Introduction

Although this is the first time *redundant trinomials* are used in cryptography, Brent and Zimmermann introduced the similar concept of *almost irreducible trinomials* in the context of random number generators in [2,3]. In particular, they have shown that there exist almost irreducible trinomials of degree n for every $n \in [2, 10000]$ and explained how to compute efficiently with them.

We discovered the concept of *redundant trinomial*, designed an algorithm to find them, and searched for efficient arithmetic independently. Then Brent and Zimmermann pointed out that some of this work was already contained in [2]. Additionally, this paper provides a careful analysis of the best algorithm to use to compute an inverse in a redundant trinomial basis depending on the extension degree chosen. Also, we implemented our ideas and give a precise comparison of running times between redundant trinomials and irreducible pentanomials, focused on extension degrees of cryptographic interest. Tests reveal that a certain class of redundant trinomial, called *optimal redundant trinomials*, give even better results. This leads us to introduce *optimal redundant quadrinomials* that can outclass irreducible pentanomials and even trinomials.

At present, let us recall basic facts on fields of characteristic 2. There are mainly two types of bases to compute in finite fields of characteristic 2, namely polynomial and normal bases. It is well known that there is a normal basis of $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$ for every extension degree n. However only a certain category of normal

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 122–133, 2005.

bases, namely optimal normal basis of type I or II can be used in practice. Those bases are quite rare. Considering extension fields of degree up to $10,000$, only 17.07% of them have an optimal normal basis.

For every extension degree, there is a polynomial basis as well. Sparse irreducible polynomials are commonly used to perform arithmetic in extension fields of $\mathbb{F}_2$ since they provide a fast modular reduction. As a polynomial with an even number of terms is always divisible by $x+1$, we turn our attention to so-called *trinomials*. When no such irreducible polynomial exists, one can always find an irreducible *pentanomial*, at least for extension degrees up to $10,000$. In this range this situation occurs quite often. In fact one has to choose an irreducible pentanomial in about 50% of the cases (precisely 4853 out of 9999 [11]).

The next section describes in more detail efficient algorithms to perform reduction, addition, multiplication, and inversion in $\mathbb{F}_{2^n}/\mathbb{F}_2$.

2 Finite Field Arithmetic

Let $\mu(x)$ be an irreducible polynomial of degree n over $\mathbb{F}_2$. An element of $\mathbb{F}_{2^n} \simeq \mathbb{F}_2[x]/\big(\mu(x)\big)$ is uniquely represented as a polynomial f of degree less than n with coefficients in $\mathbb{F}_2$. If f is a polynomial such that $\deg f \geqslant n$ one first reduces f modulo the irreducible polynomial μ. The usual way to get this reduction is to compute the remainder of the Euclidean division of f by μ. When μ is sparse there is a dedicated algorithm which is much faster [7].

Algorithm 1. Division by a sparse polynomial

INPUT: Two polynomials $\mu(x)$ and $f(x)$ with coefficients in a commutative ring, where $\mu(x)$ is the sparse polynomial $x^n + \sum_{i=1}^{t} a_i x^{b_i}$ with $b_i < b_{i+1}$.
OUTPUT: The polynomials u and v such that $f = u\mu + v$ with $\deg v < n$.

1. $v \leftarrow f$ and $u \leftarrow 0$
2. **while** $\deg(v) \geqslant n$ **do**
3. $\quad k \leftarrow \max(n, \deg v - n + b_t + 1)$
4. $\quad$ write $v(x) = u_1(x)x^k + w(x)$
5. $\quad v(x) \leftarrow w(x) - u_1(x)\big(\mu(x) - x^n\big)x^{k-n}$
6. $\quad u(x) \leftarrow u_1(x)x^{k-n} + u(x)$
7. **return** (u, v)

Remarks.

- *If* $\deg f = m$ *then Algorithm 1 needs at most* $2(t-1)(m-n+1)$ *field additions to compute* u *and* v *such that* $f = u\mu + v$. *In this case the number of loops is at most* $\lceil (m-n+1)/(n-b_t-1) \rceil$. *If* $m \leqslant 2n-2$, *as is the case*

when performing arithmetic modulo μ, then the number of loops is at most equal to 2 as long as $1 \leqslant b_t \leqslant n/2$.
- *To avoid computing the quotient u when it is not required, simply discard Line 6 of Algorithm 1.*

Concerning operations, additions are performed at a word level and correspond to XOR. Computing a squaring only costs a reduction modulo f. Indeed if $f(x) = \sum a_i x^i$ then $f^2(x) = \sum a_i x^{2i}$. Multiplications are also performed at a word level, but processors do not provide single precision multiplication for polynomials. Nevertheless it is possible to emulate it doing XOR and shifts. One can also store all the possible single precision products and find the global result by table lookup. This method is fast but for 32-bit words the number of precomputed values is far too big. A tradeoff consists in precomputing a smaller number of values and obtaining the final result with Karatsuba's method. Typically two 32-bit polynomials can be multiplied with 9 precomputed multiplications of 8-bit block polynomials [6].

Once the single precision multiplication is defined, different multiplication methods can be applied depending on the degree of the polynomials. In [7] the crossover between the schoolbook multiplication and Karatsuba's method is reported to be equal to 576. Other more sophisticated techniques like the F.F.T. or Cantor's multiplication [6] based on evaluation/interpolation methods can be used for larger degrees. For example, the crossover between Karatsuba's method and Cantor's multiplication is equal to 35840 in [7].

There are usually two different ways to compute the inverse of an element of $\mathbb{F}_{2^n}$. The first one is to compute an extended Euclidean GCD. The second one takes advantage of the group structure of $\mathbb{F}_{2^n}^*$ and computes α^{-1} as α^{2^n-2}.

3 Redundant Trinomials

With Algorithm 1, the product of two elements in $\mathbb{F}_{2^n}$ can be reduced with at most $4(n-1)$ elementary operations using trinomials and at most $8(n-1)$ operations using pentanomials.

For some extension degree there is an even better choice, namely *all one polynomials*. They are of the form

$$\mu(x) = x^n + x^{n-1} + \cdots + x + 1. \tag{1}$$

Such a $\mu(x)$ is irreducible if and only if $n+1$ is prime and 2 is a primitive element of $\mathbb{F}_{n+1}$. This occurs for 470 values of n up to 10,000, but n has to be even.

It is clear from the definition of $\mu(x)$ that $\mu(x)(x+1) = x^{n+1} + 1$. Thus an element of $\mathbb{F}_{2^n}$ can be represented on the *anomalous basis* $(\alpha, \alpha^2, \ldots, \alpha^n)$ where α is a root of $\mu(x)$. In other words an element of $\mathbb{F}_{2^n}$ is represented by a polynomial of degree at most n with no constant coefficient, the unity element 1 being replaced by $x + x^2 + \cdots + x^n$.

The reduction is made modulo $x^{n+1} + 1$ and a squaring is simply a permutation of the coordinates. In one sense computations in $\mathbb{F}_{2^n}$ are performed in the

ring $\mathbb{F}_2[x]/(x^{n+1}+1)$. Unfortunately this very particular and favorable choice does not apply very well to odd degrees. When n is odd, one can always embed $\mathbb{F}_{2^n}$ in a cyclotomic ring $\mathbb{F}_2[x]/(x^m+1)$. But $m \geqslant 2n+1$ so that the benefits obtained from a cheap reduction are partially obliterated by a more expensive multiplication [13]. Note that for elliptic and hyperelliptic curve cryptography only prime extension degrees are relevant [5,8,10].

We now adopt this idea and transfer it to the setting of polynomial bases. When there is no irreducible trinomial for some extension degree n one can try to find a trinomial $t(x) = x^m + x^k + 1$ with m slightly bigger than n such that $t(x)$ admits an irreducible factor $\mu(x)$ of degree n. Such a trinomial is called a *redundant trinomial*. The idea is then to embed $\mathbb{F}_{2^n} \simeq \mathbb{F}_2[x]/\big(\mu(x)\big)$ into $\mathbb{F}_2[x]/\big(t(x)\big)$. From a practical point of view an element of $\mathbb{F}_{2^n}$ is represented on the redundant basis $1, \alpha, \ldots, \alpha^{m-1}$ where α is a root of $\mu(x)$ and the computations are reduced modulo $t(x)$. As $\mu(x)$ divides $t(x)$, one can reduce modulo $\mu(x)$ at any time and obtain consistent results. If $m-n$ is sufficiently small then the multiplication of two polynomials of degree less than m has the same cost as the multiplication of two polynomials of degree less than n, since multiplications are performed at a word level.

To reduce the results one needs at most 2 iterations using Algorithm 1 since one can always choose $t(x) = x^m + x^k + 1$ such that $k \leqslant \lfloor m/2 \rfloor$. Indeed if $k > \lfloor m/2 \rfloor$ the reciprocal polynomial of $t(x)$ can be considered instead.

However with these settings, the expression of a field element is no longer unique, but the result can of course be reduced modulo $\mu(x)$, when it is required. Note that it is possible to perform a fast reduction modulo $\mu(x)$ knowing only $t(x)$ and $\delta(x) = t(x)/\mu(x)$. The same kind of idea provide a quick way to test if two polynomials represent the same field element. Finally, one examines how inversion algorithms behave with this representation.

These topics are discussed in the next section.

4 Efficient Implementation of Redundant Trinomials

To reduce a polynomial $f(x)$ modulo $\mu(x)$ one could perform the Euclidean division of $f(x)$ by $\mu(x)$, but this method has a major drawback. It obliges to determine or to know $\mu(x)$ which is not sparse in general. A better idea is to write $f(x) = q(x)\mu(x) + r(x)$. Then $f(x)\delta(x) = q(x)t(x) + r(x)\delta(x)$ so that

$$f(x) \bmod \mu(x) = \frac{f(x)\delta(x) \bmod t(x)}{\delta(x)}. \tag{2}$$

The last division is exact and can be obtained by an Algorithm easily derived from Jebelean's one for integers [9]. Now two elements $f_1(x)$ and $f_2(x)$ correspond to the same element in $\mathbb{F}_{2^n}$ if and only if $\mu(x) \mid \big(f_1(x) + f_2(x)\big)$. This implies that $t(x) \mid \delta(x)\big(f_1(x) + f_2(x)\big)$. We could compute an exact division but there is a more efficient way to proceed. First note that if $f_1(x)$ and $f_2(x)$ are both of degree at most $m-1$ then

$$\deg\big(\delta(x)\big(f_1(x) + f_2(x)\big)\big) \leqslant 2m - n - 1. \tag{3}$$

So the quotient $q(x)$ of the division of $\delta(x)\big(f_1(x)+f_2(x)\big)$ by $t(x)=x^m+x^k+1$ is of degree at most $m-n-1$. Writing the division explicitly we see that if

$$m-k>m-n-1 \tag{4}$$

then $q(x)$ is equal to the quotient of the division of $\delta(x)\big(f_1(x)+f_2(x)\big)$ by x^m. This is just a shift and it is a simple matter to determine if $\delta(x)\big(f_1(x)+f_2(x)\big)$ is equal to $q(x)(x^m+x^k+1)$ or not.

Now one can check, *cf.* [4], that all the redundant trinomials found for n up to $10,000$ satisfy $m-k>m-n-1$.

Concerning inversion, it is clear that the algorithm based on Lagrange's theorem works without any problem with redundant polynomials. One must be careful with the extended GCD algorithm. Let $\alpha\in\mathbb{F}_{2^n}$ be represented by $f(x)$. When the algorithm returns u and v such that

$$f(x)u(x)+t(x)v(x)=1 \tag{5}$$

then the inverse of α is given by $u(x)$. But one could have

$$f(x)u(x)+t(x)v(x)=d(x) \tag{6}$$

with $\deg d(x)>0$. In this case two possibilities arise. If $\mu(x)\mid d(x)$, which can be checked by looking at the degree of $d(x)$, then $\alpha=0$. Otherwise $d(x)\mid\delta(x)$ and the inverse of α is given by $u(x)e(x)$ where $e(x)$ is the inverse of $d(x)$ modulo $\mu(x)$. Nevertheless there is a more simple technique. Indeed, as we will see, $t(x)$ is squarefree. So the gcd of $f(x)\delta(x)$ and $t(x)$ is equal to $\delta(x)$ and

$$f(x)\delta(x)u_1(x)+t(x)v_1(x)=\delta(x) \tag{7}$$

so that

$$f(x)u_1(x)+\mu(x)v_1(x)=1 \tag{8}$$

and the inverse of $f(x)$ is directly given by $u_1(x)$. The degree of $\delta(x)$ is usually much smaller than the degree of $e(x)$. So the multiplication is faster. No reduction modulo $t(x)$ is required at the end. It is not necessary to compute or precompute anything new. Even when $\gcd\big(f(x),t(x)\big)=1$ this last technique works. So one can either compute the extended $\gcd\big(f(x),t(x)\big)$, test its value and compute the extended $\gcd\big(f(x)\delta(x),t(x)\big)$ if necessary, or always perform only this last computation. The tradeoff in time depends on the number of irreducible factors of δ and the cost of a modular multiplication. Indeed the degree and the number of factors of $\delta(x)$ determine the probability that a random polynomial is prime to $t(x)$. If $\delta(x)$ is irreducible of degree r then this probability is clearly equal to $1-1/2^r$. If $\delta(x)$ has two factors of degree r_1 and r_2, necessarily distinct since $t(x)$ is squarefree, the probability becomes $1-1/2^{r_1}-1/2^{r_2}+1/2^{r_1+r_2}$. By induction, if $\delta(x)$ has ℓ distinct factors of degree $r_1,r_2,\ldots,r_\ell$ then the probability that $t(x)=x^m+x^k+1$ is prime to a random polynomial of degree less than m is

$$1\quad-\sum_{\substack{n=1\\ 1\leqslant i_1<\cdots<i_n\leqslant\ell}}^{\ell}\quad\frac{(-1)^n}{2^{r_{i_1}+\cdots+r_{i_n}}}. \tag{9}$$

Note that $\delta(x)$ is irreducible in about 95% of the cases, *cf.* Section 6.

5 Example

Let us consider $\mathbb{F}_{2^8}$. There is no trinomial of degree 8 irreducible over $\mathbb{F}_2$. Instead one usually chooses the irreducible pentanomial $p(x) = x^8 + x^4 + x^3 + x + 1$. Nevertheless it is easily seen that $t(x) = x^{11} + x^5 + 1$ splits as $\mu(x)$ times $\delta(x)$ where $\mu(x) = x^8 + x^6 + x^5 + x^4 + x^2 + x + 1$ and $\delta(x) = x^3 + x + 1$ are both irreducible. The explicit expression of $\mu(x)$ is not important. In fact $t(x)$ and $\delta(x) = x^3 + x + 1$ are enough to compute in $\mathbb{F}_{2^8}$.

Let $f(x)$ and $g(x)$ be two polynomials of degree 7, namely

$$f(x) = x^7 + x^6 + x^2 + x + 1$$

and

$$g(x) = x^7 + x^6 + x^3 + x^2 + x + 1. \tag{10}$$

The product of $f(x)$ and $g(x)$ reduced modulo $t(x)$ is $h(x) = x^{10} + x^9 + x^8 + x^6 + x^5 + x^2 + x + 1$, whereas it is equal to $x^6 + x^4 + x^2 + 1$ modulo $\mu(x)$. Of course $h(x) \equiv x^6 + x^4 + x^2 + 1 \pmod{\mu(x)}$ but there is no need to reduce $h(x)$ at this stage.

Now let us compute the inverse of $f(x)$ and $g(x)$. Using an extended GCD algorithm. One obtains

$$f(x)(x^9 + x^8 + x^7 + x^4 + x^2 + x + 1) + t(x)(x^5 + x^2) = 1 \tag{11}$$

and

$$g(x)(x^4 + x^3 + x^2 + x) + t(x) = x^3 + x + 1. \tag{12}$$

We conclude immediately that the inverse of $f(x)$ is

$$f(x)^{-1} \equiv x^9 + x^8 + x^7 + x^4 + x^2 + x + 1 \pmod{t(x)}. \tag{13}$$

For the inverse of $g(x)$ one can first multiply $g(x)$ with $\delta(x)$ and compute an extended Euclidean GCD again. We get

$$g(x)\delta(x)(x^6 + x^5 + x^2 + 1) + t(x)(x^5 + x^2 + x) = x^3 + x + 1 \tag{14}$$

so that

$$g(x)^{-1} \equiv x^6 + x^5 + x^2 + 1 \pmod{t(x)}. \tag{15}$$

Using Lagrange's theorem, one gets directly

$$f(x)^{-1} \equiv f(x)^{2^8-2} \equiv x^3 + x^2 + x \pmod{t(x)} \tag{16}$$

and

$$g(x)^{-1} \equiv g(x)^{2^8-2} \equiv x^{10} + x^9 + x^5 \pmod{t(x)}. \tag{17}$$

The results are different representations of the same elements. If one wants to check it out, for example for the inverse of $f(x)$, it is enough to compute

$$(x^3 + x + 1)\big((x^9 + x^8 + x^7 + x^4 + x^2 + x + 1 + x^3 + x^2 + x) + (x^3 + x^2 + x)\big) \tag{18}$$

which is equal to $x^{12}+x^{11}+x^6+x^5+x+1$ and test if this polynomial is a multiple of $t(x)$. If so the quotient must be $x+1$ and indeed

$$(x+1)(x^{11}+x^5+1) = x^{12}+x^{11}+x^6+x^5+x+1 \tag{19}$$

so that

$$x^9+x^8+x^7+x^4+x^2+x+1+x^3+x^2+x \equiv x^3+x^2+x \pmod{\mu(x)}. \tag{20}$$

6 Search of Redundant Trinomials

An exhaustive search of redundant trinomials has been conducted using `NTL` [12] for extension degrees $n \leqslant 10,000$ when no irreducible trinomial exists. More precisely, given n we try to find a trinomial $t(x) = x^m + x^k + 1$ such that

- $t(x)$ has an irreducible factor of degree n
- m is as small as possible
- k is as small as possible.

It turns out that such a polynomial always exists for the investigated range of degree. To simplify the search one notes that such a trinomial is necessarily squarefree. Indeed $\gcd\big(t(x), t'(x)\big)$ is equal to 1 when m or k is odd. Both m and k cannot be even otherwise $x^m + x^k + 1 = (x^{m/2} + x^{k/2} + 1)^2$ and one should have chosen $x^{m/2} + x^{k/2} + 1$ instead.

Then the idea is to test all the trinomials $x^m + x^k + 1$ with $n+1 \leqslant m$ and $1 \leqslant k \leqslant \lfloor m/2 \rfloor$ until a good candidate is found, that is a trinomial with a factor of degree $m-n$.

It is well known that $x^{2^k} + x$ is equal to the product of all irreducible polynomials of degree d such that $d \mid k$. Since $t(x)$ is squarefree it is easy to determine if it has a factor $\delta(x)$ of degree $m-n$, computing $\gcd\big(x^{2^i} + x, x^m + x^k + 1\big)$ for successive $i \leqslant m-n$. Note that such a gcd computation can be very costly when $m-n$ is large. It is much faster to compute $g(x) \equiv x^{2^i} \pmod{t(x)}$ by successive squarings and reductions first and then $\gcd\big(g(x)+x, t(x)\big)$. If $t(x)$ has a factor $\delta(x)$ of degree $m-n$ the irreducibility of $t(x)/\delta(x)$ is finally checked.

For all the extensions up to the degree $10,000$ which do not have an irreducible trinomial, our proposal provides a redundant trinomial. There are 4748 such extensions. Note that when an all one polynomial is available it is given even if an irreducible trinomial exists for that extension degree.

Tables containing the redundant trinomials discovered, or all one polynomials when they exist, can be found in [4]. In this paper, we only give results for extensions of degree less than or equal to 1002, see Table 1.

The redundant trinomials $x^m + x^k + 1$ where $m = n + \deg \delta$ and the all one polynomial $(x^{n+1}+1)/(x+1)$ are respectively represented by $n, \deg \delta, k$ and $n, 1$. The degree of δ is rather small in general. In about 95% of the cases it is less than or equal to 10. It is maximum for $n = 5373$ and equals 40.

In about 87% of the cases δ is irreducible. With 32-bit processors, redundant trinomials require the same number of words as an irreducible polynomial of

2,1	4,1	8,3,5	10,1	12,1	13,3,3	16,3,4	18,1	19,3,3	24,3,4
26,3,12	27,2,1	28,1	32,5,16	36,1	37,6,4	38,2,17	40,3,3	43,10,2	45,7,9
48,3,20	50,3,5	51,2,4	52,1	53,8,28	56,2,5	58,1	59,2,26	60,1	61,5,17
64,7,12	66,1	67,9,29	69,3,13	70,7,11	72,3,8	75,2,4	77,3,9	78,2,31	80,3,11
82,1	83,2,14	85,8,28	88,8,19	91,8,1	96,2,1	99,2,13	100,1	101,2,2	104,5,9
106,1	107,2,8	109,9,21	112,3,22	114,3,4	115,10,6	116,5,17	117,6,31	120,2,25	122,3,9
125,3,3	128,2,17	130,1	131,7,61	133,3,43	136,3,30	138,1	139,3,3	141,3,13	143,3,53
144,7,19	148,1	149,2,2	152,2,65	157,7,25	158,5,19	160,3,27	162,1	163,8,70	164,5,59
165,5,9	168,3,1	171,2,10	172,1	173,3,5	176,2,53	178,1	179,2,14	180,1	181,7,51
184,3,60	187,7,45	188,4,61	189,3,37	190,4,33	192,3,53	195,2,25	196,1	197,3,69	200,5,42
203,7,73	205,5,29	206,2,17	208,3,45	210,1	211,3,103	213,3,37	216,3,101	219,2,1	221,15,77
222,2,37	224,3,86	226,1	227,2,77	229,3,61	230,2,35	232,3,69	235,14,7	237,3,41	240,2,37
243,2,52	245,2,2	246,2,109	248,2,41	251,2,74	254,2,71	256,16,45	259,5,103	261,3,109	262,4,89
264,3,68	267,2,88	268,1	269,5,47	272,3,87	275,3,99	277,6,12	280,5,103	283,3,51	285,6,122
288,2,133	290,3,114	291,2,31	292,1	293,2,2	296,5,15	298,7,91	299,2,5	301,6,78	304,3,46
306,8,55	307,5,119	309,7,87	311,8,139	312,9,143	315,2,127	316,1	317,3,113	320,3,26	323,2,41
325,3,151	326,2,5	328,3,52	331,13,69	334,5,115	335,6,20	336,3,1	338,3,32	339,2,13	341,10,124
344,2,125	346,1	347,2,173	348,1	349,6,177	352,3,78	355,11,173	356,15,38	357,3,79	360,3,53
361,8,64	363,2,169	365,3,89	368,8,55	371,2,56	372,1	373,3,85	374,2,5	376,3,159	378,1
379,3,187	381,8,99	384,3,94	387,2,67	388,1	389,5,193	392,3,71	395,11,187	397,5,13	398,9,203
400,3,159	403,5,127	405,3,13	408,9,90	410,4,107	411,5,105	413,3,9	416,6,15	418,1	419,2,176
420,1	421,8,14	424,9,112	427,5,5	429,3,137	430,8,91	432,3,38	434,3,170	435,2,61	437,6,12
440,3,146	442,1	443,10,68	445,5,193	448,3,78	451,7,139	452,7,211	453,3,227	454,5,10	456,2,25
459,2,202	460,1	461,3,27	464,2,101	466,1	467,2,29	469,8,109	472,3,214	475,5,133	477,3,89
480,7,224	482,3,108	483,2,16	485,7,181	488,3,180	490,1	491,2,224	493,3,37	496,3,66	499,16,137
501,10,101	502,5,70	504,5,167	507,2,49	508,1	509,12,204	512,9,252	515,2,5	517,5,65	520,3,18
522,1	523,3,3	525,10,89	528,3,121	530,3,24	531,2,226	533,3,195	535,7,25	536,2,113	539,2,92
540,1	541,6,37	542,2,209	544,5,215	546,1	547,7,131	548,2,107	549,5,261	552,2,133	554,3,27
555,2,58	556,1	557,8,12	560,3,99	562,1	563,2,86	565,3,3	568,3,40	571,5,187	572,5,281
573,5,249	576,2,169	578,9,153	579,2,148	581,11,241	584,3,72	586,1	587,2,104	589,3,97	591,8,67
592,7,37	595,3,135	597,7,257	598,5,13	600,2,145	603,2,4	605,3,219	608,3,48	611,2,11	612,1
613,10,76	616,3,64	618,1	619,5,265	621,3,283	624,2,193	627,5,261	629,3,269	630,2,37	632,2,281
635,2,290	637,3,127	638,2,89	640,7,23	643,5,191	644,2,287	645,3,103	648,5,26	652,1	653,3,155
656,2,125	658,1	659,2,80	660,1	661,3,81	664,3,297	666,3,173	667,3,211	669,5,139	672,3,8
674,3,186	675,8,219	676,1	677,3,59	678,2,169	680,2,269	681,2,193	683,2,47	685,3,255	688,3,204
691,14,298	693,8,258	696,3,95	699,2,160	700,1	701,3,167	703,3,25	704,5,169	706,3,34	707,2,74
708,1	709,3,123	710,2,251	712,3,136	715,7,165	717,6,110	720,3,251	723,3,295	725,8,168	728,2,53
731,2,146	733,3,45	734,4,329	736,3,174	739,7,27	741,7,83	744,2,49	747,2,241	749,8,205	752,2,353
755,2,98	756,1	757,3,97	760,5,46	763,3,247	764,4,299	765,3,127	766,5,130	768,3,19	770,3,44
771,2,103	772,1	773,3,11	776,3,132	779,2,161	781,6,375	784,9,86	786,1	787,3,67	788,9,266
789,10,276	790,5,136	792,2,325	795,2,169	796,1	797,7,347	800,2,77	802,7,341	803,2,89	805,3,219
808,6,403	811,5,161	813,3,181	816,5,288	819,2,313	820,1	821,6,363	824,2,149	826,1	827,2,68
828,1	829,3,291	830,2,323	832,3,94	835,5,101	836,2,275	837,5,223	840,3,155	843,2,187	848,2,341
851,2,119	852,1	853,3,307	854,2,161	856,3,235	858,1	859,9,197	863,6,300	864,5,144	867,2,25
869,3,75	872,9,27	874,6,111	875,2,392	876,1	877,10,69	878,7,341	880,3,61	882,1	883,5,395
885,3,137	886,9,314	888,3,241	891,2,442	893,5,59	896,3,65	899,2,329	901,6,1	904,3,6	906,1
907,7,105	909,3,55	910,7,131	912,2,337	914,9,369	915,2,349	917,3,9	920,3,375	922,3,229	923,2,389
925,12,18	928,7,64	929,2,302	931,10,415	933,3,1	934,5,428	936,2,25	939,2,67	940,1	941,3,317
944,2,125	946,1	947,2,50	949,8,38	950,2,383	952,3,324	955,7,321	957,3,367	958,7,174	960,2,241
962,3,464	963,2,7	965,3,293	968,2,29	970,7,226	971,2,179	973,12,233	974,7,65	976,3,394	978,3,425
980,5,11	981,5,235	984,2,313	987,2,28	989,3,11	992,5,472	995,2,89	997,3,319	1000,9,140	1002,3,41

Table 1.

degree n in more than 86% of the cases to represent field elements. Otherwise one more word is necessary, except for the extension of degree 5373 which needs two more words.

For each degree, the factor δ is not explicitly given in Table 1, but it is easy to retrieve since

$$\delta(x) = \gcd\left(x^m + x^k + 1, \prod_{i=1}^{m-n}\left(x^{2^i} + x\right)\right). \tag{21}$$

Also $\delta(x)$ can be found by trial divisions when its degree is small.

The complete data, including the expression of $\delta(x)$, are available on the Internet [4].

7 Tests

Computations have been done on a PC with a Pentium IV processor at 2.6 Ghz running Linux. The test program was written in `C++`, compiled with `gcc-2.96` using `NTL 5.3.1` [12] and compares the efficiency of irreducible pentanomials against redundant trinomials for some basic operations within extension fields of $\mathbb{F}_2$ of prime degree between 50 and 400. For both systems of representation, namely $\mathbb{F}_2[x]/\big(p(x)\big)$ and $\mathbb{F}_2[x]/\big(t(x)\big)$, we give in Table 2 the running times and the respective speedup (in percent) for

- the reduction of a polynomial of degree $2n-2$ (resp. $2m-2$) modulo $p(x)$ (resp. $t(x)$).
- the squaring of an element of $\mathbb{F}_{2^n}$
- the multiplication of two elements of $\mathbb{F}_{2^n}$
- the exponentiation of an element of $\mathbb{F}_{2^n}$ to an exponent less than 2^n.

The unit used is 10^{-7}s for reduction, squaring and multiplication. It is 10^{-5}s for exponentiation.

Redundant trinomials are not well suited for inversions, at least when computed with an extended GCD computation. Results show that inversions are about 15% slower with redundant trinomials.

We remark that prime extension degrees 59, 197, 211, 277, 311, 317, 331, 347, 389, and 397 are quite particular. Indeed for these n, there exists a trinomial of degree $m = \lceil n/32 \rceil \times 32$ with an irreducible factor of degree n. We call such a polynomial an *optimal redundant trinomial*. For all these degrees, except for $n = 317$, another redundant trinomial of smaller degree exists. However tests show that the results are much better with optimal trinomials. Thus when it is possible, these polynomials are used instead. With the same conventions as previously they are

$59, 5, 9$	$197, 27, 103$	$211, 13, 67$	$277, 11, 83$	$293, 27, 91$
$311, 9, 33$	$331, 21, 81$	$347, 5, 127$	$389, 27, 205$	$397, 19, 175$

		Red.			Sqr.			Mul.			Exp.		
n	$\deg \delta$	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*
53	8	1.63	1.37	15.95	2.17	1.77	18.43	3.51	3.04	13.39	1.82	1.53	15.93
59	5	1.64	0.89	45.73	2.17	1.37	36.87	3.51	2.63	25.07	2.01	1.39	30.85
61	5	1.63	1.33	18.40	2.20	1.70	22.73	3.49	4.87	−39.54	2.07	2.05	0.97
67	9	1.57	1.31	16.56	2.18	1.80	17.43	5.37	4.99	7.08	2.67	2.28	14.61
83	2	2.01	1.48	26.37	2.46	1.89	23.17	5.70	5.40	5.26	3.51	3.00	14.53
101	2	1.88	1.50	20.21	2.42	2.01	16.94	6.60	6.08	7.88	4.44	3.88	12.61
107	2	1.91	1.50	21.47	2.49	2.02	18.88	6.53	6.02	7.81	4.64	4.05	12.72
109	9	1.93	1.64	15.03	2.47	2.16	12.55	6.53	6.26	4.13	4.76	4.33	9.03
131	7	2.04	1.50	26.47	2.62	2.14	18.32	10.28	10.07	2.04	7.26	6.28	13.50
139	3	2.37	1.87	21.10	3.00	2.16	28.00	10.77	10.24	4.92	8.19	6.95	15.14
149	2	2.69	1.86	30.86	3.24	2.39	26.23	11.02	10.55	4.26	9.15	7.68	16.07
157	7	2.73	1.82	33.33	3.31	2.39	27.79	11.01	12.46	−13.17	9.73	8.92	8.32
163	8	2.50	1.72	31.20	3.02	2.28	24.50	13.31	12.08	9.24	10.65	8.90	16.43
173	3	2.73	1.90	30.40	3.38	2.47	26.92	13.00	12.39	4.69	11.61	9.87	14.99
179	2	3.01	2.15	28.57	3.61	2.67	26.04	13.09	12.66	3.28	12.90	10.66	17.36
197	27	3.03	1.51	50.17	3.78	2.14	43.39	15.16	13.50	10.95	14.50	10.74	25.93
211	13	3.43	1.55	54.81	4.14	2.14	48.31	15.35	13.50	12.05	16.49	11.50	30.26
227	2	3.17	2.27	28.39	4.01	2.98	25.69	17.08	15.51	9.19	18.29	15.53	15.09
229	3	3.25	2.35	27.69	4.18	3.03	27.51	16.70	15.28	8.50	18.24	15.75	13.65
251	2	3.70	2.52	31.89	4.72	3.07	34.96	16.79	15.27	9.05	21.14	17.70	16.27
269	5	3.71	3.04	18.06	4.62	3.77	18.40	27.05	26.49	2.07	28.65	26.51	7.47
277	11	4.12	1.97	52.18	4.80	2.70	43.75	27.43	25.37	7.51	30.44	23.42	23.06
283	3	4.08	3.16	22.55	4.86	3.89	19.96	27.43	26.47	3.50	31.22	28.30	9.35
293	27	3.81	2.12	44.36	4.69	2.88	38.59	31.09	29.12	6.34	34.15	28.03	17.92
307	5	4.50	2.96	34.22	5.32	3.67	31.02	31.70	30.11	5.02	38.10	32.48	14.75
311	9	4.52	2.09	53.76	5.33	2.90	45.59	31.74	29.11	8.29	38.58	29.63	23.20
317	3	4.52	2.12	53.10	5.36	2.87	46.46	31.74	29.12	8.25	39.18	30.01	23.40
331	21	4.57	2.26	50.55	5.58	3.18	43.01	35.95	33.54	6.70	44.07	35.56	19.31
347	5	4.98	2.20	55.82	5.83	3.12	46.48	36.18	33.53	7.32	47.41	37.04	21.87
349	6	4.99	3.16	36.67	5.83	4.06	30.36	36.17	37.58	−3.90	47.77	43.24	9.48
373	3	5.18	3.51	32.24	6.23	4.33	30.50	38.44	36.55	4.92	53.66	45.72	14.80
379	3	5.20	3.34	35.77	6.25	4.26	31.84	38.44	36.67	4.60	54.44	46.21	15.12
389	5	4.50	3.29	26.89	5.50	4.15	24.55	41.67	40.44	2.95	56.47	50.41	10.73
389	27	4.56	2.41	47.15	5.52	3.35	39.31	41.67	39.40	5.45	56.13	46.48	17.19
397	19	5.24	2.39	54.39	6.20	3.36	45.81	42.14	39.41	6.48	60.50	47.41	21.64

Table 2.

		Red.			Sqr.			Mul.			Exp.		
n	$\deg \delta$	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*
1019	2	1.22	0.75	38.52	1.41	0.96	31.91	1.36	1.32	2.94	39.84	33.97	14.73
2499	2	2.57	1.80	29.96	2.94	2.05	30.27	7.60	7.50	1.32	365.91	340.75	6.88
5013	9	4.68	3.31	29.27	5.45	4.00	26.61	22.68	22.54	0.62	1840.55	1757.94	4.49
7597	17	7.87	5.05	35.83	8.65	5.97	30.98	35.34	35.09	0.71	4133.90	3896.40	5.75
9995	2	9.92	6.59	33.57	11.22	7.78	30.66	67.96	67.62	0.50	9561.80	9180.50	3.99

Table 3.

		Dbl.			Add.			Mul.		
n	$\deg \delta$	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*
163	8	1.35	1.24	8.15	3.60	3.33	7.50	1.79	1.61	10.06
197	27	1.52	1.09	28.29	4.07	3.49	14.25	2.42	2.10	13.22
277	6	1.81	1.57	13.26	6.72	6.45	4.02	5.69	5.41	4.92
317	3	1.91	1.30	31.94	7.61	6.74	11.43	7.41	6.65	10.26

Table 4.

Unfortunately the extension degrees which allow the use of optimal redundant trinomials are quite rare. However an *optimal redundant quadrinomial* whose degree is a multiple of 32 and having an irreducible factor of degree n are much easier to find for a given n. Tests with `NTL` showed that in some cases optimal redundant quadrinomials give better result than nonoptimal redundant trinomials and even than irreducible trinomials.

In Table 3 we perform the same computation for bigger degrees. The units are in μs for reduction and squaring, 10^{-5}s for multiplication and 10^{-4}s for exponentiation.

Finally, we have done some computations on elliptic curves defined over finite fields represented with pentanomials and redundant trinomials. Table 4 contains the running times of an addition and a doubling in μs with Montgomery's method. The times for scalar multiplications, also with Montgomery's method, are in ms.

8 Conclusion

In this paper we propose to use reducible trinomials, called redundant trinomial, instead of irreducible pentanomials to represent finite fields of characteristic 2. This allows a faster reduction and more generally a faster arithmetic. The improvement is about 20% for reductions and squarings. For multiplications it is usually less than 5%. We also propose to use sparse reducible polynomials of degree a multiple of the word length (usually 32 bits) having an irreducible factor of degree n to represent $\mathbb{F}_{2^n}$. This idea seems promising but has to be investigated further. Testing the equality of two elements is a costly operation, and should be avoided if possible.

This work naturally extends to other fields, in particular extension fields of characteristic 3. It can be applied to larger characteristic as well. Indeed Mersenne prime numbers or primes of the form $2^n \pm c$ with c small are used to define prime fields of large characteristic and Optimal Extension Fields [1] because of the fast integer reduction they provide. However these primes are quite rare, but when $N = 2^n \pm c$ is not prime but has a large prime factor p the same kind of idea applies, namely working in $\mathbb{F}_p$ by actually computing in $\mathbb{Z}/N\mathbb{Z}$.

Acknowledgment

The author would like to thank Richard Brent and Paul Zimmermann who made him aware of their work [2,3].

References

1. D. V. Bailey and C. Paar. Efficient arithmetic in finite field extensions with application in elliptic curve cryptography. *Journal of Cryptology*, 14(3):153–176, 2001.

2. R. Brent and P. Zimmermann. Algorithms for finding almost irreducible and almost primitive trinomials. Primes and Misdemeanours: Lectures in Honour of the Sixtieth Birthday of Hugh Cowie Williams, The Fields Institute, Toronto, to be published by the American Mathematical Society.
See `http://web.comlab.ox.ac.uk/oucl/work/richard.brent/pd/rpb212.pdf`.
3. R. Brent and P. Zimmermann. Random number generators with period divisible by a mersenne prime. In *Computational Science and its Applications - ICCSA 2003*, volume 2667, pages 1–10. Springer-Verlag, Berlin, 2003.
See `http://web.comlab.ox.ac.uk/oucl/work/richard.brent/pd/rpb211.pdf`.
4. C. Doche. A table of redundant trinomials in characteristic 2 up to the degree 10000.
See `http://www.math.u-bordeaux.fr/~cdoche/documents/redundant.gp.gz`.
5. G. Frey. Applications of arithmetical geometry to cryptographic constructions. In D. Jungnickel and H. Niederreiter, editors, *Fifth International Conference on Finite Fields and Applications*, pages 128–161. Springer-Verlag, Berlin, 2001.
6. J. von zur Gathen and J. Gerhard. Arithmetic and factorization of polynomials over $\mathbb{F}_2$, 1996.
7. J. von zur Gathen and M. Nöcker. Polynomial and normal bases for finite fields. To appear.
8. P. Gaudry, F. Hess, and N. P. Smart. Constructive and destructive facets of Weil descent on elliptic curves. *Journal of Cryptology*, 15(1):19–46, 2002. Online publication: 29 August 2001.
9. T. Jebelean. An algorithm for exact division. *J. Symbolic Computation*, 15(2):169–180, 1993.
10. A. Menezes and M. Qu. Analysis of the Weil descent attack of Gaudry, Hess and Smart. In *Topics in Cryptology – CT-RSA 2001*, volume 2020 of *Lecture Notes in Comput. Sci.*, pages 308–318. Springer-Verlag, Berlin, 2001.
11. G. Seroussi. Table of low–weight binary irreducible polynomials. Technical Report HPL–98–135, Hewlett–Packard, August 1998.
12. V. Shoup. NTL: A Library for doing Number Theory, ver. 5.3.1.
13. H. Wu, M. A. Hasan, I. F. Blake, and S. Gao. Finite field multiplier using redundant representation. *IEEE Trans. Computers*, 51(11):1306–1316, 2002.

Efficient Tate Pairing Computation for Elliptic Curves over Binary Fields

Soonhak Kwon

Inst. of Basic Science and Dept. of Mathematics, Sungkyunkwan University, Suwon 440-746, Korea
shkwon@skku.edu

Abstract. In this paper, we present a closed formula for the Tate pairing computation for supersingular elliptic curves defined over the binary field $\mathbb{F}_{2^m}$ of odd dimension. There are exactly three isomorphism classes of supersingular elliptic curves over $\mathbb{F}_{2^m}$ for odd m and our result is applicable to all these curves.

Keywords: supersingular elliptic curve, Tate pairing, divisor, automorphism, roots of unity.

1 Introduction

Many cryptographic schemes are based on the bilinear pairings arising from the rank two abelian group structure of the points of prescribed order of the given elliptic curve. Bilinear pairings were originally used as tools for attacking discrete logarithm problem for supersingular elliptic curves by Menezes et al. [1] and also by Frey and Rück [2], and they become popular these days for efficient encryption and signature schemes. Examples of such cryptographic protocols are, to name just a few, identity based encryption scheme by Boneh and Franklin [3], short signature scheme by Boneh et al. [4], tripartite Diffie-Hellman key agreement protocol by Joux [5], identity based authenticated key agreement protocol by Smart [7], and identity based signature schemes by Sakai et al. [6], Hess [16], Cha and Cheon [19], Baek and Zheng [28]. In most of these applications, the Tate pairing of supersingular elliptic curves (or curves of small embedding degrees) is an essential tool. Therefore efficient computation of the Tate pairing is a crucial factor for practical applications of the above mentioned cryptographic protocols.

Recently many progresses have been made on the computation of the Tate pairing. A few refined techniques and ideas to speed up the computation of the Tate pairing are suggested in [8,9,10,14,21,24]. The notion of the squared Tate pairing is introduced by Eisenträger [11]. Barreto et al. [14] showed that the algorithm of Miller [22] can be modified to a new algorithm where division in a finite field can be omitted since the denominator becomes one after final powering. Also Duursma and Lee [10] presented a closed formula for the computation of the Tate pairing for a finite field with characteristic *three*, which significantly reduces the cost of computation.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 134–145, 2005.

In this paper, we show that an efficient closed formula can also be obtained for the computation of the Tate pairing for supersingular elliptic curves over a binary field $\mathbb{F}_{2^m}$ with odd dimension m. There are exactly three isomorphism classes of supersingular elliptic curves over $\mathbb{F}_{2^m}$ with m odd [17] and our method is applicable to all these curves. Also we present a method of avoiding inverse Frobenius operations in our and Duursma-Lee's algorithms. When one wants to use a polynomial basis, inverse Frobenius operation is not at all trivial unlike the case of a normal basis. We propose new modified algorithms which avoid the inverse Frobenius map without affecting the computational merits of the original algorithms.

A preliminary version of this work was posted through e-print archive, http://eprint.iacr.org/2004/303.pdf. Subsequently, the author was informed that a similar work was already presented by Barreto, Galbraith, O hEigeartaigh and Scott in ECC 2004 (slides are available through http://www.cacr.math.uwaterloo.ca/conferences/2004/ecc2004/barreto.pdf). Their preprint containing generalization to hyperelliptic case has appeared through http://eprint.iacr.org/2004/375.pdf.

2 Elliptic Curves and Miller's Algorithm

Let E be an elliptic curve over a finite field $\mathbb{F}_q$ where q is a power of a prime. We may express E as the standard Weierstrass form, $E : Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6$, where the coefficients a_1, a_2, a_3, a_4, a_6 are in $\mathbb{F}_q$. Let $E(\mathbb{F}_q)$ be the additive group of all points $P = (x, y)$, $x, y \in \mathbb{F}_q$, on the curve with the point at infinity O. Let l be a positive integer and let $E[l]$ (resp. $E[l](\mathbb{F}_q)$) be the set of points $P \in E(\overline{\mathbb{F}}_q)$ (resp. $P \in E(\mathbb{F}_q)$) satisfying $lP = O$, where $\overline{\mathbb{F}}_q$ is an algebraic closure of $\mathbb{F}_q$. Let k be the minimal degree of the extension satisfying $E[l] \subset E(\mathbb{F}_{q^k})$. Such k is called the embedding degree (or the security multiplier) of $E[l]$ [17,25] and is dependent on E and l. If l is prime to q, then it is well known [13] that $E[l] \cong \mathbb{Z}/l \oplus \mathbb{Z}/l$.

A divisor D on E is a formal (finite) sum of the points P on the curve $D = \sum n_p(P)$, $n_p \in \mathbb{Z}$. We call D a degree 0 divisor if $\sum n_p = 0$. A principal divisor is a divisor of the form $(f) = \sum n_p(P)$, where f is a rational function on E and P is a point of E with n_P the order of multiplicity of f at P, i.e. $n_P > 0$ if f has a zero at P and $n_P < 0$ if f has a pole at P. We say two divisors D and D' are equivalent if $D - D'$ is a principal divisor. It is well known [13,17] that a principal divisor (f) is a degree 0 divisor, and a divisor $D = \sum n_p(P)$ is a principal divisor if D is a degree 0 divisor and $\sum n_p P = O$ in the abelian group $E(\overline{\mathbb{F}}_q)$. More precisely, there is an isomorphism

$$Div_0/Div_{prin} \longrightarrow E, \quad \text{with} \quad D = \sum n_p(P) \longmapsto \sum n_p P, \tag{1}$$

where the summation in the right side is the addition of points on the elliptic curve E and Div_0 (resp. Div_{prin}) is a free abelian group generated by the degree 0 divisors (resp. principal divisors). Now suppose that $P \in E[l]$. Then the divisor $l(P) - l(O)$ is a principal divisor so that there is a rational function f_P such that $(f_P) = l(P) - l(O)$. For any rational function f and any divisor $D = \sum n_p(P)$

having disjoint supports, one naturally defines $f(D) = \prod f(P)^{n_P}$. The Tate pairing τ_l on the set $E[l]$ is defined as follows.

Definition 1. *Let $P \in E[l](\mathbb{F}_q)$ and $Q \in E[l](\mathbb{F}_{q^k})$. The Tate pairing is a map*

$$\tau_l : E[l](\mathbb{F}_q) \times E[l](\mathbb{F}_{q^k}) \longrightarrow \{\zeta_l\}, \quad \text{with} \quad \tau_l(P,Q) = f_P(D_Q)^{\frac{q^k-1}{l}},$$

where f_P is a rational function satisfying $(f_P) = l(P) - l(O)$ and D_Q is a degree 0 divisor equivalent to $(Q) - (O)$ such that D_Q and (f_P) have disjoint supports. Also $\{\zeta_l\}$ is the group of l-th roots of unity in $\mathbb{F}_{q^k}^{\times}$.

It is well known that τ_l is a non-degenerate bilinear pairing and a proof can be found in [2,15]. It is also easy to verify $\tau_{ld}(P,Q) = \tau_l(P,Q)$ for $P, Q \in E[l]$ and $d > 0$ with ld dividing $|E(\mathbb{F}_q)|$.

An effective algorithm for finding a rational function f_P satisfying $(f_P) = l(P) - l(O)$ with $P \in E[l]$ is found by Miller [17,22]. Let us briefly explain the idea of Miller. For any degree 0 divisor D and D', the isomorphism in (1) implies that there exist points P and P' such that $D = (P) - (O) + (f)$ and $D' = (P') - (O) + (f')$ for some rational functions f and f'. Then one has the following formula due to Miller,

$$D + D' = (P + P') - (O) + (ff'\frac{\ell_{P,P'}}{\ell_{P+P'}}), \tag{2}$$

where $\ell_{P,P'}$ is an equation of a line intersecting P and P', and ℓ_P is an equation of a vertical line intersecting P and $-P$. This can be verified using the relation $(\frac{\ell_{P,P'}}{\ell_{P+P'}}) = (\ell_{P,P'}) - (\ell_{P+P'}) = (P) + (P') + (-P - P') - 3(O) - \{(P + P') + (-P - P') - 2(O)\} = (P) + (P') - (P + P') - (O)$.

An elliptic curve E over $\mathbb{F}_q$ is called supersingular if $Tr(\varphi) \equiv 0 \pmod p$ where φ is the Frobenius map and p is the characteristic of $\mathbb{F}_q$. If an elliptic curve E over $\mathbb{F}_q$ is supersingular, then it is well known [17] that for any l dividing $|E(\mathbb{F}_q)|$, the embedding degree k is bounded by 6. More precisely, we have $E[l] \subset E(\mathbb{F}_{q^k})$ with $k = 2, 3, 4, 6$. It is also well known that the embedding degree $k = 6$ is attained when the characteristic of $\mathbb{F}_q$ is *three* and the embedding degree $k = 4$ is attained when the characteristic of $\mathbb{F}_q$ is *two*. It should be mentioned that non-supersingular curves of low embedding degrees (≤ 6) are found by Miyaji et al. [12], which have some potential security advantage over supersingular curves.

3 Review of Previous Works

For some families of supersingular curves with embedding degree $k = 2, 4, 6$, Barreto et al. [14] showed that one can speed up the computation of the Tate pairing by observing that the denominators ℓ_Q appearing in the Miller's algorithm can be omitted using the idea of the distortion map ϕ introduced by Verheul [25], where ϕ is a suitably chosen nontrivial automorphism of the given supersingular elliptic curve. That is, since the line $X - \alpha$ intersecting $Q = (\alpha, \beta) \in \mathbb{F}_q$ and $-Q$

has only X-coordinate and since this X-coordinate has the value in $\mathbb{F}_{q^{k/2}}$ after applying ϕ to Q, it becomes one after taking the final power by $\frac{q^k-1}{l}$ because $l|q^{k/2}+1$ and $q^k-1=(q^{k/2}-1)(q^{k/2}+1)$. By the similar reasoning, they also showed that it is not necessary to evaluate the Tate pairing at the point at infinity O. To summarize, one may twist the pairing in Definition 1 such as $\tau_l(P,Q)=f_P(\phi(Q))^{\frac{q^k-1}{l}}$, which simplifies all the necessary computations.

For a field with characteristic *three*, $\mathbb{F}_q$ with $q=3^m$, Duursma and Lee [10] noticed that one can obtain a faster Tate pairing computation if one uses $l=q^3+1=3^{3m}+1$, since the ternary expansion of q^3+1 is trivial. That is, if one write g_Q as a rational function satisfying $3(Q)-3(O)=(3Q)-(O)+(g_Q)$, then, by repeated applications of the above equation, one has

$$3^{3m}(P)-3^{3m}(O)=(3^{3m}P)-(O)+(g_P^{3^{3m-1}}g_{3P}^{3^{3m-2}}\cdots g_{3^{3m-2}P}^{3}g_{3^{3m-1}P}).$$

It is shown [10] that the rational function $f=\prod_{i=1}^{3m}g_{3^{i-1}P}^{3^{3m-i}}$ can be used for a computation of the Tate pairing as $\tau_l(P,Q)=f(\phi(Q))^{3^{3m}-1}$. Duursma and Lee [10] showed that the value $f(\phi(Q))=\prod_{i=1}^{3m}\{g_{3^{i-1}P}(\phi(Q))\}^{3^{3m-i}}$ has certain cyclic property with regard to the polynomials $g_{3^{i-1}P}^{3^{3m-i}}$ so that they found a nice closed formula for f as a product of m (not $3m$) polynomials.

4 Tate Pairing Computation for Binary Fields

4.1 Supersingular Elliptic Curves over Binary Fields

For cryptographic purposes, it is natural to think of elliptic curves defined over $\mathbb{F}_{2^m}$ with m odd or more strongly a prime. There are exactly three isomorphism classes of supersingular elliptic curves over $\mathbb{F}_{2^m}$ when m is odd [17]. Namely they are $Y^2+Y=X^3+X$, $Y^2+Y=X^3+X+1$ and $Y^2+Y=X^3$. Among them, the curves

$$E_b: Y^2+Y=X^3+X+b, \quad b=0,1 \tag{3}$$

have the embedding degree (or security multiplier) $k=4$ while the curve $Y^2+Y=X^3$ has $k=2$. Thus we are mainly interested in the curves E_b though our method is also applicable to the curve $Y^2+Y=X^3$. The Frobenius map $\varphi: E_b \longrightarrow E_b$ with $\varphi(x,y)=(x^2,y^2)$ is a root of the characteristic polynomial $h(X)=X^2\pm 2X+2=(X-\varphi)(X-\bar{\varphi})$. We also have the order $|E_b(\mathbb{F}_{2^m})|$ of the group of rational points $E_b(\mathbb{F}_{2^m})$ as $|E_b(\mathbb{F}_{2^m})|=2^m+1-Tr(\varphi^m)$, where $Tr(\varphi^m)=\varphi^m+\bar{\varphi}^m$ and $\varphi^m(x,y)=(x^{2^m},y^{2^m})$. Letting $c_j=Tr(\varphi^j)$, one can find the values of c_j using the second order linear recurrence relations (or Lucas type sequences) arising from the characteristic polynomial $h(X)$, $c_j=2(\mp c_{j-1}-c_{j-2})$, $j\geq 0$, with $c_0=2$ and $c_1=\mp 2$. From these relations, it is straightforward to see [17] that $E_b(\mathbb{F}_{2^m})$ is a cyclic group of order

$$\begin{aligned} |E_b(\mathbb{F}_{2^m})| &= 2^m+1+(-1)^b\sqrt{2\cdot 2^m}, \quad \text{if} \quad m\equiv 1,7 \pmod 8 \\ &= 2^m+1-(-1)^b\sqrt{2\cdot 2^m}, \quad \text{if} \quad m\equiv 3,5 \pmod 8. \end{aligned} \tag{4}$$

4.2 Closed Formula of the Tate Pairing for $Y^2 + Y = X^3 + X + b$

As in the characteristic three case of Duursma and Lee [10], we want to derive a closed formula for the Tate pairing computation using the simple equality for our binary case, $2^{2m} + 1 = (2^m + 1 + 2^{\frac{m+1}{2}})(2^m + 1 - 2^{\frac{m+1}{2}})$. Let $P = (\alpha, \beta)$ be a point on the curve $E_b : Y^2 + Y = X^3 + X + b$, $b = 0, 1$. Then one has $-P = (\alpha, \beta + 1)$ and $2P = (\alpha^4 + 1, \alpha^4 + \beta^4)$. Thus we get $2^2P = (\alpha^{2^4}, \beta^{2^4} + 1) = -\varphi^4(P)$, $2^3P = (\alpha^{2^6} + 1, \alpha^{2^6} + \beta^{2^6} + 1)$, $2^4P = (\alpha^{2^8}, \beta^{2^8})$, where $\varphi^4 + 4 = 0$, i.e. $h(X) = X^2 \pm 2X + 2$ divides $X^4 + 4$. Using this cyclic property, one finds easily

$$\begin{aligned} 2^{i-1}P &= (\alpha^{2^{2i-2}} + i - 1, \beta^{2^{2i-2}} + (i-1)\alpha^{2^{2i-2}} + \epsilon_i) \\ &= (\alpha^{(2i-2)} + i - 1, \beta^{(2i-2)} + (i-1)\alpha^{(2i-2)} + \epsilon_i), \end{aligned} \tag{5}$$

where $\alpha^{(j)}$ (resp. $\beta^{(j)}$) is defined as $\alpha^{(j)} = \alpha^{2^j}$ (resp. $\beta^{(j)} = \beta^{2^j}$) and ϵ_i is defined as

$$\epsilon_i = 0 \quad \text{if } i \equiv 1, 2 \pmod 4 \quad \text{and} \quad \epsilon_i = 1 \quad \text{if } i \equiv 3, 4 \pmod 4. \tag{6}$$

For an effective Tate pairing computation, the following distortion map (non-trivial automorphism) $\phi : E_b \longrightarrow E_b$ with $\phi(x, y) = (x + s^2, y + sx + t)$ is chosen [14], where $s^2 + s + 1 = 0$ and $t^2 + t + s = 0$. That is, $\mathbb{F}_2(s) = \mathbb{F}_{2^2}$, $\mathbb{F}_2(t) = \mathbb{F}_{2^4}$, $s = t^5$, $t^4 + t + 1 = 0$, and t is a generator of the group $\mathbb{F}_{2^4}^\times$ of order 15.

For any point Q on the curve E_b, let us write g_Q as a rational function satisfying $2(Q) - 2(O) = (2Q) - (O) + (g_Q)$. By the Miller's formula in (2), we have $g_Q = \ell_{Q,Q}/\ell_{2Q}$ and the denominator ℓ_{2Q} can be omitted by the result in [14]. Now for a given point $P \in E_b(\mathbb{F}_{2^m})$, one repeatedly has

$$\begin{aligned} 2(P) - 2(O) &= (2P) - (O) + (g_P), \\ 2^2(P) - 2^2(O) &= 2\{(2P) - (O)\} + (g_P^2) = (2^2P) - (O) + (g_P^2 g_{2P}), \\ &\cdots \\ 2^{2m}(P) - 2^{2m}(O) &= (2^{2m}P) - (O) + (g_P^{2^{2m-1}} g_{2P}^{2^{2m-2}} \cdots g_{2^{2m-2}P}^2 g_{2^{2m-1}P}). \end{aligned}$$

Letting

$$f_P = \prod_{i=1}^{2m} g_{2^{i-1}P}^{2^{2m-i}} = g_P^{2^{2m-1}} g_{2P}^{2^{2m-2}} \cdots g_{2^{2m-2}P}^2 g_{2^{2m-1}P}, \tag{7}$$

we have $2^{2m}(P) - 2^{2m}(O) = (2^{2m}P) - (O) + (f_P)$ and $(P) - (O) = (P) - (O) + (1)$. Thus the equation (2) of the Miller's formula again says $(2^{2m} + 1)\{(P) - (O)\} = (f_P \ell_P)$ because $2^{2m}P = -P$. Note that the line ℓ_P can also be omitted in the actual computation in view of [14]. Therefore after adjusting the irrelevant factors, we can say that

$$(f_P) = (2^{2m} + 1)\{(P) - (O)\} = \tfrac{2^{2m}+1}{l} \cdot \{l(P) - l(O)\} = \tfrac{2^{2m}+1}{l}(f'_P), \tag{8}$$

where f'_P is a rational function satisfying $l(P) - l(O) = (f'_P)$. Thus we have the Tate pairing

$$\tau_l(P, Q) = f'_P(\phi(Q))^{\frac{2^{4m}-1}{l}} = f'_P(\phi(Q))^{\frac{2^{2m}+1}{l}(2^{2m}-1)} = f_P(\phi(Q))^{2^{2m}-1}. \tag{9}$$

From the equation (7), the rational function f_P is just a product of the functions of the form $g_{2^{i-1}P}$ which can be regarded as the tangent line at the point $2^{i-1}P$. Thus all we have to do is to find an explicit expression of $f_P = \prod_{i=1}^{2m} g_{2^{i-1}P}^{2^{2m-i}}$.

Lemma 2. *Let $P = (\alpha, \beta), Q = (x, y)$ be points in $E_b(\mathbb{F}_{2^m})$. Then one has the value of $\{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i}} = \{g_{2^{i-1}P}(x + s^2, y + sx + t)\}^{2^{2m-i}}$ as*

$$\{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i}} = \alpha^{(i-1)}x^{(-i)} + \beta^{(i-1)} + y^{(-i)} + s(\alpha^{(i-1)} + x^{(-i)}) + t + b,$$

where $g_R(X, Y) = \ell_{R,R}$ is an equation of the tangent line at R.

Proof. The tangent line at $P = (\alpha, \beta)$ on the curve $E_b : Y^2 + Y = X^3 + X + b$ is $Y = (\alpha^2 + 1)X + \beta^2 + b$. Thus we have $2(P) - 2(O) = (2P) - (O) + (\frac{g_P}{\ell_{2P}})$ where

$$g_P(x, y) = (\alpha^2 + 1)x + \beta^2 + b - y, \tag{10}$$

and ℓ_{2P} is the vertical line intersecting $2P$ and $-2P$. Since ℓ_{2P} can be removed without affecting the pairing value, we are mainly interested in the computations of the lines $g_{2^{i-1}P}$. Using the equation (5), one has $g_{2^{i-1}P}(x, y) = (\alpha^{(2i-1)} + i)x + \beta^{(2i-1)} + (i-1)\alpha^{(2i-1)} + \epsilon_i + b - y$. Therefore, by applying the distortion map ϕ to the point $Q = (x, y)$, we get

$$\begin{aligned} g_{2^{i-1}P}(x + s^2, y + sx + t) = (\alpha^{(2i-1)} + i)(x + s^2) + \beta^{(2i-1)} \\ + (i-1)\alpha^{(2i-1)} + \epsilon_i + b - (y + sx + t). \end{aligned} \tag{11}$$

Taking 2^{2m-i}-th power of both sides of the above equality,

$$\begin{aligned} &\{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i}} \\ &= (\alpha^{(i-1)} + i)(x^{(2m-i)} + s^{(2m-i+1)}) + \beta^{(i-1)} + (i-1)\alpha^{(i-1)} + \epsilon_i + b \\ &\quad - (y^{(2m-i)} + s^{(2m-i)}x^{(2m-i)} + t^{(2m-i)}) \\ &= \alpha^{(i-1)}x^{(2m-i)} + \{i - s^{(2m-i)}\}x^{(2m-i)} + \{s^{(2m-i+1)} + i - 1\}\alpha^{(i-1)} \\ &\quad + \beta^{(i-1)} + b - y^{(2m-i)} + \{is^{(2m-i+1)} + \epsilon_i - t^{(2m-i)}\}. \end{aligned} \tag{12}$$

From $s^2 + s + 1 = 0$, we have $s^{(2)} = s^4 = s, s^{(3)} = s + 1, s^{(4)} = s, \cdots$. That is,

$$s^{(j)} = s + j. \tag{13}$$

The coefficients $i - s^{(2m-i)}$ (resp. $i - 1 + s^{(2m-i+1)}$) of $x^{(2m-i)}$ (resp. $\alpha^{(i-1)}$) in the equation (12) have a unique value equal to s independent of the choices of i because i and $2m - i$ always have the same parity and we are in the binary field. In other words, for any $i \geq 0$, we get

$$i - s^{(2m-i)} = i + 2m - i + s = s. \tag{14}$$

From $t^2 = t+s$, we have $t^{(2)} = t^{2^2} = t+s+s^2 = t+1, t^{(3)} = t^{2^3} = t+s+1, t^{(4)} = t+s+s^2+1 = t, t^{(5)} = t^2 = t+s, \cdots$. Therefore, for any $j \geq 0$, we have

$$t^{(4j)} = t, \quad t^{(4j+1)} = t+s, \quad t^{(4j+2)} = t+1, \quad t^{(4j+3)} = t+s+1. \tag{15}$$

Now using the equations (6),(13),(15), it is trivial to show that the last term of the equation (12) has the value

$$is^{(2m-i+1)} + \epsilon_i - t^{(2m-i)} = t \tag{16}$$

independent of the choices of i. This can be proved as follows. Since the extension degree m is odd, we may write $m = 2j+1$ for some j. Therefore one has $is^{(2m-i+1)} + \epsilon_i - t^{(2m-i)} = is^{(4j+3-i)} + \epsilon_i - t^{(4j+2-i)}$. By taking $i \pmod 4$ and noticing that our field has characteristic *two*, we easily get the equation (16). Since x, y, α, β are all in $\mathbb{F}_{2^m}$, the values $x^{(j)}, y^{(j)}, \alpha^{(j)}, \beta^{(j)}$ are determined up to the residue classes of $j \pmod m$ and $x^{(j)}$ with $j \in \mathbb{Z}$ (resp. $y^{(j)}, \alpha^{(j)}, \beta^{(j)}$) is understood as $x^{(j)} = x^{2^{j'}}$ where j', $0 \leq j' \leq m-1$, is a unique integer satisfying $j' \equiv j \pmod m$. Therefore, using (14) and (16) in the equation (12), we are done. □

Theorem 3. *One has the Tate pairing* $\tau_l(P,Q) = f_P(\phi(Q))^{2^{2m}-1}$ *where*

$$f_P(\phi(Q)) = \prod_{i=1}^{m} \{\alpha^{(i)} x^{(-i+1)} + \beta^{(i)} + y^{(-i+1)} + s^2(\alpha^{(i)} + x^{(-i+1)}) + t^2 + b\}.$$

Proof. Lemma 2 implies that $\{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i}}$ is depending only on the residue classes of $i \pmod m$. Thus, from (7) and (9), we have $f_P(\phi(Q)) = \prod_{i=1}^{2m} \{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i}} = \prod_{i=1}^{m} \{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i} \cdot 2} = \prod_{i=1}^{m} \{\alpha^{(i)} x^{(-i+1)} + \beta^{(i)} + y^{(-i+1)} + s^2(\alpha^{(i)} + x^{(-i+1)}) + t^2 + b\}$. □

4.3 Closed Formula of the Tate Pairing for $Y^2 + Y = X^3$

The curve $E : Y^2 + Y = X^3$ has the embedding degree $k = 2$ and is not so interesting in terms of the bandwidth. However using the same techniques in the previous section, we can derive a similar closed formula for the pairing computation. That is, by defining the distortion map $\phi : E \longrightarrow E$ as $\phi(x,y) = (x+1, y+x+t)$ with $t^2+t+1 = 0$, we have $\{g_{2^{i-1}P}(\phi(Q))\}^{2^{m-i}} = \alpha^{(i-1)} x^{(-i)} + (\alpha+\beta)^{(i-1)} + (x+y)^{(-i)} + t$, where $P = (\alpha, \beta)$ and $Q = (x,y)$ are the points in $E(\mathbb{F}_{2^m})$. Therefore

Theorem 4. *One has the Tate pairing* $\tau_l(P,Q) = f_P(\phi(Q))^{2^m-1}$ *where*

$$f_P(\phi(Q)) = \prod_{i=1}^{m} \{\alpha^{(i-1)} x^{(-i)} + (\alpha+\beta)^{(i-1)} + (x+y)^{(-i)} + t\},$$

and f_P *is a rational function satisfying* $(2^m+1)\{(P) - (O)\}$.

5 Field Arithmetic for the Computation of $f_P(\phi(Q))$

In Theorem 3, using $s^2 = t^2 + t + 1$, we may write $\alpha^{(i)}x^{(-i+1)} + \beta^{(i)} + y^{(-i+1)} + s^2(\alpha^{(i)} + x^{(-i+1)}) + t^2 + b = w + zt + (z+1)t^2$, where

$$z = \alpha^{(i)} + x^{(-i+1)}, \quad w = z + \alpha^{(i)}x^{(-i+1)} + \beta^{(i)} + y^{(-i+1)} + b. \tag{17}$$

Letting $C = c_0 + c_1 t + c_2 t^2 + c_3 t^3$, $c_i \in \mathbb{F}_{2^m}$, be the partial product in the computation of $f_P(\phi(Q))$, we have $C \cdot (w + zt + (z+1)t^2) = c'_0 + c'_1 t + c'_2 t^2 + c'_3 t^3$, where $c'_0 = c_0 w + (c_2 + c_3)(z+1) + c_3$, $c'_1 = c_0 w + (c_1 + c_2 + c_3)w + (c_0 + c_2 + c_3)(w + z + 1) + c_3(z+1) + c_0 + c_3$, $c'_2 = c_0 w + (c_1 + c_2 + c_3)w + (c_0 + c_2 + c_3)(w + z + 1) + (c_1 + c_2)(w + z + 1) + c_1$ and $c'_3 = (c_1 + c_2 + c_3)w + (c_1 + c_2)(w + z + 1) + c_2$. Therefore one needs 6 $\mathbb{F}_{2^m}$-multiplications for the computation of $C \cdot (w + zt + (z+1)t^2)$ with respect to the basis $\{1, t, t^2, t^3\}$. One may also use the basis $\{1, s, t, st\}$ to get the same result.

Table 1. An algorithm for computing $f_P(\phi(Q))$

Input: $P = (\alpha, \beta), Q = (x, y)$
Output: $C = f_P(\phi(Q))$
$C \leftarrow 1$
for ($i = 1$ to m ; $i++$)
$\alpha \leftarrow \alpha^2, \quad \beta \leftarrow \beta^2$
$z \leftarrow \alpha + x, \quad w \leftarrow z + \alpha x + \beta + y + b$
$C \leftarrow C \cdot (w + zt + (z+1)t^2)$
$x \leftarrow x^{2^{m-1}}, \quad y \leftarrow y^{2^{m-1}}$
end for

If we ignore the costs of (inverse) Frobenius maps and $\mathbb{F}_{2^m}$-additions, we find that exactly 7 $\mathbb{F}_{2^m}$-multiplications are needed in each round of the for-loop, where the computation of w needs one multiplication in $\mathbb{F}_{2^m}$ and the computation of C needs 6 multiplications in $\mathbb{F}_{2^m}$. Compare our result with the similar result in $\mathbb{F}_{3^m}$ case of Duursma and Lee where each step of the algorithm in [10] requires 14 $\mathbb{F}_{3^m}$-multiplications [8,9] with loop unfolding technique.

6 Algorithms Without Inverse Frobenius Operations

Many computational evidence [8,23] imply that a more efficient field arithmetic can be obtained for small characteristic finite fields by using a polynomial basis than a normal basis, especially for software purposes. Though a Gaussian normal basis of low complexity [27] is a good choice for a fast arithmetic, such basis does not appear quite frequently when compared with a polynomial basis of low hamming weight (like trinomial or pentanomial). Granger et al. [8] showed that, even though a cube root operation (inverse Frobenius operation for characteristic *three*) in a polynomial basis is tricky, an algorithm for the

Tate pairing computation with a polynomial basis outperforms a method with a normal basis since the cost of a multiplication with a normal basis is quite expensive than that of a polynomial basis in general situations. Based on the idea of Vercauteren [8], Granger et al. showed that a cube root operation in $\mathbb{F}_{3^m}$ has roughly the same cost as 2/3 multiplication in $\mathbb{F}_{3^m}$ with a small amount of precomputation. A similar method for the characteristic *two* case is discussed by Fong et al. [26] so that one can show that the cost of one square root operation is roughly equal to the cost of 1/2 multiplication with a precomputation. In fact, as pointed out by Harrison [18], the cost of one inverse Frobenius (square or cube root) operation is almost equal to the cost of one Frobenius operation when the given irreducible polynomial is a trinomial. However for a general case where no irreducible trinomial exists, the computation is not so simple and even in the case of pentanomial basis, inverse Frobenius operation is quite costly compared with Frobenius operation.

6.1 Avoiding Square Root Operation

Let us define A_i as $A_i = \{\alpha^{(i)}x^{(-i+1)} + \beta^{(i)} + y^{(-i+1)} + s^2(\alpha^{(i)} + x^{(-i+1)}) + t^2 + b\}^{2^{3m+i}} = \alpha^{(2i)}x^2 + \beta^{(2i)} + y^2 + s^{(i)}(\alpha^{(2i)} + x^2) + t^{(m-1+i)} + b$, where we used the fact that $s^{(j)}$ is determined up to $j \pmod 2$ with $3m+1 \equiv 0 \pmod 2$ and $t^{(j)}$ is determined up to $j \pmod 4$ with $3m+1 \equiv m-1 \pmod 4$ as is clear from the equations (13) and (15). Then the expression of $f_P(\phi(Q))$ in Theorem 3 can be rewritten as $f_P(\phi(Q)) = \prod_{i=1}^{m} A_i^{2^{m-i}} = (\cdots(((A_1)^2A_2)^2A_3)^2\cdots)^2A_m$. Using the cyclic property of $t^{(j)}$ in the equation (15), it is not difficult to see that, for all indices $1 \le i \le m$, A_i can be written as $A_i = A_i(t) = w + zt + (z+1)t^2$ for some z and w in $\mathbb{F}_{2^m}$. Thus, similarly as in the previous section, one needs 6 $\mathbb{F}_{2^m}$-multiplications for computing $C \cdot A_i(t)$ with respect to the basis $\{1, t, t^2, t^3\}$ for any $C \in \mathbb{F}_{2^{4m}}$. We now have the following algorithm for computing $f_P(\phi(Q))$ which avoids inverse Frobenius operations.

Table 2. An algorithm for computing $f_P(\phi(Q))$ without square root operations

Input: $P = (\alpha, \beta), Q = (x, y)$
Output: $C = f_P(\phi(Q))$
$C \leftarrow 1$
$u \leftarrow x^2, \quad v \leftarrow u, \quad y \leftarrow y^2$
for ($i = 1$ to m ; $i++$)
$\alpha \leftarrow \alpha^4, \quad \beta \leftarrow \beta^4$
$A(t) \leftarrow \alpha(v+1) + u + \beta + y + b + \frac{m-1}{2} + (\alpha + v)t + (\alpha + v + 1)t^2$
$C \leftarrow C^2 \cdot A(t)$
$u \leftarrow u + v + 1, \quad v \leftarrow v + 1$
end for

Note that the coefficients of $A_i(t)$ depend on the values of $s^{(i)}$ and $t^{(m-1+i)}$ and they are recursively computed by the relation (13) and (15). We also have the initial values $s^{(1)} = s^2 = t^2 + t + 1$ and $t^{(m)} = t^2 + \frac{m-1}{2}$. In each step of the above

algorithm, one needs 7 $\mathbb{F}_{2^m}$-multiplications which is same to the algorithm in Table 1. Since the operation $C \leftarrow C^2$ needs 4 squaring operations in $\mathbb{F}_{2^m}$ and since the operations $\alpha \leftarrow \alpha^4$, $\beta \leftarrow \beta^4$ also need 4 squaring operations, the total number of necessary squaring is 8 in this new algorithm. On the other hand, the algorithm in Table 1 needs 2 squaring and 2 square root operations. Therefore our new algorithm in Table 2 is a more optimal choice if one is interested in the implementation with arbitrary polynomial basis (especially for hardware purpose) since this new algorithm uses 6 Frobenius operations instead of using 2 inverse Frobenius operations.

6.2 Avoiding Cube Root Operation from the Algorithm of Duursma and Lee

Duursma and Lee [10] found a closed formula for the following supersingular elliptic curves defined over $\mathbb{F}_{3^m}$ with m prime to 6, $E_b : Y^2 = X^3 - X + b$, $b = \pm 1$. For the above mentioned curves, the following nontrivial automorphism $\phi : E_b \longrightarrow E_b$ with $\phi(x, y) = (\rho - x, \sigma y)$ is used, where $\sigma^2 + 1 = 0$ and $\rho^3 - \rho - b = 0$. That is, $\mathbb{F}_3(\sigma) = \mathbb{F}_{3^2}$ and $\mathbb{F}_3(\rho) = \mathbb{F}_{3^3}$. A closed formula of Duursma and Lee says that, for $P = (\alpha, \beta)$ and $Q = (x, y)$ in $E[l](\mathbb{F}_{3^m})$, the Tate pairing can be written as $\tau_l(P, Q) = f_P(\phi(Q))^{3^{3m}-1}$ with $f_P(\phi(Q)) = \prod_{i=1}^{m} B_i$ where $B_i = -\sigma\beta^{(i)}y^{(-i+1)} - (\alpha^{(i)} + x^{(-i+1)} - \rho + b)^2$ and f_P is a rational function satisfying $(f_P) = (3^{3m} + 1)\{(P) - (O)\}$. Now let us define $A_i \in \mathbb{F}_{3^{6m}}$ as $A_i = B_i^{3^{5m+i}} = -\sigma^{(5m+i)}\beta^{(2i)}y^{(1)} - (\alpha^{(2i)} + x^{(1)} - \rho^{(5m+i)} + b)^2 = (-1)^{i+1}\sigma\beta^{(2i)}y^{(1)} - (\alpha^{(2i)} + x^{(1)} - \rho + (m + 1 - i)b)^2$, where we used the relations $\sigma^{(j)} = (-1)^j\sigma$ and $\rho^{(j)} = \rho + jb$. Thus, from $B_i = A_i^{3^{m-i}}$, we get $f_P(\phi(Q)) = \prod_{i=1}^{m} A_i^{3^{m-i}} = (\cdots(((A_1)^3A_2)^3A_3)^3\cdots)^3A_m$. Letting $\mu = \alpha^{(2i)} + x^{(1)} + (m + 1 - i)b \in \mathbb{F}_{3^m}$ and $\lambda = (-1)^{i+1}\sigma\beta^{(2i)}y^{(1)} - \mu^2 \in \mathbb{F}_{3^{2m}}$, one finds $A_i = \lambda - \mu\rho - \rho^2$. Therefore the modified algorithm is given as follows.

Table 3. A modified Duursma-Lee algorithm without cube root operations

Input: $P = (\alpha, \beta), Q = (x, y)$
Output: $C = f_P(\phi(Q))$
$C \leftarrow 1$
$x \leftarrow x^3, \quad y \leftarrow y^3, \quad d \leftarrow mb$
for ($i = 1$ to m ; $i++$)
$\alpha \leftarrow \alpha^9, \quad \beta \leftarrow \beta^9$
$\mu = \alpha + x + d, \quad \lambda = \sigma\beta y - \mu^2$
$C \leftarrow C^3 \cdot (\lambda - \mu\rho - \rho^2)$
$y \leftarrow -y, \quad d \leftarrow d - b$
end for

In each step of the above algorithm, the number of necessary multiplications in $\mathbb{F}_{3^m}$ is same to that of the original algorithm of Duursma and Lee. Since the cube operation $C \leftarrow C^3$ with respect to the basis $\{1, \rho, \rho^2\}$ over $\mathbb{F}_{3^{2m}}$ costs 6 cube operations in $\mathbb{F}_{3^m}$ and since the operations $\alpha \leftarrow \alpha^9$, $\beta \leftarrow \beta^9$ cost 4 cube

operations in $\mathbb{F}_{3^m}$, the total number of necessary Frobenius operations in each step of the above algorithm is 10. Note that the original Duursma-Lee algorithm needs 2 Frobenius operations plus 2 inverse Frobenius operations. Therefore our modified algorithm uses 8 Frobenius operations instead of using 2 inverse Frobenius operations. With arbitrary polynomial basis, it is safe to believe that the cost of 4 cube operations is cheaper than the cost of one cube root operation.

7 Conclusions

In this paper we showed that an efficient closed formula can be derived for the Tate pairing computation for supersingular elliptic curves over a binary field $\mathbb{F}_{2^m}$ of odd dimension. There are exactly three isomorphism classes of supersingular elliptic curves over $\mathbb{F}_{2^m}$ with m odd and our method is applicable to all these curves. Each step of our algorithm requires two inverse Frobenius operations like the characteristic three case of Duursma and Lee. To overcome the computational complexity of the inverse Frobenius operation with arbitrary polynomial basis, we modified our algorithm and the algorithm of Duursma and Lee, and presented another closed formula which does not need any inverse Frobenius operation, which is especially useful for polynomial basis arithmetic.

Acknowledgements: The author would like to thank Robert Granger and Keith Harrison who made valuable suggestions on the preprint version of this paper. Also thanks are due to the anonymous referees for their many helpful comments. Finally, this work was supported by grant No. R01-2005-000-11261-0 from Korea Science and Engineering Foundation in Ministry of Science & Technology.

References

1. A.J. Menezes, T. Okamoto, and S.A. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field," *IEEE Trans. Information Theory*, vol. 39, pp. 1639–1646, 1993.
2. G. Frey and H. Rück, "A remark concerning m-divisibility and the discrete logarithm in the divisor class groups of curves," *Math. Comp.*, vol. 62, pp. 865–874, 1994.
3. D. Boneh and M. Franklin, "Identity based encryption from the Weil pairing," *Crypto 2001, Lecture Notes in Computer Science*, vol. 2139, pp. 213–229, 2001.
4. D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing," *Asiacrypt 2001, Lecture Notes in Computer Science*, vol. 2248, pp. 514–532, 2002.
5. A. Joux, "A one round protocol for tripartite Diffie-Hellman," *ANTS 2000, Lecture Notes in Computer Science*, vol. 1838, pp. 385–394, 2000.
6. R. Sakai, K. Ohgishi, and M. Kasahara, "Cryptosystems based on pairing," *SICS 2000, Symposium on Cryptography and Information Security*, pp. 26–28, 2000.
7. N.P. Smart, "An identity based authentication key agreement protocol based on pairing," *Electronics Letters*, vol. 38, pp. 630–632, 2002.
8. R. Granger, D. Page, and M. Stam, "Hardware and software normal basis arithmetic for pairing based cryptography in characteristic three," preprint, *available at* http://eprint.iacr. org/2004/157.pdf, 2004.

9. R. Granger, D. Page, and M. Stam, "On small characteristic algebraic tori in pairing based cryptography," preprint *available at* http://eprint.iacr.org/2004/132.pdf, 2004.
10. I. Duursma and H. Lee, "Tate pairing implementation for hyperelliptic curves $y^2 = x^p - x + d$," *Asiacrypt 2003, Lecture Notes in Computer Science*, vol. 2894, pp. 111–123, 2003.
11. K. Eisenträger, K. Lauter, and P.L. Montgomery, "Improved Weil and Tate pairing for elliptic and hyperelliptic curves," preprint, 2004.
12. A. Miyaji, M. Nakabayashi, and S. Takano, "New explicit conditions of elliptic curve trace for FR-reduction," *IEICE Trans. Fundamentals*, vol. E84 A, pp. 1–10, 2001.
13. J.H. Silverman, *The Arithmetic of Elliptic Curves*, Springer-Verlag, 1985.
14. P. Barreto, H. Kim, B. Lynn, and M. Scott, "Efficient algorithms for pairing based cryptosystems," *Crypto 2002, Lecture Notes in Computer Science*, vol. 2442, pp. 354–368, 2002.
15. F. Hess, "A Note on the Tate pairing of curves over finite fields," *Arch. Math.* vol. 82, pp. 28–32, 2004.
16. F. Hess, "Efficient identity based signature schemes based on pairings," *SAC 2002, Lecture Notes in Computer Science*, vol. 2595, 310-324, 2003.
17. A.J. Menezes, *Elliptic Curve Public Key Cryptosystems*, Kluwer Academic Publisher, 1993.
18. K. Harrison, *Personal Communications*, 2004.
19. J.C. Cha and J.H. Cheon, "An identity-based signature from gap Diffie-Hellman groups," *PKC 2003, Lecture Notes in Computer Science*, vol. 2567, 18-30, 2003.
20. K. Rubin and A. Silverberg "Torus based cryptography," *Crypto 2003, Lecture Notes in Computer Science*, vol. 2729, pp. 349–365, 2003.
21. S. Galbraith, K. Harrison, and D. Soldera, "Implementing the Tate pairing," *ANTS 2002, Lecture Notes in Computer Science*, vol. 2369, pp. 324–337, 2002.
22. V. Miller, "Short programs for functions on curves," *unpublished manuscript*, 1986.
23. D. Hankerson, J.L. Hernandez, and A.J. Menezes, "Software implementation of elliptic curve cryptography over binary fields," *CHES 2000, Lecture Notes in Computer Science*, vol. 1965, pp. 1–24 , 2000.
24. S. Galbraith, "Supersingular curves in cryptography," *Asiacrypt 2001, Lecture Notes in Computer Science*, vol. 2248, pp. 495–513, 2001.
25. E.R. Verheul, "Evidence that XTR is more secure than supersingular elliptic curve cryptosystems," *Eurocrypt 2001, Lecture Notes in Computer Science*, vol. 2045, pp. 195–210, 2001.
26. K. Fong, D. Hankerson, J. López, and A. Menezes, "Field inversion and point halving revisited," *Technical Report CORR 2003-18*, Univ. of Waterloo, 2003.
27. S. Gao, J. von zur Gathen, and D. Panario, "Gauss periods and fast exponentiation in finite fields," *Latin 1995, Lecture Notes in Computer Science*, vol. 911, pp. 311–322, 1995.
28. J. Baek and Y. Zheng, "Identity-based threshold signature scheme from the bilinear pairings," *ITCC 2004, Proceedings of International Conference on Information Technology*, vol 1, pp. 124–128, 2004.
29. P. Gaudry, F. Hess, and N.P. Smart, "Constructive and destructive facets of Weil descent on elliptic curves," *J. of Cryptology*, vol. 15, pp. 19–46, 2002.
30. N. Koblitz, A. Menezes, and S. Vanstone, "The state of elliptic curve cryptography," *Design, Codes and Cryptography*, vol. 19, pp. 173–193, 2000.

A Complete Divisor Class Halving Algorithm for Hyperelliptic Curve Cryptosystems of Genus Two

Izuru Kitamura[1], Masanobu Katagi[1], and Tsuyoshi Takagi[2⋆]

[1] Sony Corporation, 6-7-35 Kitashinagawa Shinagawa-ku, Tokyo, 141-0001 Japan
{Izuru.Kitamura, Masanobu.Katagi}@jp.sony.com
[2] Future University - Hakodate, 116-2 Kamedanakano-cho Hakodate, 041-8655, Japan
takagi@fun.ac.jp

Abstract. We deal with a divisor class halving algorithm on hyperelliptic curve cryptosystems (HECC), which can be used for scalar multiplication, instead of a doubling algorithm. It is not obvious how to construct a halving algorithm, due to the complicated addition formula of hyperelliptic curves. In this paper, we propose the first halving algorithm used for HECC of genus 2, which is as efficient as the previously known doubling algorithm. From the explicit formula of the doubling algorithm, we can generate some equations whose common solutions contain the halved value. From these equations we derive four specific equations and show an algorithm that selects the proper halved value using two trace computations in the worst case. If a base point is fixed, we can reduce these extra field operations by using a pre-computed table which shows the correct halving divisor class — the improvement over the previously known fastest doubling algorithm is up to about 10%. This halving algorithm is applicable to DSA and DH based on HECC. Finally, we present the divisor class halving algorithms for not only the most frequent case but also other exceptional cases.

1 Introduction

We know from recent research that hyperelliptic curve cryptosystems (HECC) of small genus are competing with elliptic curve cryptosystems (ECC) [Ava04, Lan02a-c, PWG+03]. With an eye to further improvement of HECC we utilize its abundant algebraic structure to make HECC faster in scalar multiplication than ECC. Lange and Duquesne independently showed that Montgomery scalar multiplication is applicable to HECC [Lan04a, Duq04]. We expect other fast algorithms used for ECC can also be efficiently implemented in HECC.

A point halving algorithm is one of the effective algorithms on ECC and the algorithm tries to find a point P such that $2P = Q$ for a given point Q. Knudsen and Schroeppel independently proposed a point halving algorithm for ECC

⋆ This work was carried out when the author was in Technische Universität Darmstadt, Fachbereich Informatik, Hochschulstr.10, D-64289 Darmstadt, Germany

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 146–157, 2005.

over binary fields $\mathbb{F}_{2^n}$ [Knu99, Sch00]. Their algorithm is faster than a doubling algorithm. Moreover, there has been growing consideration of the point halving algorithm, showing, for instance, a fast implementation [FHL+03], an application for Koblitz curve [ACF04], and an improvement of curves with cofactor 4 [KR04]. The explicit doubling formula of HECC (denoted by HECDBL) is more complicated than that of ECC. It is not obvious how the algorithm of Knudsen and Schroeppel can extend to HECC.

In this paper, we propose a divisor class halving algorithm applied to HECC with genus 2 over binary fields. Let $D = (U, V)$ be a reduced divisor, where $U = x^2 + u_1 x + u_0$ and $V = v_1 x + v_0$. The doubled divisor class $2D$ can be represented as polynomials over $\mathbb{F}_{2^n}$ with coefficients u_1, u_0, v_1, v_0 and curve parameters $y^2 + h(x)y = f(x)$. We report two crucial quadratic equations which compute some candidates of the halved values. These equations are derived from the property: an equation of degree 6 appeared in the doubling algorithm can be divided by $x^4 + u_1^2 x^2 + u_0^2$. We also show a criterion and an algorithm selecting the correct divisor class from two candidates. The correct divisor class can be efficiently found if the polynomial $h(x)$ is irreducible. In order to select the correct halved value, we perform some test calculations, and notice that the number of operations can be reduced if the correct halving value is first found. We develop a divisor class halving algorithm used for not only the most frequent case but other exceptional cases, e.g. the weight of input divisor class is 1. The proposed algorithm can be optimized with careful considerations of the basic operations.

This paper is organized as follows: in Section 2 we review the algorithms of a hyperelliptic curve. In Section 3 we present our proposed divisor class halving algorithm for HECC, and compare it with existing doubling formulae. In Section 4 a complete divisor class halving algorithm is shown. In Section 5 we consider a halving algorithm for a special curve, $\deg h = 1$. Section 6 is our conclusion.

2 Hyperelliptic Curve

We review the hyperelliptic curve used in this work.

Let $\mathbb{F}_{2^n}$ be a binary finite field with 2^n elements. A hyperelliptic curve C of genus g over $\mathbb{F}_{2^n}$ with one point at infinity is defined by $C : y^2 + h(x)y = f(x)$, where $f(x) \in \mathbb{F}_{2^n}[x]$ is a monic polynomial of degree $2g + 1$ and $h(x) \in \mathbb{F}_{2^n}[x]$ is a polynomial of degree at most g, and curve C has no singular point. Let $P_i = (x_i, y_i) \in \overline{\mathbb{F}}_{2^n} \times \overline{\mathbb{F}}_{2^n}$ be a point on curve C and P_∞ be a point at infinity, where $\overline{\mathbb{F}}_{2^n}$ is the algebraic closure of $\mathbb{F}_{2^n}$. The inverse of $P_i = (x_i, y_i)$ is the point $-P_i = (x_i, y_i + h(x_i))$. P is called a ramification point if $P = -P$ holds. A divisor is a formal sum of points: $D = \sum m_i P_i, m_i \in \mathbb{Z}$. A semi-reduced divisor is given by $D = \sum m_i P_i - (\sum m_i) P_\infty$, where $m_i \geq 0$ and $P_i \neq -P_j$ for $i \neq j$, and semi-reduced divisor D is called reduced if $\sum m_i \leq g$ holds. The weight of a reduced divisor D is defined as $\sum m_i$, and we denote it by $w(D)$. Jacobian $\mathbf{J}$ is isomorphic to the divisor class group which forms an additive group. Each divisor class can be represented uniquely by a reduced divisor and so we can identify the set of points on the Jacobian with the set of reduced

divisors and assume this identification from now on. The reduced and the semi-reduced divisors are expressed by a pair of polynomials (u, v), which satisfies the following conditions [Mum84]:

$$u(x) = \prod (x + x_i)^{m_i}, v(x_i) = y_i, \deg v < \deg u, v^2 + hv + f \equiv 0 \bmod u.$$

A divisor class is defined over $\mathbb{F}_{2^n}$ if the representing polynomials u, v are defined over this field and the set of $\mathbb{F}_{2^n}$-rational points of the Jacobian is denoted by $\mathbf{J}(\mathbb{F}_{2^n})$. Note that even if $u, v \in \mathbb{F}_{2^n}[x]$, the coordinates x_i and y_i may be in extension field of $\mathbb{F}_{2^n}$. The degree of u equals the weight of the reduced divisor and we represent the zero element by $O = (1, 0)$.

To compute the additive group law of $\mathbf{J}(\mathbb{F}_{2^n})$, Cantor gave an addition algorithm which is applicable to a hyperelliptic curve of any genus. However, this algorithm is relatively slow due to its generality. Harley then proposed an efficient addition and doubling algorithm for a hyperelliptic curve of genus 2 over $\mathbb{F}_p$ [GH00, Har00a, Har00b]. This algorithm achieved speeding up by detailed classification into the most frequent case and some exceptional cases. This classification allows us to avoid extra field operations. Sugizaki et al. expanded the Harley algorithm to HECC over $\mathbb{F}_{2^n}$ [SMC+02], and around the same time Lange expanded the Harley algorithm to HECC over general finite field [Lan02a]. The most frequent case of doubling algorithm HECDBL is defined as follows:

Algorithm 1 HECDBL

Input: $D_1 = (U_1, V_1)$, D_1 has no ramification points.
Output: $D_2 = (U_2, V_2) = 2D_1$, $U_i = x^2 + u_{i1}x + u_{i0}, V_i = v_{i1}x + v_{i0}$, where $i = 1, 2$

1. $U_1' \leftarrow U_1^2$	*4.* $U_2' \leftarrow (f + hV_1' + V_1'^2)/U_1'$
2. $S \leftarrow (f + hV_1 + V_1^2)/U_1$	*5.* $U_2 \leftarrow$ MakeMonic (U_2')
$S \leftarrow Sh^{-1} \bmod U_1$	*6.* $V_2 \leftarrow V_1' + h \bmod U_2$
3. $V_1' \leftarrow SU_1 + V_1$	*7.* return (U_2, V_2)

In HECDBL, from Step 1 to Step 3 is called the composition part and from Step 4 to Step 6 is called the reduction part. From Algorithm 1, it is clear that the number of field operations depends on the curve parameters. To reduce the number of field operations, in previous works, a transformed curve $y^2 + (x^2 + h_1'x + h_0')y = x^5 + f_3'x^3 + \cdots + f_0'$, via isomorphic transformations: $y \rightarrow h_2^5 y$ and $x \rightarrow h_2^2 x + f_4$, is used. We call this transformed curve a *general curve*.

In this paper, our aim is to present the divisor class halving algorithm for the general curve. Additionally, we consider a simple polynomial $h(x) = h_1 x + h_0$ and we call this curve a *special curve*. In a cryptographic application, we are only interested in a curve whose order of $\mathbf{J}(\mathbb{F}_{2^n})$ is $2 \times r$, i.e. whose cofactor is two, where r is a large prime number. Note that the cofactor is always divisible by 2 [KKT05]. Moreover, as inputs and outputs for the halving and doubling algorithm we use the divisor classes whose order is r.

3 Proposed Halving Algorithm for General Curve

In this section we propose a divisor class halving algorithm (HECHLV) on hyperelliptic curve cryptosystems of genus two. We derive HECHLV by inverse com-

puting of HECDBL. For HECHLV, the significant problem is to find the *missing polynomial* k such that $V_1' + h = kU_2 + V_2$ in Algorithm 1. First, we compute k by a reverse operation of the reduction part, then the semi-reduced divisor (U_1', V_1') via k, at last $D_1 = \frac{1}{2}D_2$ by a reverse operation of the composition part.

3.1 Main Idea

We follow the opposite path to HECDBL. From Step 6 of HECDBL, there is a unique polynomial $k = k_1x + k_0$ such that $V_1' + h = (k_1x + k_0)U_2 + V_2$. Substituting V_1' to equation $(f + hV_1' + {V_1'}^2)$ appeared in Step 4, the following relationship yields:

$$U_2'U_1' = f + h(kU_2 + V_2) + k^2U_2^2 + V_2^2. \tag{1}$$

Because the doubled divisor class (U_2, V_2) is known, we can obtain the relationship between k and U_1'. Note that $U_2' = k_1^2U_2$ from the highest term of equation (1). Recall that $U_1' = U_1^2$ from Step 1, namely, we know

$$U_1' = x^4 + u_{11}^2x^2 + u_{10}^2. \tag{2}$$

In other words, the coefficients of degree 3 and 1 are zero. From this observation, there are polynomials whose solutions includes k_0 and k_1. In our algorithm we try to find k_0 and k_1 by solving the polynomials. Once k_0 and k_1 are calculated, we can easily compute the halved divisor class $D_1 = (U_1, V_1)$ from equation (1). We describe the sketch of the proposed algorithm in the following.

Algorithm 2 *Sketch* HECHLV

Input: $D_2 = (U_2, V_2)$
Output: $D_1 = (U_1, V_1) = \frac{1}{2}D_2$, $U_i = x^2 + u_{i1}x + u_{i0}, V_i = v_{i1}x + v_{i0}$, where $i = 1, 2$

1. **determine** $k = k_1x + k_0$ **by the reverse operation of the reduction part**
   1.1 $V_1' \leftarrow V_2 + h + kU_2$, $k = k_1x + k_0$, $U_1' \leftarrow (f + hV_1' + {V_1'}^2)/(k_1^2U_2)$
   1.2 derive k_0, k_1 from two equations $\mathrm{coeff}(U_1', 3) = 0$ and $\mathrm{coeff}(U_1', 1) = 0$
2. **compute** $U_1' = x^4 + u_{11}^2x^2 + u_{10}^2$ **in the semi-reduced divisor by using** k_0, k_1
   2.1 compute u_{11}^2, u_{10}^2 by substituting k_0, k_1 in $\mathrm{coeff}(U_1', 2)$, $\mathrm{coeff}(U_1', 0)$
3. **compute** U_1, V_1 **by the reverse operation of the composition part**
   3.1 $U_1 \leftarrow \sqrt{U_1'} = x^2 + u_{11}x + u_{10}$, $V_1 \leftarrow V_2 + h + kU_2 \bmod U_1$
4. return (U_1, V_1)

We explain Algorithm 2 in detail. The coeff(U, i) is the coefficient of x^i in polynomial U. In Step 1.1, we compute polynomial U_1' in equation (1):

$$\begin{aligned}
\mathrm{coeff}(U_1', 3) &= (k_1h_2 + k_1^2u_{21} + 1)/k_1^2 \\
\mathrm{coeff}(U_1', 2) &= (k_1h_1 + k_0h_2 + k_1^2u_{20} + k_0^2 + c_2)/k_1^2 \\
\mathrm{coeff}(U_1', 1) &= (k_1h_0 + k_0h_1 + k_0^2u_{21} + c_1)/k_1^2 \\
\mathrm{coeff}(U_1', 0) &= (k_0h_0 + k_0^2u_{20} + c_0)/k_1^2,
\end{aligned}$$

where

$$c_2 = f_4 + u_{21}, \quad c_1 = f_3 + h_2 v_{21} + u_{20} + c_2 u_{21},$$
$$c_0 = f_2 + h_2 v_{20} + h_1 v_{21} + v_{21}^2 + c_2 u_{20} + c_1 u_{21}.$$

Equation (2) yields the explicit relationship between variables k_0, k_1, u_{11}, and u_{10}:

$$k_1 h_2 + k_1^2 u_{21} + 1 = 0 \tag{3}$$

$$k_1 h_0 + k_0 h_1 + k_0^2 u_{21} + c_1 = 0 \tag{4}$$

$$u_{11} = \sqrt{k_1 h_1 + k_0 h_2 + k_1^2 u_{20} + k_0^2 + c_2}/k_1 \tag{5}$$

$$u_{10} = \sqrt{k_0 h_0 + k_0^2 u_{20} + c_0}/k_1 \tag{6}$$

In the algorithm we used the following lemma in order to uniquely find k_0, k_1. The proof of this lemma is shown in the full version of this paper [KKT05].

Lemma 1. *Let $h(x)$ be an irreducible polynomial of degree 2. There is only one value k_1 which satisfies both equations (3) and (4). Equation (4) has a solution only for the correct k_1. There is only one value k_0 which yields the halved divisor class D_1 in algorithm 2. Equation $xh_2 + x^2 u_{11} + 1 = 0$ has a solution only for the correct k_0.*

After calculating k_0, k_1, we can easily compute u_{11}, u_{10}, v_{11}, and v_{10} via equations (5), (6), and $V_1 \leftarrow V_2 + h + (k_1 x + k_0)U_2 \bmod U_1$.

3.2 Proposed Algorithm

We make the assumption that the polynomial h has degree two and is irreducible. We present the proposed algorithm in Algorithm 3.

The proposed algorithm requires to solve quadratic equations. It is well known that equation $ax^2 + bx + c = 0$ has roots if and only if $\mathrm{Tr}(ac/b^2) = 0$. Let one root of $ax^2 + bx + c = 0$ be x_0, then the other root be $x_0 + b/a$. If this equation has roots, i.e. $\mathrm{Tr}(ac/b^2) = 0$, then we can solve this equation by using half trace, namely $x_0 = H(ac/b^2), x_0' = x_0 + b/a$. This equation has no root if $\mathrm{Tr}(ac/b^2) = 1$.

We explain the proposed algorithm as follows. The correctness of this algorithm is shown in Lemma 1. In Step 1, we compute two roots k_1 and k_1' of equation (3). In Step 2, the correct k_1 is selected by checking the trace of equation (4). Then we obtain two solutions k_0 and k_0' of equation (4). In Step 3, the correct k_0 is selected by checking trace of $xh_2 + x^2 u_{11} + 1 = 0$. In Steps 4 and 5 we compute the halved divisor class.

Algorithm 3 HECHLV

Input: $D_2 = (U_2, V_2)$
Output: $D_1 = (U_1, V_1) = \frac{1}{2}D_2$
$U_i = x^2 + u_{i1}x + u_{i0},\ V_i = v_{i1}x + v_{i0},$ where $i = 1, 2,\ h_2 \neq 0$

step	*procedure*
1.	**Solve** $k_1h_2 + k_1^2u_{21} + 1 = 0$
	$\alpha \leftarrow h_2/u_{21}, \gamma \leftarrow u_{21}/h_2^2,\ k_1 \leftarrow H(\gamma)\alpha,\ k_1' \leftarrow k_1 + \alpha$
2.	**Select correct** k_1 **by solving** $k_1h_0 + k_0h_1 + k_0^2u_{21} + c_1 = 0$
	$c_2 \leftarrow f_4 + u_{21},\ c_1 \leftarrow f_3 + h_2v_{21} + u_{20} + c_2u_{21},$
	$c_0 \leftarrow f_2 + h_2v_{20} + h_1v_{21} + v_{21}^2 + c_2u_{20} + c_1u_{21}, \alpha \leftarrow h_1/u_{21},$
	$w \leftarrow u_{21}/h_1^2,\ \gamma \leftarrow (c_1 + k_1h_0)w$
	if $\mathrm{Tr}(\gamma) = 1$ then $k_1 \leftarrow k_1',\ \gamma \leftarrow (c_1 + k_1h_0)w$
	$k_0 \leftarrow H(\gamma)\alpha,\ k_0' \leftarrow k_0 + \alpha$
3.	**Select correct** k_0 **by checking trace of** $xh_2 + x^2u_{11} + 1 = 0$
	$u_{11} \leftarrow \sqrt{k_1h_1 + k_0h_2 + k_1^2u_{20} + k_0^2 + c_2}/k_1,\ \gamma \leftarrow u_{11}/h_2^2$
	if $\mathrm{Tr}(\gamma) = 1$ then $k_0 \leftarrow k_0',\ u_{11} \leftarrow \sqrt{k_1h_1 + k_0h_2 + k_1^2u_{20} + k_0^2 + c_2}/k_1$
4.	**Compute** U_1
	$u_{10} \leftarrow \sqrt{k_0h_0 + k_0^2u_{20} + c_0}/k_1$
5.	**Compute** $V_1 = V_2 + h + kU_2 \bmod U_1$
	$w \leftarrow h_2 + k_1u_{21} + k_0 + k_1u_{11}$
	$v_{11} \leftarrow v_{21} + h_1 + k_1u_{20} + k_0u_{21} + u_{10}k_1 + u_{11}w,\ v_{10} \leftarrow v_{20} + h_0 + k_0u_{20} + u_{10}w$
6.	$D_1 \leftarrow (x^2 + u_{11}x + u_{10}, v_{11}x + v_{10}),$ return D_1

3.3 Complexity and Improvement

In order to estimate the complexity of HECHLV shown in Algorithm 3, we consider four cases with respect to the selection of k_1 and k_0. When we get incorrect k_1 and k_0 (k_1' and k_0' are correct) in Steps 1 and 2, respectively, we have to replace $k_1 \leftarrow k_1'$, $k_0 \leftarrow k_0'$ and compute γ, u_{11} again in Steps 2 and 3, respectively. In the worst case this requires $4M+1SR$ as additional field operations compared to the best case, and we have another two cases: one is k_0 and k_1' are correct and the other is k_0' and k_1 are correct. Note that a multiplication by M for short and other operations are expressed as follows: a squaring (S), an inversion (I), a square root (SR), a half trace (H), and a trace (T). Our experimental observations found that these four cases occur with almost the same probability. Therefore, we employ the average of these four cases as the average case.

Now we consider how to optimize the field operations in Algorithm 3. We will discuss the optimization under the two topics: choices of the curve parameter and scalar multiplication using a fixed base point.

Choices of the curve parameter. The complexity of HECHLV depends on the coefficients of the curve. If the coefficients are small, one, or zero, we reduce some field operations. Firstly, we reduce some inversion operations to one. If $1/h_1^2$ and $1/h_2^2$ are allowed as inputs, we reduce two inversion operations and we compute $1/k_1 = h_2 + k_1u_{21}$ from equation (3), then Algorithm 3 requires only one inversion operation $1/u_{21}$. Secondly, we use the general curve. When $f_4 = 0$ we reduce $3M$ to $1M + 1S$ by $c_2u_{21} = u_{21}^2$ and $c_2u_{20} + c_1u_{21} = u_{21}(u_{20} + c_1)$.

When $h_2 = 1$, two multiplications by h_2 and two multiplications by $1/h_2^2$ are omitted. Thirdly, we use the general curve when $h_1 = 1$. In this case, we change $1M$ to $1S$ by $v_{21}(h_1 + v_{21}) = v_{21} + v_{21}^2$, where $1S$ is faster than $1M$, and two multiplications by h_1 and one multiplication by $1/h_1^2$ are reduced. Finally, we use the general curve when $h_1 = h_0 = 1$ then we skip one multiplication $k_1 h_0$. We summarize these improvements in Algorithm 4.

Algorithm 4 HECHLV ($h_2 = 1, f_4 = 0$)

Input: $D_2 = (U_2, V_2), 1/h_1^2$
Output: $D_1 = (U_1, V_1) = \frac{1}{2} D_2$
$U_i = x^2 + u_{i1}x + u_{i0},\ V_i = v_{i1}x + v_{i0},$ where $i = 1, 2$

step	*procedure*	*cost*
1.	**Solve** $k_1 + k_1^2 u_{21} + 1 = 0$	$1M + 1I + 1H$
	$\alpha \leftarrow 1/u_{21},\ k_1 \leftarrow H(u_{21})\alpha,\ k_1' \leftarrow k_1 + \alpha$	
2.	**Select correct** k_1	$9M + 1S + 1H + 1T$
	by solving $k_1 h_0 + k_0 h_1 + k_0^2 u_{21} + c_1 = 0$	
	$c_1 \leftarrow f_3 + v_{21} + u_{20} + u_{21}^2$	
	$c_0 \leftarrow f_2 + v_{20} + v_{21}(h_1 + v_{21}) + u_{21}(u_{20} + c_1)$	$(h_1 = 1 : v_{21}(h_1 + v_{21}) = v_{21} + v_{21}^2)$
	$w_0 \leftarrow u_{21}/h_1^2,\ \alpha \leftarrow h_1\alpha, \gamma \leftarrow (c_1 + k_1 h_0)w_0$	
	if $\mathrm{Tr}(\gamma) = 1$ then $k_1 \leftarrow k_1',\ \gamma \leftarrow (c_1 + k_1 h_0)w_0$	$(h_1 = 1 : \gamma \leftarrow \gamma + h_0)$
	$k_0 \leftarrow H(\gamma)\alpha,\ k_0' \leftarrow k_0 + \alpha$	
3.	**Select correct** k_0 **by solving** $x + x^2 u_{11} + 1 = 0$	$5M + 1S + 2SR + 1T$
	$w_0 \leftarrow k_1^2,\ w_1 \leftarrow w_0 u_{20} + k_1 h_1 + u_{21}$	
	$w_2 \leftarrow k_0 + \sqrt{w_1 + k_0},\ w_4 \leftarrow k_1 u_{21} + 1,\ u_{11} \leftarrow w_2 w_4$	
	if $\mathrm{Tr}(u_{11}) = 1$ then	
	$k_0 \leftarrow k_0',\ w_2 \leftarrow k_0 + \sqrt{w_1 + k_0},\ u_{11} \leftarrow w_2 w_4$	
4.	**Compute** U_1	$4M + 1SR$
	$w_1 \leftarrow k_0 u_{20}, w_5 \leftarrow w_4 + 1, w_6 \leftarrow (k_0 + k_1)(u_{20} + u_{21})$	
	$u_{10} \leftarrow w_4 \sqrt{k_0(w_1 + h_0) + c_0}$	
5.	**Compute** $V_1 = V_2 + h + kU_2 \bmod U_1$	$2M$
	$w_4 \leftarrow w_5 + k_0 + 1,\ w_5 \leftarrow w_1 + w_5 + w_6 + v_{21} + h_1$	
	$w_6 \leftarrow w_1 + v_{20} + h_0,\ w_7 \leftarrow w_2 + w_4$	
	$w_1 \leftarrow w_7 u_{10},\ w_3 \leftarrow (k_1 + w_7)(u_{10} + u_{11})$	
	$v_{11} \leftarrow w_1 + w_2 + w_3 + w_5,\ v_{10} \leftarrow w_1 + w_6$	
6.	$D_1 \leftarrow (x^2 + u_{11}x + u_{10}, v_{11}x + v_{10}),$ return D_1	
total	(k_1, k_0) *is correct*	$18M + 2S + 1I + 2SR + 2H + 2T$
	(k_1, k_0') *is correct*	$19M + 2S + 1I + 3SR + 2H + 2T$
	(k_1', k_0) *is correct*	$20M + 2S + 1I + 2SR + 2H + 2T$
	(k_1', k_0') *is correct*	$21M + 2S + 1I + 3SR + 2H + 2T$
	$h_1 = 1$	
	(k_1, k_0) *or* (k_1', k_0) *is correct*	$14M + 3S + 1I + 2SR + 2H + 2T$
	(k_1, k_0') *or* (k_1', k_0') *is correct*	$15M + 3S + 1I + 3SR + 2H + 2T$
	$h_1 = h_0 = 1$	
	(k_1, k_0) *or* (k_1', k_0) *is correct*	$13M + 3S + 1I + 2SR + 2H + 2T$
	(k_1, k_0') *or* (k_1', k_0') *is correct*	$14M + 3S + 1I + 3SR + 2H + 2T$

Scalar multiplication with a fixed base point. We explain the scalar multiplication using divisor class halvings. Knudsen and Schroeppel proposed the ECC scalar multiplication algorithm, halve-and-add binary method, which replaces point doublings in double-and-add binary methods with point halvings. Similarly, the halve-and-add binary method can be applied to HECC via the divisor class halving proposed in Algorithm 4. In the case of scalar multiplication with a fixed base point D, we improve a computation method of $\frac{1}{2^i} D$ via a pre-computed table which shows whether $k_1(k_0)$ or $k_1'(k_0')$ is the correct value in each halving, since whether $k_1(k_0)$ is correct or not depends on D. This improvement can be applied to a right-to-left binary method by adding $\frac{1}{2^i} D$. We adopt this table-lookup method to the general curve [KKT05] which requires only $18M + 2S +$

$1I + 2SR + 2H$. The pre-computed table requires only the same bit length as D since D needs $4n$ bits while the scalar value has length $2n$ and we need two bits to encode the right choices of k_1 and k_0.

3.4 Comparison of Doubling and Halving

We compare field operations cost of doubling algorithms to halving algorithms. Table 1 provides a comparison of HECDBL and the above halving algorithms in the average case.

Table 1. Comparison of Halving and Doubling

Scheme	HECHLV	HECDBL [LS04]
$h_2 = 1, f_4 = 0$	$19.5M + 2S + 1I + 2.5SR + 2H + 2T$	$21M + 5S + 1I$
random base point	$(27.5M_N, 29.95M_P)$	$(29M_N, 29.5M_P)$
$h_2 = 1, f_4 = 0$	$18M + 2S + 1I + 2SR + 2H$	—
fixed base point	$(26M_N, 28.2M_P)$	—
$h_2 = h_1 = 1, f_4 = 0$	$14.5M + 3S + 1I + 2.5SR + 2H + 2T$	$18M + 7S + 1I$
random base point	$(22.5M_N, 25.05M_P)$	$(26M_N, 26.7M_P)$
$h_2 = h_1 = 1, f_4 = 0$	$14M + 3S + 1I + 2SR + 2H$	—
fixed base point	$(22M_N, 24.3M_P)$	—
$h_2 = h_1 = h_0 = 1, f_4 = 0$	$13.5M + 3S + 1I + 2.5SR + 2H + 2T$	$15M + 7S + 1I$
random base point	$(21.5M_N, 24.05M_P)$	$(23M_N, 23.7M_P)$
$h_1 = h_1 = h_0 = 1, f_4 = 0$	$13M + 3S + 1I + 2SR + 2H$	—
fixed base point	$(21M_N, 23.3M_P)$	—

By using the normal basis, we can neglect the computation time of a squaring, a square root, a half trace, and a trace compared to that of a field multiplication or an inversion [Knu99]. Menezes [Men93] showed that an inversion operation requires $\lfloor \log_2(n-1) \rfloor + \#(n-1) - 1$ multiplications, where $\#(n-1)$ is the number of 1's in the binary representation of $n-1$. By neglecting these operations, for the general curve, HECHLV and HECDBL require $19.5M_N + 1I$ and $21M_N + 1I$, respectively, where M_N is a multiplication over the normal basis. When we assume $1I = 8M_N$ HECHLV and HECDBL require $27.5M_N$ and $29M_N$, respectively.

On the other hand, by using the polynomial basis, we cannot ignore the computation time of a squaring, a square root, and a half trace. Assuming that $1S = 0.1M_P$, $1SR = 0.5M_P$, $1H = 0.5M_P$, and $1I = 8M_P$, where M_P is a multiplication over the polynomial basis. For the general curve, HECHLV and HECDBL require $29.95M_P$ and $29.5M_P$, respectively. By selecting the polynomial basis, however, we can compute these arithmetic faster than half the time of multiplication, and there is a possibility to reduce the cost of these operations.

Table 1 shows that when we use the normal basis HECHLV is faster than HECDBL for all the cases. On the contrary by using the polynomial basis, HECHLV is faster than HECDBL when $h_2 = h_1 = 1$ and $f_4 = 0$, especially the improvement by using a fixed base point over HECDBL is up to about 10%.

4 Complete Procedures for Divisor Class Halving Algorithm

In the previous sections, we proposed the halving algorithm, which corresponds to the most frequent case in the doubling algorithm. However, we also have to consider several exceptional procedures for giving complete procedures of the halving algorithm. These cases appear with very low probability, but we cannot ignore them. Therefore, we have to implement these procedures in order to perform the scalar multiplication correctly. In this paper we only deal with a divisor class whose order is r (not order $2 \times r$), and thus the divisor class does not include any ramification points. Therefore, we have to consider four inverse operations of HECDBL, HECDBL$^{2\to1}$, HECDBL$^{1\to2}$, and HECDBL$^{2\to2}$ as follows:

HECDBL: $w(D_1) = 2,\ w(D_2) = 2,\ D_2 = 2D_1$.
HECDBL$^{2\to1}$: $w(D_1) = 2,\ w(D_2) = 1,\ D_2 = 2D_1$.
HECDBL$^{1\to2}$: $w(D_1) = 1,\ w(D_2) = 2,\ u_{21} = 0, D_2 = 2D_1$.
HECDBL$^{2\to2}$: $w(D_1) = 2,\ w(D_2) = 2,\ u_{21} = 0, D_2 = 2D_1$.

Note that HECDBL$^{2\to2}$ is computed via HECDBL. In the halving algorithm, however, we have to care HECDBL$^{2\to2}$ because the inverse map of HECDBL$^{2\to2}$ is indistinguishable from the inverse map of HECDBL$^{1\to2}$. Therefore, the halving algorithms can be classified into four cases: HECHLV, HECHLV$^{1\to2}$, HECHLV$^{2\to2}$, and HECHLV$^{2\to1}$. These cases are inverse maps of HECDBL, HECDBL$^{2\to1}$, HECDBL$^{2\to2}$, and HECDBL$^{1\to2}$, respectively. The Complete HECHLV is as follows:

Algorithm 5 Complete HECHLV

Input: $D_2 = (U_2, V_2)$
Output: $D_1 = (U_1, V_1) = \frac{1}{2}D_2$
$U_i = u_{i2}x^2 + u_{i1}x + u_{i0}, V_i = v_{i1}x + v_{i0}, u_{i2} \in \mathbb{F}_2,$ where $i = 1, 2,\ h_2 \neq 0$

step	*procedure*
1.	HECHLV$^{1\to2}$**:** $w(D_2) = 1,\ w(D_1) = 2$
	if $\deg U_2 = 1$ then $D_1 \leftarrow$ HECHLV$^{1\to2}(D_2)$, return D_1
2.	HECHLV$^{2\to1}$**:** $w(D_2) = 2, w(D_1) = 1, u_{21} = 0$ or
	HECHLV$^{2\to2}$**:** $w(D_2) = 2, w(D_1) = 2, u_{21} = 0$
	if $\deg U_2 = 2$ and $u_{21} = 0$ then $D_1 \leftarrow$ HECHLV$^{2\to2}(D_2)$, return D_1
3.	HECHLV**:** $w(D_2) = w(D_1) = 2,\ u_{21} \neq 0$
	if $\deg U_2 = 2$ and $u_{21} \neq 0$ then $D_1 \leftarrow$ HECHLV(D_2), return D_1

In the following paragraphs we explain exceptional procedures. The explicit algorithms are presented in [KKT05].

HECHLV$^{1\to2}$. A divisor class halving algorithm HECHLV$^{1\to2}$ is the analogy of HECHLV. The main difference between HECHLV$^{1\to2}$ and HECHLV is weight of input D_2. For example, in HECHLV$^{1\to2}$, $f + hV_1' + {V_1'}^2$ is a monic polynomial with degree five because of $\deg(V_1') = 2$ and U_2 is a monic polynomial, so $U_1' \leftarrow (f + hV_1' + {V_1'}^2)/U_2$ not divided by k_1^2 like HECHLV.

$\mathsf{HECHLV}^{2\to 1}$. In this case, $D_1 = (x+u_{10}, v_{10})$ is computed by reverse operation of $\mathsf{HECDBL}^{1\to 2}$. $D_2 = (x^2+u_{20}, v_{21}x+v_{20}) = 2D_1$ is computed as follows: $x^2+u_{20} = (x+u_{10})^2$, $v_{21} = (u_{10}^4+f_3u_{10}^2+f_1+h_1v_{10})/h(u_{10})$, and $v_{20} = v_{10}+v_{21}u_{10}$. Then we can easily express u_{10}, v_{10} by u_{20}, v_{21}, v_{20} and curve parameters by $u_{10} = \sqrt{u_{20}}$, $v_{10} = (v_{21}h(u_{10}) + u_{10}^4 + f_3u_{10}^2 + f_1)/h_1$.

$\mathsf{HECHLV}^{2\to 2}$. In the case $u_{21} = 0$, there are two candidate of $\frac{1}{2}D_2$: $D_1 = (x + \sqrt{u_{20}}, v_2(\sqrt{u_{20}}))$ and $D_1' = (x^2+u_{11}x+u_{10}, v_{11}x+v_{10})$. If D_1 is a correct divisor class, we use $\mathsf{HECHLV}^{2\to 1}$. On the other hand, if D_1' is a correct one, we use $\mathsf{HECHLV}^{2\to 2}$. We need to select a correct algorithm $\mathsf{HECHLV}^{2\to 1}$ or $\mathsf{HECHLV}^{2\to 2}$ as follows: First, we assume that D_1' is a correct divisor class, second compute u_{11}, then check the trace of $xh_2+x^2u_{11}+1 = 0$. If $\mathrm{Tr}(u_{11}/h_2^2) = 0$, D_1' is correct, then select the algorithm $\mathsf{HECHLV}^{2\to 2}$. If $\mathrm{Tr}(u_{11}/h_2^2) = 1$, D_1 is correct, then select the algorithm $\mathsf{HECHLV}^{2\to 1}$.

$\mathsf{HECHLV}^{2\to 2}$ is similar to HECHLV. The main difference between $\mathsf{HECHLV}^{2\to 2}$ and HECHLV is a value of u_{21}, e.g. $u_{21} = 0$ in $\mathsf{HECHLV}^{2\to 2}$ and $u_{21} \neq 0$ in HECHLV. In $\mathsf{HECHLV}^{2\to 2}$, $u_{21} = 0$ leads to that there is only one value k_1 not two values like HECHLV.

5 Halving Algorithm for Other Curves

In this section, we focus on other curves: (1) $h(x)$ is reducible in $\mathbb{F}_{2^n}$ with $\deg h = 2$, and (2)the special curve with $\deg h = 1$, i.e. $h_2 = 0$.

Let $h(x)$ be a reducible polynomial of degree 2, namely $h(x) = (x+x_1)(x+x_2)$ where $x_1, x_2 \in \mathbb{F}_{2^n}$. Assume that $x_1 \neq x_2$, then there are three different divisor classes of order 2, say D_1, D_2, and D_3 [KKT05]. In this case, Lemma 1 is no longer true, and there are four different candidates of the halved value arisen from equation (3) and equation (4). They are equal to $\frac{1}{2}D$, $\frac{1}{2}D + D_1$, $\frac{1}{2}D + D_2$, and $\frac{1}{2}D + D_3$. In order to determine the proper divisor class, we have to check the trace of both equation (3) and (4). Therefore the halving algorithm for this case requires more number of field operations than that required for the general curve. If $x_1 = x_2$ holds, we know $h_1 = 0$ and there is only one divisor class of degree 2. In this case, equation (4) has a unique root for each solution k_1 of equation (3), namely we have only two candidates of the halved value. It can be distinguished by the trace of equation $xh_2 + x^2h_{11} + 1 = 0$ as we discussed in Lemma 1.

For the special curve of $\deg h = 1$, we have only one value k_1 not two, recall for the general curve, there are two value k_1 and k_1' and we need to select the correct one. This is the main difference between the general curve and the special curve. For the special curve, we obtain a system of equations related to variables k_0, k_1, u_{11}, and u_{10} by the same method for the general curve.

$$k_1^2u_{21} + 1 = 0 \tag{7}$$

$$k_1h_0 + k_0h_1 + k_0^2u_{21} + c_1 = 0 \tag{8}$$

In the case of the general curve, we select correct k_0 by checking trace of the degree two equation of k_1 in next halving. If this equation has roots (no roots) i.e. trace is zero, k_0 is correct (not correct). However in the case of the special curve, we have only one value k_1 from equation (7), so we select correct k_0 by checking a degree two equation (8) of k_0 in next halving, instead of the equation of k_1. If the equation of k_0 in next halving has roots (no roots), k_0 is correct (not correct).

6 Conclusion

In this paper, we presented the first divisor class halving algorithm for HECC of genus 2, which is as efficient as the previously known doubling algorithm. The proposed formula is an extension of the halving formula for elliptic curves reported by Knudsen [Knu99] and Schroeppel [Sch00], in which the halved divisor classes are computed by solving some special equations that represent the doubled divisor class. Because the doubling formula for HECC is relatively complicated, the underlying halving algorithm is in general less efficient than that for elliptic curves. However, we specified two crucial equations whose common solutions contain the proper halved values, then an algorithm for distinguishing a proper value was presented. Our algorithm's improvement over the previously known fastest doubling algorithm is up to about 10%. Moreover, the proposed algorithm is complete — we investigated the exceptional procedures appearing in the divisor class halving algorithm, for example, operations with divisor classes whose weight is one. The presented algorithm has not been optimized yet, and there is a possibility to enhance its efficiency.

Acknowledgment. The authors thank Toru Akishita for helpful discussions.

References

[Ava04] R. Avanzi, "Aspects of Hyperelliptic Curves over Large Prime Fields in Software Implementations," CHES 2004, LNCS 3156, pp.148-162, 2004.

[ACF04] R. Avanzi, M. Ciet, and F. Sica, "Faster Scalar Multiplication on Koblitz Curves Combining Point Halving with the Frobenius Endomorphism," PKC 2004, LNCS 2947, pp.28-40, 2004.

[Can87] D. Cantor, "Computing in the Jacobian of a Hyperelliptic Curve," Mathematics of Computation, 48, 177, pp.95-101, 1987.

[Duq04] S. Duquesne, "Montgomery Scalar Multiplication for Genus 2 Curves," ANTS 2004, LNCS 3076, pp.153-168, 2004.

[FHL+03] K. Fong, D. Hankerson, J. López, and A. Menezes, "Field inversion and point halving revised," Technical Report CORR2003-18, http://www.cacr.math.uwaterloo.ca/techreports/2003/corr2003-18.pdf

[GH00] P. Gaudry and R. Harley, "Counting Points on Hyperelliptic Curves over Finite Fields," ANTS 2000, LNCS 1838, pp.313-332, 2000.

[HHM00] D. Hankerson, J. Hernandez, A. Menezes, "Software Implementation of Elliptic Curve Cryptography over Binary Fields," CHES 2000, LNCS 1965, pp.1-24, 2000.

[Har00a] R. Harley, "Adding.txt," 2000. http://cristal.inria.fr/~harley/hyper/
[Har00b] R. Harley, "Doubling.c," 2000. http://cristal.inria.fr/~harley/hyper/
[KKT05] I. Kitamura, M. Katagi, and T. Takagi, "A Complete Divisor Class Halving Algorithm for Hyperelliptic Curve Cryptosystems of Genus Two," Cryptology ePrint Archive, 2004/255, IACR, 2004.
[KR04] B. King and B. Rubin, "Improvements to the Point Halving Algorithm," ACISP 2004, LNCS 3108, pp.262-276, 2004.
[Kob89] N. Koblitz, "Hyperelliptic Cryptosystems," Journal of Cryptology, Vol.1, pp.139-150, 1989.
[Knu99] E. Knudsen, "Elliptic Scalar Multiplication Using Point Halving," ASIACRYPT '99, LNCS 1716, pp.135-149, 1999.
[Lan02a] T. Lange, "Efficient Arithmetic on Genus 2 Hyperelliptic Curves over Finite Fields via Explicit Formulae," Cryptology ePrint Archive, 2002/121, IACR, 2002.
[Lan02b] T. Lange, "Inversion-Free Arithmetic on Genus 2 Hyperelliptic Curves," Cryptology ePrint Archive, 2002/147, IACR, 2002.
[Lan02c] T. Lange, "Weighted Coordinates on Genus 2 Hyperelliptic Curves," Cryptology ePrint Archive, 2002/153, IACR, 2002.
[Lan04a] T. Lange, "Montgomery Addition for Genus Two Curves," ANTS 2004, LNCS 3076, pp.309-317, 2004.
[Lan04b] T. Lange, "Formulae for Arithmetic on Genus 2 Hyperelliptic Curves," J.AAECC Volume 15, Number 5, pp.295-328, 2005.
[LS04] T. Lange, M. Stevens, "Efficient Doubling on Genus Two Curves over Binary Fields," SAC 2004, pre-proceedings, pp.189-202, 2004.
[Men93] A. Menezes, *Elliptic Curve Public Key Cryptosystems*, Kluwer Academic Publishers, 1993.
[Mum84] D. Mumford, *Tata Lectures on Theta II*, Progress in Mathematics 43, Birkhäuser, 1984.
[MCT01] K. Matsuo, J. Chao, and S. Tsuji, "Fast Genus Two Hyperelliptic Curve Cryptosystems," Technical Report ISEC2001-31, IEICE Japan, pp.89-96, 2001.
[PWP03] J. Pelzl, T. Wollinger, and C. Paar, "High Performance Arithmetic for Hyperelliptic Curve Cryptosystems of Genus Two," Cryptology ePrint Archive, 2003/212, IACR, 2003.
[PWG+03] J. Pelzl, T. Wollinger, J. Guajardo and C. Paar, "Hyperelliptic Curve Cryptosystems: Closing the Performance Gap to Elliptic Curves," CHES 2003, LNCS 2779, pp.351-365, 2003.
[Sch00] R. Schroeppel, "Elliptic curve point halving wins big. 2nd Midwest Arithmetic Geometry in Cryptography Workshop, Urbana, Illinois, November 2000.
[SMC+02] T. Sugizaki, K. Matsuo, J. Chao, and S. Tsujii, "An Extension of Harley Addition Algorithm for Hyperelliptic Curves over Finite Fields of Characteristic Two," Technical Report ISEC2002-9, IEICE Japan, pp.49-56, 2002.

Using "Fair Forfeit" to Prevent Truncation Attacks on Mobile Agents

Min Yao, Kun Peng, and Ed Dawson

Information Security Institute, Queensland University of Technology
Brisbane, QLD 4000, Australia
{m.yao,k.peng,e.dawson}@qut.edu.au

Abstract. Protection of data integrity in mobile agents has drawn much attention in recent years. Various degrees of agent data integrity have been achieved by a number of proposed schemes. A known vulnerability of these published techniques is the truncation attack. In this paper we propose a "fair forfeit" technique to prevent the truncation attack. It also prevents other known attacks such as the modification, insertion and deletion attacks.

1 Introduction

Mobile agents are autonomous software entities that move code, data and state to remote hosts. They have great potential for electronic commerce applications.

We consider a scenario where a mobile agent is ordered to search for the best price of a specific product [11]. The agent migrates to multiple vendors' servers, collects price quotes and is free to choose its next move dynamically based on the data it acquired from its journey. The agent finally returns to the buyer with the offers of all the vendors. Using the agent data, the buyer chooses the best offer.

However, vendors (servers) may try to delete, replace, or invalidate the offers that have been collected by the agent. Hence, the integrity of the offers that are made by visited servers needs to be protected along the agent's journey.

Karjoth *et al.* [5] published a family of protocols - referred to as the KAG protocols - to ensure the integrity of the offers acquired from the visited hosts. A common vulnerability of these protocols is that they cannot resist the "truncation" attack. In this attack, a server currently visited by the agent colludes with a previously visited server to discard all the offers made between the two visits. A *stemming attack* is an extension of the truncation attack where one or more faked offers are inserted in place of the truncated data.

Possible solutions to the truncation attack have been proposed in the literature. Loureiro *et al* [6] published a technique to hash together a set of data blocks in an order-independent fashion that possesses a security property to defend against the truncation attack. However the technique can only prevent a truncation attack by a vendor that is not listed in the agent's itinerary.

Cheng and Wei [3] proposed a "co-signing" scheme to prevent the truncation attack, where a server S_{i-1} helps its successor server S_i sign its computed data

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 158–169, 2005.

O_i. If a malicious server S_i wishes to truncate a string of existing data, S_{i-1} must consent to re-sign the new offer O_i which includes an indicator of S_i's successor server S_{i+1}. It is however still vulnerable to the "colluding-servers" truncation if S_{i-1} conspires with S_i and is willing to co-sign the new offer.

Contribution. In this paper, we have devised a "fair forfeit" technique to deter servers from launching various attacks against the integrity of mobile agents' data. All of attacks, including modification, insertion, deletion, truncation and stemming, require the malicious server to perform re-computation on previously computed data. Especially, the truncation attack, a vulnerability existing in most of the current mobile agent systems, can be prevented.

We assume that, for fairness, every server can provide one and only one offer. The "fair forfeit" is realised by an "e-cash division" mechanism, which is accomplished by 2-out-of-2 secret sharing of each server's e-cash in such a way that with high probability, the e-cash of malicious servers becomes available for use by anyone. The main deterrent in this system is therefore exposure of e-cash of the malicious server, since the cost of losing e-cash is obvious.

In this paper we assume that the proposed technique can be implemented in an electronic market environment and that a trusted third party in the e-market plays a role of judge to settle any disputes rising.

Organisation. The rest of the paper is organised as follows: Section 2 describes related work. Section 3 proposes a new protocol to identify illegally repeated operations on the existing data and to detect the "truncation" and "stemming" attacks. We analyse the new protocol in terms of security and efficiency in Sect. 4. We conclude the paper in Sect. 5.

For ease of reading, the notation used in the paper is listed in Table 1.

2 Related Work

This section first discusses the attacks to the offer integrity of shopping mobile agents. We then briefly describe the e-cash technique that is employed in the new scheme.

2.1 Shopping Mobile Agents and Attacks

Common cryptographic techniques used to protect offer integrity for shopping mobile agents are digital signatures over the servers' offers to provide the signer's non-repudiation and data integrity, and *chaining relationships*.

The concept of the chaining relationship has been used often in the mobile agent applications. It was first noted by Karjoth *et. al* [5]. When the agent collects data from the server, the server must provide a short proof of the computation (such as a digital signature) that is stored in the agent. The *chaining relationship* is established between proofs by cryptographically linking each proof with the one computed at the previous site. This makes it impossible to modify an intermediate proof without modifying all the subsequent ones. The originator verifies the integrity of the "chain" of cryptographic proofs.

Notation	Meaning
$\prod$	An agent's code.
S_0	ID of the originator.
S_i, $1 \leq i \leq n$	ID of server i.
S_{0-1}, $1 \leq i \leq n$	ID of the last server in the agent's itinerary.
o_0	A secret possessed by S_0. It can be regarded as a dummy offer and is only known to the originator.
o_i, $1 \leq i \leq n$	An offer (a partial result) from S_i .
εo_i	An encrypted offer of server S_i
O_i	An encapsulated offer (cryptographically protected) from S_i.
(yi, xi)	A public/private key pair of server S_i.
c_i	Server S_i's e-cash.
(e_i, n_i); d_i	A public/private key pair of the issuing bank that issues c_i.
$(si_{\varepsilon_j}, si_{o_j})$	jth pair of shares in the signature of S_i''s e-cash.
si_{c_j}	A jth chosen share from pair $(si_{\varepsilon_j}, si_{o_j})$. si_{c_j} can be either si_{ε_j} or si_{o_j}.
(di_{e_j}, di_{o_j})	A pair of commitments corresponding to the pair of shares$(si_{\varepsilon_j}, si_{o_j})$, where $di_{\varepsilon_j} = si_{\varepsilon_j}^{e_i}$ and $yi_{o_j} = g^{xi_{o_j}}$.
di_{c_j}	A jth commitment to the chosen share si_{c_j} where $di_{c_j} = si_{c_j}^{e_i}$. di_{c_j} can be either di_{e_j} or di_{o_j}.
κ_i	A set of chosen shares of server S_i
hd_i	A hash value used by server S_i to select partial shares in a secret.
r_i	A nonce generated by S_i.
H(m)	A one-way collision-free hash function.
$E_{yi}\{m_i\}$	Message m_i encrypted with the encryption key yi associated with S_i
$\mathrm{Sig}_{xi}(m_i)$	A signature of S_i on the message m_i with xi.
$\mathrm{Ver}_{yi}(s_i, m_i)$ = true or false	Verification of the signature s_i on the message m_i. It is true when $\mathrm{Sig}_{xi}(m_i) = s_i$; false otherwise.
$S_i \rightarrow S_{i+1}$: m	Server S_i sending a message m to S_{i+1}.

Table 1. Notation used in this paper ($0 \leq i \leq n$ unless i is indicated)

The general form of a chaining relationship can be expressed as: ($h_i = H(O_{i-1}, I_{i+1})$) where $H()$ is a one-way hash function, O_{i-1} is the offer made by the previous server S_{i-1}, and I_{i+1} is the identity of its next server S_{i+1}. Each entry of the chain depends on some of the previous and succeeding members, therefore, any illegitimate change in O_{i-1} and/or I_{i+1} will invalidate the chaining relationship.

However, many of the schemes that use the chaining relationship are still vulnerable to the "truncation" and "stemming" attacks.

Truncation attack. An attacker S_i captures an agent with encapsulated offers O_0, O_1,... O_{i-1}, and colludes with a previously visited server S_m. The attack is launched when S_i, with the agent at hand, sends the agent back to S_m. Hence S_m can delete the offers between S_m and S_i from the agent. S_m sends the agent back to S_i after the truncation; the agent continues to execute on S_i. A special

case occurs when S_m sends the agent back to the originator after the attack. In this case, S_m and S_i both have to agree to sacrifice S_i's interest.

Stemming attack. This attack takes place in conjunction with the truncation attack. S_m inserts a series of fake offers under the names of victim hosts S'_{m+1}, S'_{m+2}...until S_i or the originator S_0. S_m first replaces its previous offer with O'_m using its own identity and a fake server S'_{m+1} as its successor. It signs offer O'_m using S_m's long term private key. For constructing fake offers O'_{m+1}, $O'_{m+2}, \ldots, O'_{i-1}$ (or O'_{0-1}), S_m arbitrarily chooses fake private keys and signs on behalf of the fake servers S'_{m+1}, S'_{m+2}..., S'_{i-1} (or S'_{0-1}).

There are some other attacks, such as *modification*, *insertion* and *deletion* [11], to offer integrity that also feature repeated operations. They require the malicious server S_m to replace either O_{m-1} or I_{m+1} in order to retain the validity of $h_m = H(O_{m-1}, I_{m+1})$ at S_m, or to recalculate O_m.

2.2 E-cash

E-cash is a self-authenticating digital payment instrument that can be stored in an electronic wallet and the electronic equivalent of real paper cash. E-cash is a type of pre-paid electronic payment where payers withdraw electronic cash from their bank accounts prior to making a purchase and payment. To make a payment, the payer simply passes the required amount of electronic cash to the payee. The payee is not referred to any bank account of the payer. E-cash typically comes in the form of electronic coins of various face values, to which digital signatures issued by the issuing banks are attached. Hence the basic form of e-cash is composed of two components: a data component that contains certain information such as the issuing bank, sum etc., and the signature component that is generated over the data. Any payee can immediately validate electronic coins by checking the signature on them against the public verifying key of the respective issuing bank.

E-cash was first introduced by Chaum *et. al* [2], which is based on the use of zero-knowledge proofs. The drawbacks of Chaum's proposal lie in the expensive computation real applications [8]. Some subsequent works [1][10][4] have achieved various improvements on Chaum's scheme.

E-cash must not be illegally forgeable and cannot be double spent. It must also provide anonymity to clients and untracebility to digital coins.

3 The New Protocol

Our new protocol is based on a proposed "fair forfeit" technique and attempts to provide deterrence to repeated operations in situations where only one-time operations are allowed.

This section first depicts an electronic market, in which the proposed mechanism and its application can be designed and implemented. A "e-cash division" technique is introduced and applied in a simple digital signature scheme to detect and prevent truncation and stemming attacks. It also prevents other attacks against integrity.

3.1 Architecture

The participants in the e-market in our setting include: (1) a buyer, (2) a number of vendors' servers and (3) a trusted third party. In Fig. **??**, a buyer's mobile agent enters the e-market through the trusted third party, which may provide yellow-page like services. At last the agent travels to the trusted third party to verify its collected results and finally returns to its originator.

The trusted third party plays an important role in the proposed new scheme. The trusted third party verifies the requesting servers' e-cash and registers them if the e-cash is valid. In case of a dispute between an e-market member and a customer or another member, the trusted third party serves as an arbitration board. In our architecture, the behavior of the trusted third party can be publicly verified by a prover. The prover functionality can be shared among multiple servers to distribute the trust and strengthen the robustness of the system.

3.2 The E-cash Division Mechanism

The "e-cash division" mechanism is employed to guarantee detection of illegally repeated operations. In the "e-cash division", an e-cash token c is divided into t pairs, where t is a security parameter in the system. Putting each pair together can recover the e-cash token.

As we discussed above, an e-cash token c has a data component b and a signature component s. Hence we denote $c = (b, s)$. c is divided into pairs as follows:
- In each pair, the data part b is divided into halves $(b_{\varepsilon_i}, b_{o_i})$, where b_{ε_i} and b_{o_i} denote one half in d with even and odd number subscripts respectively. $(b_{\varepsilon_i}, b_{o_i})$ are published as commitments.
- In each pair, the signature part s is divided into two halves $(s_{\varepsilon_i}, s_{o_i})$, where s_{ε_i} and s_{o_i} denote partial shares in S with even and odd number subscripts respectively. $(s_{\varepsilon_i}, s_{o_i})$ are kept secret. When both halves of the signature are put together, the e-cash is valid and useable, while only one half of the signature is unrecognised by the issuing bank and unusable.

The "e-cash division" technique can be realised using the RSA [7] signature scheme. Assume the e-cash issuing bank has RSA keys $(e, n; d)$, where pair (e, n) is the public key, and d is the corresponding secret key. Therefore $s = b^d \bmod n$.

A RSA signature can be divided as follows: choose s_{ε_i} and then $s_{o_i} = s/s_{\varepsilon_i}$; or vice versa. The following equations then obtained:

$b = s^e \bmod n$
$b = (s_{o_i} \times s_{\varepsilon_i})^e$
$b = (s_{o_i})^e \times (s_{\varepsilon_i})^e$

Let $(s_{o_i})^e = b_{o_i}$ and $(s_{\varepsilon_i})^e = b_{\varepsilon_i}$. Then $b = b_{o_i} \times b_{\varepsilon_i}$.

3.3 The New Protocol

We employ the "e-cash division" technique in mobile agent applications to defend against truncation attacks. The new protocol can be implemented in an

e-market, since we need a trusted third party to manage the commitments from the participant servers.

The protocol using "e-cash division" is illustrated in Fig. 1.

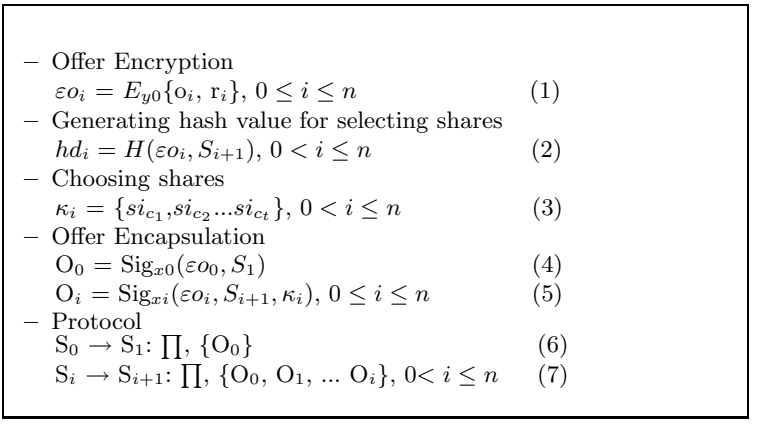


Fig. 1. The protocol using "e-cash division" mechanism

We divide the agent's journey into **Preparation**, **Execution** and **Finishing** phases. There are seven stages involved in these phases: *Setup*, *Offer encryption*, *Choose and Share*, *Sign*, *Verify*, *Update* and *Reveal*. All the stages except the *Reveal* stage are always performed in each protocol run. The *Reveal* stage is conducted only when a malicious action is detected. Any invalid data is detected in the *Verify* stage. The application contains three parties: the sending server, the receiving server and the trusted third party.

Preparation Phase

In this phase, all the participant servers in the e-market should purchase a bond in the form of e-cash from any bank that is recognised by the trusted third party. *Setup Stage.* The participant servers register with the trusted third party by revealing their e-cash. If the amount of the e-cash is correct and the signature of the issuing bank is valid, the trusted third party requests the servers to make commitments to the next transaction. Assume a server S_i has e-cash $c_i = (b_i, s_i)$ where $s_i = b_i^{e_i}$ with (e_i, n_i) as the public key of the issuing bank of c_i.

To make a commitment, S_i performs the following steps:

- S_i splits s_i, finding t equations such that:

$s_{\varepsilon_1} \times s_{o_1} = s_i \bmod n_i$ (1)

$s_{\varepsilon_2} \times s_{o_2} = s_i \bmod n_i$ (2)

...

$s_{\varepsilon_t} \times s_{o_t} = s_i \bmod n_i$ (t)

- S_i publishes b_i, (e_i, n_i) and $2t$ commitments $b_{\varepsilon_1} = s_{\varepsilon_1}^{e_i} \bmod n_i$, $b_{o_1} = s_{o_1}^{e_i} \bmod n_i$, $\cdots$, $b_{\varepsilon_t} = s_{\varepsilon_t}^{e_i} \bmod n_i$, $b_{o_t} = s_{o_t}^{e_i} \bmod n_i$. This commitment mechanism is

unconditionally binding, therefore at the end of the **Preparation** phase, the secret holder is unable to change its chosen shares. The commitments can be verified by anyone against b_i by testing:
$b_{\varepsilon_1} \times b_{o_1} = b_i$ (1)
$b_{\varepsilon_2} \times b_{o_2} = b_i$ (2)
...
$b_{\varepsilon_t} \times b_{o_t} = b_i$ (t)

where $b_{\varepsilon_i} \times b_{o_i} = s_{\varepsilon_i}^{e_i} \times s_{o_i}^{e_i} = (s_{\varepsilon_i} \times s_{o_i})^{e_i} = {s_i}^{e_i} = b_i$

In the *Setup* stage, the originator S_0 also initialises the protocol (shown in Fig. 1) by randomly generating r_0. It then encrypts r_0 and a secret token o_0 with its public key. S_0 signs this encrypted value together with the identity of S_1 to construct a dummy encapsulated offer O_0. Finally S_0 sends O_0 to the first server S_1. After the *Setup* stage is completed, the agent enters the e-market.

Execution Phase
From S_1 onwards, each server S_i will perform the following four stages:
Offer encryption stage. When the agent arrives at a server S_i, the server makes an offer o_i and also computes the identity of the next host S_{i+1}. Then S_i constructs εo_i by encrypting both o_i and a random value r_i with the public key of the agent's originator. Therefore only the agent's originator can retrieve o_i.

Choose and Share stage. Prior to sending the agent to S_{i+1}, S_i chooses t shares, each of which is a partial share of the half-divided signature of the e-cash. S_i sends these shares to S_{i+1} alongside the other data.

The shares are chosen using a hash-based algorithm:
1. S_i takes its encrypted offer εo_i and identity of the successor host S_{i+1} as inputs to a one-way hash function $hd_i = H(\varepsilon o_i, S_{i+1})$ that returns t bits output. We choose hd_i to be a string of bits $a_1 a_2 ... a_t$ since we have k sets of shares in this case.
2. S_i chooses one share si_{ε_j} or si_{o_j} out of the set (si_{ε_j},si_{o_j}) ($0 \leq j \leq t$) in the equations above according to the value of each bit in $a_1 a_2 ... a_t$. For example, S_i can choose the first share in equation (1) with even number subscript si_{ε_1} if $a_1 = 0$; or si_{o_2} if $a_1 = 1$. For convenience, we use a notation si_{c_1} to indicate a chosen share. si_{c_j} is either si_{ε_j} or si_{o_j} ($1 \leq j \leq t$). As such, t shares $\kappa_i = \{xi_{c_1}, xi_{c_2} ... xi_{c_k}\}$ are selected. The probability of producing two identical sets of shares in this case is $\frac{1}{2^t}$.

Note the same set of shares cannot be reused in different protocol runs. Otherwise the e-cash can be reconstructed and used by anyone who possesses it.

Sign stage. S_i constructs an encapsulated offer O_i by signing the encrypted offer εo_i, the identity of its next server S_{i+1} and κ_i.

Verify stage. During the agent's execution, a list of public keys of the participant servers should be either published and/or carried with the gent. The public keys

of the e-cash issuing banks should also be available to all entities in the e-market. The commitments of the participant servers can be stored locally in each server's database. When these public keys and commitments are available, each server in the agent's itinerary can verify offers obtained at previous servers.

When the agent arrives at server S_{i+1}, carrying a set of previously collected encapsulated offers $\{O_0, O_1, ... O_i\}$, S_{i+1} can conduct verification as follows:

- S_{i+1} obtains O_i from the chain and searches for the corresponding public key y_i from the key list. If $\text{Ver}_{y_i}(O_i) =$ true, S_{i+1} ensures that the offer signature is authentic. S_{i+1} recovers $\{o_i, r_i\}_{y_0}$, the identity of S_{i+1} and cs_i. S_{i+1} can not view o_i since it was encrypted using the originator's public key y_0. S_{i+1} verifies its own identity and the shares in κ_i.

- S_{i+1} gains shares $\{si_{c_1}, si_{c_2} ... si_{c_t}\}$ from κ_i and the searches for the corresponding public key (e_i, n_i) of the issuing bank. S_{i+1} computes $di'_{c_1} = si_{c_1}^{e_i} \bmod n_i$, $di'_{c_2} = si_{c_2}^{e_i} \bmod n_i ... \ di'_{c_t} = si_{c_t}^{e_i} \bmod n_i$. S_{i+1} also computes $hd_i = H(\varepsilon o_i, S_{i+1})$ and gains the t-bit string $a_1 a_2 ... a_t$ of hd_i. It searches the corresponding commitments in its own database and checks whether these equations are satisfied: If $a_j = 0$ $(1 \leq j \leq t)$, $di'_{c_j} \stackrel{?}{=} di_{e_j}$; else $a_j = 1$, $di'_{c_j} \stackrel{?}{=} di_{o_j}$. If this correspondence does not exist, S_{i+1} knows that S_i did not correctly choose the set of secret shares and should report this action to the trusted third party. As such S_{i+1} is able to verify κ_{i-1}, ..., κ_2, κ_1.

Following the same line of reasoning, O_0, O_1, ... O_{i-1} can be verified. If no integrity violation is detected and the shares are matched with the commitments, the agent continues its execution; otherwise, the agent's computation aborts early. In the latter case, S_{i+1} reports the abnormality to the trusted third party and sends the identity of the suspected server to the trusted party.

Finishing Phase

Once the agent visits all the servers, it arrives at the trusted third party prior to returning to the originator. The trusted third party verifies all the signatures and shares from all of the visited servers in the agent's itinerary. To detect if any illegally repeated operations have taken place, the trusted third party publishes all the received offers O_1,... O_n and their associated shares $\{s1'_{c_1}, s1'_{c_2} ... s1'_{c_t}\}$,... $\{sn'_{c_1}, sn'_{c_2} ... sn'_{c_t}\}$. If a visited server S_i ($1 < i \leq n$) discovers any mismatch between the published shares and its received shares, it contacts the trusted third party and sends the mismatched shares. We assume that only the trusted third party has the authority to reveal a dishonest server's long-term private key. The *reveal* stage then will be conducted.

Reveal stage. The trusted third party compares published shares $\{sj'_{c_1}, sj'_{c_2}, ..., sj'_{c_t}\}$ with the received shares $\{sj_{c_1}, sj_{c_2}, ..., sj_{c_t}\}$. If at any position two bits are not matched, the signature of the e-cash token can be reconstructed by simply computing the multiplication of those bits. For instance, if $sj'_{c_2} \neq sj_{c_2}$, then $sj = sj'_{c_2} \times sj_{c_2}$. This token becomes forfeited by the server.

If the trusted third party can successfully complete the verification, it performs the *Update* stage.

Update stage. At the beginning of *each* protocol execution, the trusted third party informs all the participant servers to choose a new set of shares. It erases the old commitments and publishes the new commitments. The servers must ensure that they do not use the same set of shares in two protocol runs, otherwise the signatures of their e-cash can be reconstructed. After these seven stages are completed, the trusted third party will dispatch the agent back to the originator (the buyer) with the collected data.

The protocol discussed above prevents a malicious server from any illegally repeated operations that corrupt the collected data chain, by making the server's e-cash available for use. The cost of losing its e-cash is obvious. This is the main deterrent in this system. The magnitude of this cost depends upon the amount of the e-cash that the participant server needs to purchase the good.

4 Security Analysis

Theorem 1. *If a host S_m computes hd_m more than once using different εo_m or S_{m+1} as inputs and produces t bits output, the signature of S_m's e-cash can be reconstructed with a probability no less than $1 - 2^{-t}$.*

Proof. Let $H()$ be a collision-resistant hash function with t bits output. Suppose a malicious host S_m launches a truncation attack by replacing its successor server S_{m+1} and its own data εo_m with S'_{m+1} and $\varepsilon o_m{}'$ where $(\varepsilon o_m, S_{m+1}) \neq (\varepsilon o_m{}', S'_{m+1})$, then $hd_m \neq hd'_m$ is satisfied with a probability no less than $1 - 2^{-t}$ where $hd_m = H(\varepsilon o_m, S_{m+1})$ and $hd'_m = H(\varepsilon o_m{}', S'_{m+1})$ as $H()$ is collision-resistant. So two different sets of shares $\{sm_{c_1}, sm_{c_2}, \ldots, sm_{c_t}\}$ and $\{sm_{c'_1}, sm_{c'_2}, \ldots, xm_{c'_t}\}$ determined by hd_m and hd'_m respectively are revealed and there exists $\{c_\alpha, c'_\alpha\} = \{\varepsilon_\alpha, o_\alpha\}$, $1 \leq \alpha \leq y$, also with a probability no less than $1 - 2^{-t}$. Therefore, S_m's private key $sm = sm_{\varepsilon_\alpha} + sm_{o_\alpha}$ can be reconstructed with the same probability. □

The protocol in Fig. 1 can effectively prevent the truncation attack, the stemming attack and some other attacks, as follows:

- *Sending incorrect shares.* In this attack, the malicious server S_j $(0 < j \leq n)$ sends unpublished shares to S_{j+1}. This attack can be easily identified by S_{j+1} in its *Verify* stage.
- *Truncation and Stemming attacks.* As we have discussed in the Sect. 2.1, the truncation and stemming attacks require one-time operations to be repeated. According to Theorem 1, a malicious server's e-cash will be forfeited with an overwhelming large probability. In addition, since S_j cannot tamper with the encapsulated offers of other servers, it cannot shift the responsibility of a truncation attack that it launches. All the encapsulated offers are digitally signed by the server's long term private key, of which the malicious server has no knowledge. Any illegal modification can be detected during the *verify* stage.

As we have discussed in Sect. 2.1, the malicious server that launches a stemming attack is difficult to identify. With the "e-division" technique, the malicious

server S_m can be discovered: (1) during the *verify* stage when S_{m+1} checks S_m's digital signature, if S_m deliberately inserted an offer but did not honestly sign it, or (2) during the *reveal* stage after the trusted third party publishes the received data, if S_m truncates a string of data and inserts fake offers under the victim servers' names.

A special case of the truncation and the stemming attacks occurs when a server S_j that has not made an offer deletes a number of consecutive offers $\{O_i, O_{i+1}, ...\}$ in the mobile agent and appends fake offers $\{O_j, O_{j+1}, ...\}$ instead. Since it has not sent its shares, there is no way to reconstruct the signature of its e-cash. However this attack is very easy to detect as its identity is not included in the offer O_{j-1} produced by S_{j-1}, unless it colludes with S_{j-1}. However this collusion will lead to the reconstruction of S_{j-1}'s e-cash signature, simply because S_{j-1} has to recompute O_{j-1} in order to include S_j in the encapsulated offer.

- *Modification, insertion and deletion attacks.* Following the same argument as for the truncation attack above, these three attacks that require illegally repeated operations will also be identified in the *unveil* stage. A special case of the modification attack occurs when a re-visited server replaces its previous committed offers. This also requires the server to re-calculate a new offer. Therefore it can be prevented with our new protocol.

Defense against other attacks is described in the extended paper.

The proposed scheme relies on the fact that each server will honestly utilise the hash-based algorithm to choose a certain set of its signature shares to publish. This can be checked at the "*verify*" stage when a verifier checks the correspondence between the commitments calculated from a server's chosen shares and the same server's published commitments, based on each bit of the hash value (see "*verify*" stage in Sect. 3.3).

Note this technique can be also applied in any circumstance where repeated operations are forbidden, and the choices of shares with each operation can be varied. For example, they can be determined by data, time or other characters of the operation.

Computation Efficiency.

Suppose each server chooses t sets of shares. The mechanism uses DSA for digital signatures with 160-bit SHA-1, as well as a 1024-bit RSA signature scheme. The computational cost of RSA signature scheme is one exponentiation for each generation and verification, and provides 128 bytes output. DSA needs two exponentiations with 40 bytes output.

Referring to Table 2, we count the average computational cost for each server in terms of the number of modular exponentiations required by DSA signatures as an example. To analyse the communication complexity, we count the number of bytes required to transmit the message from one server to another. The communication cost only occurs when all the participant servers send their commitments to the trusted third party and when the agent is dispatched from one

server to another. The maximum bandwidth is required when the agent is sent from S_{n-1} to S_n (assume the agent visits n servers).

Average Computational Cost		
Stages	Without "e-cash division" technique (exponentiations)	With "e-cash division" technique (exponentiations)
Setup	-	$1 \times 2t$
Offer Encryption	2	2
Choose and Share	-	-
Sign	1	1
Verify	n	$n + \frac{tn}{2}$
Update	-	$1 \times 2t$
Reveal	-	-

Maximum Communication Cost		
Stages	Without "e-cash division" technique	With "e-cash division" technique
Setup	-	$2t \times 128$bytes
Dispatch	$\sum_{i=0}^{n-1} \text{size}(\varepsilon o_i) + \sum_{i=0}^{n-1} \text{size}(S_i) + n \times 40\ bytes$	$\sum_{i=0}^{n-1} \text{size}(\varepsilon o_i) + \sum_{i=0}^{n-1} \text{size}(S_i) + n \times 40\ bytes + n \times t \times 128\ bytes$

Table 2. Comparison of computational and communication cost of the protocol

From the analysis above, some additional computation and communication cost can be observed. The computation complexity relies on the number of the shares chosen. The greater the number of shares chosen, the greater the computation cost and also the larger the payload of the agent as the shares have to be sent with the agent. However the system is more secure with a larger number of shares. Therefore there is a tradeoff between the computation complexity and the security requirements.

However, the extra cost does not impact a great deal in the performance of the protocol, since all the additional computation described above can be done off-line. Communication cost during the protocol run grows linearly on the number of the servers to be visited.

5 Conclusion

A number of published protocols [5][11] for data integrity in mobile agents have been vulnerable to truncation attacks where a sequence of data is deleted by ma-

licious hosts. In this paper, we proposed a robust defense against the truncation and stemming attacks for the purpose of better protection of the computation results of mobile agents. The proposed "e-cash division"technique attempts to detect and punish any illegally repeated operations of a malicious server. The proposed new protocol also effectively detects and prevents other attacks that require repeated operations. More extensible usage of the new protocol can be achieved by adjusting the choice of shares according to the nature of the operations in different applications.

However, in the proposed technique the sets of shares grow in size as the mobile agent travels. There is a tradeoff between the security and the computation complexity. Therefore future work will focuse on how to improve the performance of the proposed technique.

References

1. Brands, S.: Untraceable off-line cash in wallet with observers Proceedings of the 13th annual international cryptology conference on Advances in cryptology. Lecture Notes in Computer Science. Springer-Verlag, Berlin Heidelberg (1994), 302 – 318.
2. Chaum, D., Fiat, A., Naor, M.: Untraceable electronic cash. Proceedings on Advances in cryptology. Lecture Notes in Computer Science. Springer-Verlag, Berlin Heidelberg (1990), 319 – 327.
3. Cheng, Jeff S.L., Wei, Victor K.: Defenses against the Truncation of Computation Results of Free-Roaming Agents. In: Deng, R., Qing, S., Bao, F., Zhou, J. (eds.): Proceedings of the 4th International Conference (ICICS 2002). Information and Communications Security, Lecture Notes in Computer Science, Vol. 2513. Springer-Verlag, Berlin Heidelberg New York (2002) 1–12.
4. Ferguson, N.: Single term off-line coins. In Advances in Cryptology— EUROCRYPT '93. Lecture Notes in Computer Science, Vol. 765. Springer-Verlag, Berlin Heidelberg New York (1994) 318–328.
5. Karjoth, G., Asokan, N., Gülcü, C.: Protecting the Computation Results of Free-Roaming Agents. In: Rothermel, K., Hohl, F.. (eds.): Proceedings of the 2nd International Workshop on Mobile Agents (MA '98). Lecture Notes in Computer Science, Vol. 1477. Springer-Verlag, Berlin Heidelberg New York (1998) 195–207.
6. Loureiro, S., Molva, R., and Pannetrat, A.: Secure Data Collection with Updates. Electronic Commerce Research Journal, Vol. 1/2. Kluwer Academic Publishers (2001) 119–130.
7. Menezes, A., Oorschot, P. van, Vanstone, S.: Handbook of Applied Cryptography. CRC Press Inc. (1996)
8. Mu, Y., Varadharajan, V., Nguyen, K. Q.: Digital cash. Payment technologies for E-commerce. Springer-Verlag, Berlin Heidelberg (2003) 171–194.
9. Piccinelli, G., Stefanelli, C., Trastour, D.. : Trusted Mediation for E-service Provision in Electronic Marketplaces. Electronic Commerce : 2nd International Workshop, WELCOM 2001. Lecture Notes in Computer Science, Vol. 2232. Springer-Verlag, Berlin Heidelberg (2001) 39–50
10. Schoenmakers, L. A. M.: An efficient electronic payment system withstanding parallel attacks. Technical Report CS-R9522, CWI(1995).
11. Yao, M., Foo, E., Peng, K., Dawson, E.: An Improved Forward Integrity Protocol for Mobile Agents. Proceeding of the 4th International Workshop on Information Security Applications (WISA 2003). Springer-Verlag, (2003) 272–285

An Improved Execution Integrity Solution for Mobile Agents

Michelangelo Giansiracusa[1], Selwyn Russell[1], Andrew Clark[1], and John Hynd[2]

[1] Information Security Institute
[2] School of Software Engineering and Data Communications
Queensland University of Technology
Brisbane, Queensland, Australia
{m.giansiracusa,s.russell,a.clark,j.hynd}@qut.edu.au

Abstract. In this paper we introduce a new infrastructural approach to providing mobile agent execution integrity, a very important property - especially to the confidence an agent user can place in the results of its deployed autonomous mobile agents. Existing mobile agent execution integrity schemes are shown to be comparatively inferior when analysed in light of a number of essential robustness properties. We provide an analysis of Hohl's reference states scheme and introduce our novel execution integrity scheme, which builds - in part - on Hohl's scheme. Besides significantly improving on this existing scheme, our scheme meets all of the desired criteria for a robust execution integrity scheme.

Keywords: Mobile agent security, RECDAM, execution integrity, real-time execution checking, reference state, MASHIn, MASH, checking host.

1 Introduction

Since the early-to-mid 1990s, mobile agents have been marketed as the next big thing in distributed application technologies. Contrary to their great promise, their uptake has been limited mostly to single-domain Intranet environments only. Their use on the Internet, in multi-domain environments, has largely yet to migrate from the research community into practical real-world applications. Their potential as a technology to revolutionise the way we perform computations, develop applications and systems [1] persists, despite limited investment and usage from industry in mobile agent technology.

One frequent explanation why mobile agents face significant obstacles before they can be widely adopted is the inherent security risks they face and pose to others [2,3,4]. In fact, there are many angles and issues of concern pertaining to security with Internet mobile agent usage. We concentrate on execution integrity for mobile agents in this paper, and contribute to advancing the state-of-the-art in this area by presenting a new scheme for ensuring execution integrity of mobile agents - an important property to both major paradigm stakeholders (i.e. mobile agent users and mobile agent platform owners).

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 170–183, 2005.

1.1 Background

Whilst a plethora of definitions have been given for mobile agents, including but hardly limited to [1,5,6,7], we describe a mobile agent as: *A software coded abstraction of a number of tasks assigned on behalf of its user. The mobile agent is capable of autonomous migration to networked mobile agent platforms where it performs a subset of its work. The mobile agent's execution state is maintained as it hops between mobile agent platforms in its itinerary. A mobile agent's work at each agent platform provides partial results which are accumulated and analysed in achieving its high-level purpose - that is, the mobile agent's mission goal.*

In their purest form, mobile agents must be capable of performing their assigned tasks autonomously and securely. This means they must perform their mission without the need for premature or intermittent return to their user's home agent platform to assure their data and logic integrity or to complete a task securely. Whilst the property of autonomous behaviour is very powerful, it would seem to be limited due to a conflict of interest with traditional security values. From the agent user's perspective, sensitive data and code should not be disclosed to untrusted agent platforms. From an agent platform owner's perspective, flexible agent usage on their agent platform should only be permitted to trusted principals. Thus, with the two major paradigm stakeholders mistrusting each other, the pursuit for truly autonomous and secure mobile agents remains largely an unsolved problem.

The security threats, especially from malicious mobile agent platforms, to mobile agents have been widely reported [8,9,10]. The associated risks are real and often non-trivial to counter, so much so that they have limited investment in the adoption of mobile agents as a viable paradigm for global electronic services. While some countermeasures have been proposed [10,11,12], their lack of robustness is of some concern.

Mobile agent execution integrity is a very important property for both paradigm stakeholders. From the perspective of agent users, they want to be sure that their deployed agents return reliable, correct mission results. From the perspective of agent platform owners, they do not want to be accused of maliciously influencing the execution, or results, of an agent that has done some work on their agent platform in completing the agent's mission.

In this paper, we present a novel and robust execution integrity scheme which builds - in part - on Hohl's reference states [13,14] scheme. Our *Real-time Execution Checking and Deterrent Against Misbehaviour* (RECDAM) scheme is an infrastructural strategy which fits in appropriately with our broader *Mobile Agent Secure Hub Infrastructure* (MASHIn) vision of a secure community for paradigm stakeholders to interact. The MASHIn concept, presented elsewhere [15,16], offers a promising new macro-level approach to tackling the troubling security issues limiting Internet mobile agent prospects by strategically incorporating novel TTPs for the mobile agent paradigm [17]. The TTPs, particularly MASHs and checking hosts, play a pivotal role in reducing the lingering mistrust between the major paradigm stakeholders, thus promoting an interaction that was otherwise unlikely.

1.2 Outline of Paper

We start in Section 2 by discussing a number of important properties necessary in providing a robust mobile agent execution integrity solution. In Section 3 we present our new execution integrity solution, RECDAM, which builds on Hohl's reference states scheme [13,14] - but base it in a macro-level context, specifically the MASHIn. In Section 4, an analytical comparison is provided - using the robustness criteria introduced in Section 2 - of existing schemes with our novel RECDAM scheme. The paper's base conclusive observations are given in Section 5, as well as avenues for future work.

2 Execution Integrity Robustness Properties

We define a competent mobile agent *execution integrity* scheme as not only providing comprehensive protection for the mobile agent's data and code integrity, but also irrefutably linking the agent user and agent platforms to their inputs into any interaction via the agent's execution.

Furthermore, an execution integrity scheme for mobile agents is ineffective without a readily accommodating means of recovering from a hijacked agent session on an agent platform. In more traditional distributed systems, the distinction between the desired properties of *integrity* per se and *availability* per se is more clear. However, in mobile agent systems the "playing field" for mobile agents is a series of agent platforms. There must be a way to irrefutably link both the agent and agent platform inputs into an interaction at least up until a point-of-failure in the agent's itinerary. For example, if there is a significant or potentially fatal disruption to the agent's mission at or around the fifth agent platform in an agent's itinerary, the results and interactions of the agent at agent platforms 1-4 inclusive must not be lost. Surprisingly, this aspect is not accommodated in many existing execution integrity or denial-of-service schemes for mobile agents, with drastic consequences likely since there is no accountability up until the point-of-failure, meaning that these interactions are left in a state of jeopardy (for example, should they be rolled-back or agent platform results/service remuneration somehow fairly discarded/handled?) and the responsibility for the failure cannot be attributed to an entity (was the mobile agent responsible, or was the agent platform responsible?). Besides not losing these previous work-flows, the scheme should also provide a means around the point of disruption in the mobile agent's itinerary.

Now we elaborate on the specific qualities that we view as essential in a comprehensive execution integrity package for the Internet mobile agent paradigm:

- *Application-Independent*: This one should be quite clear; if the execution integrity scheme is not mobile agent application-independent, then only a subset of agent applications can benefit, limiting the usefulness of the scheme and further hampering the interoperability and practicality of safe methods for enabling ubiquitous Internet mobile agents.

- *Dynamic Itinerary Changes Possible*: The execution integrity scheme must be capable of handling dynamic changes to an agent's itinerary, otherwise all agents would be confined to pre-defined statically-defined mission itineraries - limiting the autonomous benefits of mobile agents. Feasible dynamic changes that should be supported include appending and truncating agent platforms in a pre-defined agent itinerary, and inserting and removing agent platforms during an agent's route of a pre-defined agent itinerary.
- *Real-Time In-Mission Checking Possible*: The execution integrity scheme must be capable of detecting breaches of integrity in an agent and claimed stakeholder inputs in "real-time in-mission", i.e. as the agent is in the midst of its mission, before and after the visit to an agent platform. Otherwise, once again, the damage done can leave agent platforms in an ambiguous state (how to appropriately recover from a post-mission detected breach of execution integrity?) with regard to issues of rollback and remuneration for services delivered. If real-time in-mission checking is supported, the damages from a breach of execution integrity can be localised, and the responsible entity more easily identified.
- *Unconditional Autonomous Checking Post-Mission Possible*: Regardless of whether real-time in-mission checking is possible, in some instances post-mission checking may be necessary; for (example) reasons of efficiency or legal enactment, it may be necessary to only perform execution checks post-mission, or in addition to real-time in-mission checking. A high quality execution checking scheme for mobile agents must be capable of performing post-mission autonomous execution checks. By autonomous checking, we mean the scheme should not rely on the agent user (or their home platform) to perform the execution checking, and the checking should not be triggered only on the suspicion of the agent user.
- *Resistant Against Stakeholder Denial-of-Service*: As mentioned in the introductory discussion to this list, for the reasons given there, a reliable mobile agent execution integrity scheme (and, thus, the mobile agent's mission results prior to, and in future of, the disruption) should not be rendered useless if a stakeholder either deliberately or unintentionally causes a denial-of-service to the mobile agent.
- *Agent Platform Collaboration Attack Resistant*: An execution integrity scheme must not be susceptible to collaboration attacks[3] from agent platforms breaking the execution integrity of a mobile agent.
- *Real Punishment Deterrent*: Catching breaches to execution integrity from either stakeholder is only part of the battle. In fact, if there is not an automated system mechanism to punish stakeholders for their discrepant behaviour, then the perpetrators may feel confident in continuing with their reckless/malicious behaviours. Ideally, the punishment should not just be a post-malicious-reactionary event for breaches but should also act as a sig-

[3] Some agent platform collaboration attack threats against existing execution integrity mechanisms are discussed in [4,13,18].

nificant real deterrent against stakeholders breaching execution integrity in the first place.
- *Effective*: The execution integrity scheme must be easy to manage and transparent to incorporate, and not simply be a fantastic theoretical proposal which has limited/non-robust utility in generic mobile agent applications.
- *Efficient*: Given all of the above features are present, the execution integrity scheme should not place unreasonable pressures - neither on an agent platform, nor in terms of network messages - in meeting its objectives.

Corresponding to all protective measures for mobile agents, not just execution integrity, is the important principle: Autonomous (of agent user and home agent platform) mobile agent execution integrity checking is important because the agent user may have disconnected from the network (i.e. their mobile device may not be in a hot spot, or it may be running short of battery-life) or be unable to perform the necessary cryptographic processing (due to hardware and/or software limitations). From the perspective of agent users, they simply want a reliable summary of their agents' security-ensured mission results. These mission results could be securely communicated to the agent user in a variety of ways (e.g. s/mime email, secured SMS mobile phone message, certified or discreetly packaged postal mail) with the important fact that the agent user may not even have a home agent platform processing environment (e.g. an end-user mobile agent may be launched from completing a web page form).

In the next section (i.e. Section 3) we provide the reader with an introduction to our new execution integrity solution, RECDAM[4]. In Section 4 we compare RECDAM with existing execution integrity schemes using the robustness criteria detailed in this section.

3 New Execution Integrity Solution

The reasons we chose to build on Hohl's reference states scheme and the limitations identified in that scheme are detailed in Section 3.1; and the adapted reference states protocols underpinning our macro-level fitted RECDAM scheme are presented in Section 3.2.

3.1 Analytical Review of Hohl's Reference States Scheme

None of the reviewed existing execution schemes (see [18]) are sufficiently robust in satisfying our requirements for a solid mobile agent execution integrity scheme, but we identify Hohl's reference states (RS) [13,14] execution integrity approach as particularly admirable because:

- The execution integrity checks are performed *autonomously* in *real-time*, and
- The scheme aims to non-refutably tie stakeholder inputs with a mobile agent's execution session(s).

[4] Due to publication size limits we were unable to provide a thorough description of neither Hohl's underlying reference states scheme nor a complete annotation of RECDAM protocols and RECDAM extensions; these can both be found in [18].

However, we see a number of limitations with the RS scheme. These limitations include (listed in no particular order of weighting):

1. Agent platform collaboration attacks are possible, particularly two or more consecutive agent platforms in an agent's itinerary maliciously corroborating to cover breaches of execution integrity (including malevolent agent platform inputs and improper interpretation of the agent's code).
2. Agent platform inputs cannot be kept secret from checking agent platforms, due to the protocol's requirement that these inputs be forwarded for execution checking on the next agent platform. However, agent platforms in the agent's itinerary may be competitors, or at least not trusting the other's sincerity or practices.
3. There is no robustness in terms of preventing denial-of-service attacks from agent platforms, who may refuse to service agents or not abide by the reference states protocol.
4. There is no possibility for agent user or itinerary anonymity, since stakeholders must digitally sign their inputs according to the protocol: The agent user signs its agent; agent platforms, on behalf of their owners, sign resulting calculated states for agents and their inputs are encapsulated in signed messages.
5. Furthermore to the previous limitation, the scheme necessitates that agent platforms know the agent user trust statements in agent platforms. Both may be viewed as breaches of privacy, and could result in mining agent user preferences and an aggressive malicious campaign by agent platforms against (agent user) preferred competitors.
6. The protocol is overly complex, for example requiring agent platforms to sign the resulting state element and then sign the encapsulating whole message as well. In addition, it is not clear why the agent code is not transferred with the signed message - how is it protected, and how does an agent platform link agent code to its pertinent reference state message?
7. There is only one mode for the reference states protocol - perform the execution checks as a pre-condition that must be met before executing the agent on the agent platform. This will increase an agent platform's processing load significantly (i.e. double it on average), and the agent user may - due to mission urgency reasons - not always wish for the agent's execution checks to be performed until after the agent has travelled to all agent platforms.
8. The robustness of long-term stakeholder non-repudiation is questionable. For example, how are the reference states maintained? Does each agent platform append to one long, growing reference state message list that is returned to the agent user along with the agent at the end of the agent's mission? If any agent platform in the agent's itinerary destroys the agent reference state, previous irrefutable agent platform statements are broken unless each agent platform maintains a log of all its transported reference state messages. Checking would then be instigated spasmodically by an agent user, triggered only on their suspicion that something might have gone wrong - breaking the definition of autonomous, asynchronous execution for mobile agents.

Despite these shortcomings with Hohl's RS scheme, we build and improve on his scheme for two main reasons. Firstly, we favour the scheme's very admirable qualities of real-time in-mission execution integrity checking and performing these security checks completely autonomous of the agent's user and their home agent platform. And, secondly, we can solve a number of the shortcomings associated with the RS scheme by appropriately adapting it to work within our broader MASHIn vision [15,16,18] of a secure Internet framework for mobile agents promoting stakeholder interaction; at the same time increasing the scheme's effectiveness.

3.2 Real-Time Execution Checking and Deterrent Against Misbehaviour

Three protocols comprise Hohl's RS scheme [13], but Hohl's scheme is not set in a macro-level context (with equitable terms for both major stakeholders). In the MASHIn, MASHs and checking hosts are introduced as neutral and authoritative TTPs [18]. RECDAM is also comprised of three protocols, but now related to these introduced TTPs. In terms of protocol overheads, the changes are not as costly as one would expect.

We must point out that all messages sent in the following protocols are also protected for confidentiality (as well as integrity and sender authentication) using asymmetric encryption, by enveloping the messages for the intended recipient. This step was also not explicitly included in Hohl's original RS protocols [13] as this is standard, accepted safe practice when considering protocols related to the migration of agents [19].

In Figure 1 the RECDAM protocol for a MASH (the starting itinerary location for an agent in the MASHIn) is presented. MASHs have by this stage inserted a number of checking hosts in between agent platforms in an agent's itinerary [18]. The checking hosts, in cooperation with the pertinent MASH and the other selected checking hosts, enable - among many responsibilities[5] - real-time in-mission execution checking and support for many of the other desired properties of a robust execution integrity solution described in Section 2.

In Figure 2 the RECDAM protocol for an agent platform is presented. Contrary to Hohl's RS scheme [13], RECDAM does not differentiate between agent platforms trusted or untrusted by an agent user. This is because RECDAM is macro-level, targeting without bias the interests of *both* major stakeholders. Agent platform inputs, for example, can be protected for privacy from other agent platforms in RECDAM - something not possible in Hohl's RS scheme. The indirection of the introduced neutral checking hosts is pivotal in keeping both stakeholder groups accountable for their inputs - regardless of individual stakeholder trust relationships which may not be mutual (between agent user and agent platform owner) and/or may not be transitive (between an agent

[5] Callback classes, for example, are part of a MASHIn facility enabling high quality mobile agent privacy, and could also support flexible real-time handling of execution integrity checking feedback [16,18].

MASH.1 Compute $state_2$
MASH.2 Add $state_2$ to message for first checking host
MASH.3 Add $state_2$ to message for first agent platform
MASH.4 Add agent code (including 'real' callback classes) to message for first checking host
MASH.5 Add agent code (*not* including 'real' callback classes) to message for first agent platform
MASH.6 Sign message intended for first checking host and send it
MASH.7 Sign message intended for first agent platform and send it

Fig. 1. RECDAM protocol for the MASH (i.e. an agent's "home" agent platform in the MASHIn).

AP.1 Get message from previous checking host and check signature
AP.2 If not true, COMPLAIN and STOP
AP.3 Get $state_{i+1}$ from message
AP.4 Compute $state_{i+2}$ from $state_{i+1}$ using $input_{i+1}$
AP.5 Add agent code to message
AP.6 Add $input_{i+1}$, $state_{i+2}$ to message
AP.7 Sign message and send it to the next checking host

Fig. 2. RECDAM protocol for agent platforms.

CH.1 Get message from previous checking host or the MASH, and check signature
CH.2 If not true, COMPLAIN and STOP
CH.3 Get message from previous agent platform and check signature
CH.4 If not true, COMPLAIN and STOP
CH.5 Get $state_{i+1}$ from previous agent platform message
CH.6 Get $input_i$ from previous agent platform message
CH.7 Get $state_i$ from previous checking host/MASH message
CH.8 Compute $state_{i+1}$ from $state_i$ using $input_i$
CH.9 Check if $state_{i+1}$ from line CH.8 and $state_{i+1}$ received from previous agent platform differ
CH.10 If true, COMPLAIN and STOP
CH.11 Add $state_{i+1}$ to message for next checking host
CH.12 Add $state_{i+1}$ to message for next agent platform
CH.13 Add agent code (including 'real' callback classes) to message for next checking host
CH.14 Add agent code (*not* including 'real' callback classes) to message for next agent platform
CH.15 Sign message intended for next checking host and send it
CH.16 Sign message intended for next agent platform and send it

Fig. 3. RECDAM protocol for checking hosts.

user/agent platform owner and subsequent agent platform owner in the agent's itinerary).

In Figure 3 the RECDAM protocol for a checking host is presented. In some respects it is quite similar to the agent platform protocols in Hohl's RS scheme, because it is the checking hosts in the MASHIn RECDAM scheme which perform the actual execution integrity checks (not agent platforms). Steps CH.6-CH.10 are the actual execution checking steps. Firstly, the checking host must extract the claimed inputs from the previous agent platform's execution session with the agent. The initial state of the agent on the previous agent platform is then extracted from the previous checking host's message (step CH.7). This makes it more secure (against agent platform collaboration attacks, for example) then the case in Hohl's original scheme whereby only agent platforms were making these initial agent state claims. Step CH.8 is the re-execution of the agent execution session on the previous agent platform. If the re-executed resulting state differs from the previous agent platform's purported resulting state the discrepancy is noted to the appropriate authorities (this would be the MASH in the first instance), and the protocol processing is ceased.

If however the execution integrity check was verified, appropriate messages are constructed and sent to the agent's next checking host and agent platform in steps CH.11-CH.16 inclusive (these steps are logically similar to the last six steps in the RECDAM MASH protocol in Figure 1). Note that these last six steps are *not* processed if this "checking host" is the MASH (i.e. the agent is back at the MASH for the final time after visiting the last agent platform in its itinerary, and the MASH is simply acting in the role of a checking host at this final point in the agent's itinerary). There is one caveat to be aware of in step CH.11 and CH.12. The agent state to be transported to the next checking host and agent platform is in fact the agent state after the secure 'real' callback class/es (privy only to the MASH and checking hosts) have been applied; the resulting agent state may or may not be the same. The 'real' agent state component in the 'real' callback classes is forwarded to the next checking host only (i.e. not the next agent platform) - encapsulated in step CH.13.

It is not possible to give detailed comments on the RECDAM protocols, and related design decisions in the available space; the interested reader can find this level of detail in [18]. In addition to a lot more detailed explanations, there is also discussion on supporting other modes of operations for RECDAM and a fee-charge structure for remunerating MASHIn TTPs is postulated.

4 RECDAM Comparison with Existing Schemes

Table 1 compares the capabilities of existing execution integrity strategies for mobile agents with our RECDAM strategy capabilities. Our RECDAM strategy satisfies all of the desired execution integrity objectives (defined and discussed in Section 2) for a robust execution integrity strategy, whilst the existing strategies did not fare nearly as well - notably, none of them bettering support for more than four of the seven desired ("present"/"non-present") properties.

The Partial Result Encapsulation (PRE), Execution Tracing (ET), and Reference States (RS) approaches are mobile agent application-independent schemes, meaning they can be reasonably applied to agents of arbitrary application-type and code size. The direct construction of Holographic Proofs (HP) for arbitrary agents is not feasible at this time.

Dynamic itinerary changes are possible in all four reviewed schemes, with the caveat that it is not possible in all implementation varieties of the PRE concept.

Real-time in-mission checking is supported fully only in the RS scheme. However, in some implementation varieties of PRE it is supported, and in the extended ET scheme by Tan and Moreau [20] it is directly supported.

Unconditional post-mission autonomous checking is not supported by any of the schemes. The first three schemes rely on the agent user (or their agent platform) to conduct post mission checks, thus breaking the autonomous property we desire. The RS scheme does not support post-mission checking directly, and it would only be possible to extend the scheme by introducing a TTP receiving reference states messages (otherwise the agent user/home platform would have to perform the checks, thus breaking the autonomous checking property).

None of the schemes, except Tan and Moreau's extended ET [20] scheme, provide resistance against denial-of-service or denial-of-execution by stakeholders preventing correct interpretation of an agent on an agent platform (session). Moreover, the results from agent platform sessions in the agent's itinerary preceding the point-of-failure are often lost or left in a state of inconsistency.

All of the schemes are directly vulnerable to agent platform collaboration attacks, and none of them provide any automated system punishment mechanism for deterring future attacks.

The PRE scheme is only medium effective because its implementations often rely on the agent user/home platform to perform the execution integrity checks. Cryptographic key generation and/or storage management issues and agent platform collaboration attacks are also limiting factors. Nevertheless, the PRE scheme scores admirably in terms of efficiency.

The HP scheme rates poorly in terms of both effectiveness and efficiency, and cannot be seen to be a viable solution to execution integrity of arbitrary mobile agents in the near future.

The ET scheme is only medium-effective because the cryptographic traces to generate and manage would be large for both sets of stakeholders. However, the extended ET scheme lessens this load, but introduces its own set of problems (particularly, vulnerability to malicious collaboration between an agent platform and verification server). The number and size of the cryptographic messages are also a concern to the scheme's efficiency.

The RS scheme is limited in its design, leaving it - for example - highly vulnerable to agent platform collaboration attacks. However, the scheme's efficiency is admirable, and the RS property of real-time in-mission checking is very admirable.

Scheme	AI	DIP	RTC	UPMAC	SDoSR	APCAR	RPD	Effectiveness	Efficiency
Partial Result Encapsulation (PRE)	√	%	%					Medium; often relies on agent user/home platform to perform checks	Medium-Good
Holographic Proofs (HP)		√						Impractical now, and likely to remain so	Poor
Execution Tracing (ET)	√	√	+		+			Medium; but the cryptographic trace management is problematic	Poor-Medium
Reference States (RS)	√	√	√					Poor-medium, because of the collaboration attack and input privacy problems	Medium-Good
RECDAM	√	√	√	√	√	√	√	Very-High, more secure and reliable than reference states	Low-Medium, but very bearable given the high protection afforded by the scheme

Table 1: A comparison of existing mechanisms for mobile agent execution integrity with our RECDAM scheme.

Table Abbreviations: AI = Application-Independent; DIP = Dynamic Itinerary Possible; RTC = Real-Time Checking; UPMAC = Unconditional Post-Mission Autonomous Checking; SDoSR = Stakeholder Denial-of-Service Resistant; APCAR = Agent Platform Collaboration Attack Resistant; RPD = Real Punishment Deterrent; RECDAM = Real-time Execution Checking and Deterrent Against Malicious Behaviour (our mechanism).
Table Symbol Key: " √ " = feature supported; " + " additional supported feature in the extended execution tracing method; " % " = supported in some forms of scheme.

In analysing the robustness properties with respect to the RECDAM strategy we see:

- The RECDAM strategy is an execution integrity approach supporting arbitrary mobile agents, meaning it can support *application-independent* mobile agents and those agents are not limited by execution code size.
- In the RECDAM scheme, mobile agent *dynamic itinerary changes* are facilitated via mobile agent secure 'real' callback classes (specifically the event-response methods encapsulated therein) being retained on trusted checking hosts as discussed in [16,18]. Travel to additional agent platforms may also be negotiated autonomously when the agent returns to the MASH; importantly the agent execution sessions on these additional agent platforms are also protected under the RECDAM process because the MASH is capable of coordinating checking hosts for protecting this additional travel and agent execution.
- *Real-time in-mission agent checking* is supported on the checking hosts, via an adaptation of Hohl's reference states scheme. However, as pointed out in [18], the RECDAM scheme is significantly more secure than Hohl's original scheme.
- In [18], we also discussed three modes that could conceivably be supported for RECDAM execution checking of mobile agents. One of those modes was *unconditional autonomous checking post-mission*, but (from a short and long-term security perspective) the real-time in-mission mode for which we supplied proposed protocols for in Section 3.2 is the preferred mode.
- The RECDAM approach is both *resistant against major stakeholder denial-of-service* and *resistant against agent platform collaboration attacks* as discussed in [18].
- Finally, as elaborated on in [18], the MASHIn RECDAM offers a *real punishment deterrent* for breaches of execution integrity by the possibility of lowered stakeholder reputation/confidence scores against an offending agent platform.

To a degree, there is an increased cost in the MASHIn RECDAM scheme in terms of *efficiency* because agents, and their execution integrity specifically, are checked on checking hosts. This transport cost (agent travel to/from a checking host) is very bearable, though, considering the increased security benefits over existing schemes. One prominent example of this increased security robustness is the major improvements seen in the RECDAM scheme over Hohl's reference states scheme, which we appropriately adapted and incorporated in our scheme.

Provided the infrastructural TTP components are present, our RECDAM scheme is very *effective* and highly admirable since it does not favour protection of one stakeholder over the other. Security for both major mobile agent stakeholders is catered for in the MASHIn, which works to lower the barriers of mistrust so that the two major stakeholders can more safely form and participate in working relationships.

5 Conclusions and Future Work

The early promise of great things coming from ubiquitous utilisation of Internet mobile agent technology may be a long way off, especially if serious security problems associated with the paradigm are not adequately addressed.

This paper served to offer two main contributions: (1) The establishment of new criteria by which mobile agent execution integrity schemes can be critically analysed; following from this, it was shown that the existing state-of-the-art approaches to tackling the issue of mobile agent execution integrity are largely non-robust. (2) An introduction to our novel *Real-time Execution Checking and Deterrent Against Misbehaviour* (RECDAM) execution integrity scheme. Many of the detailed design consideration and extension possibilities pertaining to RECDAM could not be encapsulated as hoped in this publication, but are included in [18]. RECDAM fits appropriately within the MASHIn framework and objectives, strategically utilising novel TTPs (specifically MASHs and checking hosts) to reduce the concerns of both major mobile agent paradigm stakeholders with regards to the execution integrity of mobile agents. The RECDAM scheme overcomes a large number of limitations with Hohl's reference states approach, and meets all of the properties we would expect from a robust execution integrity scheme for mobile agents. A number of RECDAM properties are of special value: the significant lowering of risk in agent platform collaboration attacks, mitigation against some forms of denial-of-service attacks from agent platforms and workaround capabilities, no reliance on agent user or their home agent platform for execution checks (thus the scheme is fully agent autonomous), and there is a real deterrent against malicious breaches of execution integrity.

New execution integrity protocols could be supported (as part of a suite available) in the MASHIn framework, to complement the underlying reference states processing model in RECDAM - which may not be sufficiently efficient in some mobile agent applications. One such case is where large quantities of input data need to be transported for checking purposes.

References

1. Samaras, G.: Mobile Agents: What about Them? Did They Deliver what They Promised? Are They Here to Stay? In: International Conference on Mobile Data Management (MDM'04). (2004) 294–295
2. Roth, V.: Obstacles to the adoption of mobile agents. In: International Conference on Mobile Data Management (MDM'04). (2004) 296–297
3. Wilhelm, U.: A Technical Approach to Privacy based on Mobile Agents Protected by Tamper-resistant Hardware. PhD thesis, Ecole Polytechnique Federale de Lausanne, Switzerland (1999)
4. Yao, M.: A Security Architecture for Protecting Dynamic Components of Mobile Agents. PhD thesis, Information Security Research Centre, Faculty of Information Technology, Queensland University of Technology (2004)
5. Gray, R.: Agent Tcl: A flexible and secure mobile-agent system. PhD thesis, Dartmouth College, Hanover, New Hampshire (1997)

6. Karnik, N.: Security in Mobile Agent Systems. PhD dissertation, University of Minnesota (1998)
7. Weiss, G., ed.: Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. MIT Press (2000)
8. Farmer, W.M., Guttman, J.D., Swarup, V.: Security for Mobile Agents: Issues and Requirements (1996) Presented at the 1996 National Information Systems Security Conference, Baltimore, MD, USA.
http://csrc.nist.gov/nissc/1996/papers/ NISSC96/paper033/SWARUP96.PDF.
9. Hohl, F.: Time Limited Blackbox Security: Protecting Mobile Agents From Malicious Hosts. Lecture Notes in Computer Science **1419** (1998) 92–113
10. Jansen, W.: Countermeasures for Mobile Agent Security. Computer Communications **Special Issue on Advanced Security Techniques for Network Protection** (2000)
11. Alfalayleh, M., Brankovic, L.: An Overview of Security Issues and Techniques in Mobile Agents. In: Eighth IFIP TC-6 TC-11 Conference on Communications and Multimedia Security. (2004) 59–78 Available at
http://sec.isi.salford.ac.uk/ cms2004/Program/index.html .
12. Claessens, J., Preneel, B., Vandewalle, J.: (How) can mobile agents do secure electronic transactions on untrusted hosts? - A survey of the security issues and current solutions. ACM Transactions on Internet Technology (TOIT) **3** (2003) 28–48
13. Hohl, F.: A Protocol to Detect Malicious Hosts Attacks by Using Reference States. Technical report, Universit Stuttgart, Fakult Informatik (1999)
14. Hohl, F.: A framework to protect mobile agents by using reference states. In: International Conference on Distributed Computing Systems. (2000) 410–417
15. Giansiracusa, M., Russell, S., Clark, A., Roth, V.: Macro-level Attention to Mobile Agent Security: Introducing the Mobile Agent Secure Hub Infrastructure Concept. In: Sixth International Conference on Information and Communications Security (ICICS'04). Volume 3269 of Lecture Notes in Computer Science., Springer-Verlag (2004) 343–357
16. Giansiracusa, M., Russell, S., Clark, A., Hynd, J.: A Step Closer to a Secure Internet Mobile Agent Community. In: Fifth Asia-Pacific Industrial Engineering and Management Systems Conference (APIEMS'04). (2004) Published on CD-Rom, ISBN: 0-9596291-8-1.
17. Giansiracusa, M., Russell, S., Clark, A.: Clever Use of Trusted Third Parties for Mobile Agent Security. In: Applied Cryptography and Network Security - Technical Track, ICISA Press (2004) 398–407
18. Giansiracusa, M.: A Secure Infrastructural Strategy for Safe Autonomous Mobile Agents. PhD thesis (2005) Passed internal examination, submitted for external examination - available at
http://everest.fit.qut.edu.au/ ˜n1402773/MicPhDThesis-11April2005.pdf.
19. Biehl, I., Meyer, B., Wetzel, S.: Ensuring the Integrity of Agent-Based Computations by Short Proofs. In Rothermel, K., Hohl, F., eds.: Proceedings of the Second International Workshop, MA '98. Volume 1477 of Lecture Notes in Computer Science., Springer-Verlag, Germany (1998) 183–194
20. Tan, H.K., Moreau, L.: Extending Execution Tracing for Mobile Code Security. In Fischer, K., Hutter, D., eds.: Second International Workshop on Security of Mobile MultiAgent Systems (SEMAS'2002). (2002) 51–59

RFID Guardian: A Battery-Powered Mobile Device for RFID Privacy Management

Melanie R. Rieback, Bruno Crispo, and Andrew S. Tanenbaum

Department of Computer Science, Vrije Universiteit, Amsterdam, The Netherlands
{melanie,crispo,ast}@cs.vu.nl

Abstract. RFID tags are tiny, inexpensive, inductively powered computers that are going to replace bar codes on many products, but which have many other uses as well. For example, they will allow smart washing machines to check for incompatible clothes (e.g., white shirts and red socks) and smart refrigerators to check for milk that is too old to be consumed. Subdermal tags with medical information are already being implanted in animals and people. However, a world in which practically everything is tagged and can be read at a modest distance by anyone who wants to buy an RFID reader introduces serious security and privacy issues. For example, women walking down the street may be effectively broadcasting the sizes of their RFID-tagged bras and medical data without realizing it. To protect people in this environment, we propose developing a compact, portable, electronic device called an RFID Guardian, which people can carry with them. In the future, it could be integrated into PDAs or cell phones. The RFID Guardian looks for, records, and displays all RFID tags and scans in the vicinity, manages RFID keys, authenticates nearby RFID readers, and blocks attempted accesses to the user's RFID tags from unauthorized readers. In this way, people can find out what RFID activity is occuring around them and take corrective action if need be.

1 Introduction

Nancy buys a sweater from her favorite department store. This store is her favorite because it has one of those new-fangled checkouts, which automatically tallies up her items and charges the total cost to her credit card. Nancy is not sure exactly how this system works, but she knows that a radio tag attached to the clothing supplies information to the store's computer system. But far more interestingly, this tag can also send instructions to her washing machine at home, which sets the length and temperature of wash cycles, and warns her whenever dark and light clothing are mixed in a single batch of laundry. The store offers a kiosk to disable the tags, but Nancy has never used it. Despite hearing news reports about targeted thefts and stalking, enabled by covert reading of RFID tags, she still does not understand why anyone would want to disable such useful functionality.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 184–194, 2005.

This scenario illustrates a typical use of Radio Frequency Identification (RFID), a popular identification and automation technology with serious unaddressed security and privacy threats. Inductively-powered RFID chips transmit information via radio waves, removing the need for a clear line of sight. These passive tags are powered by their reading devices, eliminating the need for batteries (and their periodic replacement). This quality makes RFID tags useful for a variety of applications. But this usefulness comes at a cost; RFID introduces security and privacy threats that range from unauthorized data access, to snooping on tag-reader communications, to location tracking of physical objects and people. Tag deactivation has been suggested as a way to combat these threats. But a dead tag cannot speak (not even to the washing machine), and this loss of functionality is not always desired by consumers. So other methods of consumer RFID security and privacy protection are needed. Several on-tag security primitives have been proposed, like sleep/wake modes, hash locks, pseudonyms, blocker tags, on-tag cryptography, and tag-reader authentication. The problem is that many of these techniques are not implementable on low-cost Electronic Product Code (EPC) style tags (like that used in Nancy's sweater). Existing techniques also do not yet work cooperatively – they manage the security of individual RFID tags, as opposed to managing the privacy of consumers like Nancy. In the future, existing primitives must be combined to offer a holistic solution for protecting people in an RFID-enabled world.

In this paper, we suggest a new approach for personal security and privacy management called the RFID Guardian. The RFID Guardian is a compact battery-powered device, integratable into Personal Digital Assistants (PDAs) or cellphones, that people carry with them to manage their security and privacy in an RFID-tagged world. The RFID Guardian leverages in-band RFID communications to integrate four previously separate security properties into a single device: auditing, key management, access control, and authentication. This offers some functionality that is totally new within the realm of RFID, and adapts some existing functionality to work in new application scenarios and new combinations.

2 Radio Frequency Identification

Radio Frequency Identification (RFID) is the latest development in the decades-old trend of the miniaturization of computers. Passive RFID transponders are tiny resource-limited computers that are inductively powered by the energy of the request signal sent from RFID readers. Once the RFID tag receives enough energy to "power up" its internal electronics, the tag can decode the incoming query and produce an appropriate response by modulating the request signal using one or more subcarrier frequencies. These RFID tags can do a limited amount of processing, and have a small amount (<1024 bits) of storage. Semi-passive and active RFID tags require a battery for their operation, and have accordingly more functionality. However, battery-powered RFID chips present fewer security and privacy challenges than passive ones, so we will focus upon passive RFID throughout the rest of this paper.

RFID tags have become the darling of automation specialists and venture capitalists, due to their battery-free operation. This has led RFID to be used in a variety of applications, including supply chain management, automated payment, physical access control, counterfeit prevention, and smart homes and offices. RFID tags have also been integrated into an ever increasing number of personal and consumer goods including cars, passports, frozen dinners, ski-lift passes, clothing, public transportation tickets, casino chips, and medical school cadavers. Implantable RFID tags for animals allow concerned owners to label their dogs, fish, and livestock. In a logical but controversial next step, RFID has even been used for tagging people. RFID-based monitoring of school children is gaining popularity, amidst a cloud of debate. Trials have already initiated the RFID-tagging of school children in locations as diverse as Japan, India, and California. Even more surprisingly, hundreds of club-goers in three major European cities have voluntarily implanted themselves with RFID chips, about the size of a grain of rice, to pay their bar tabs and gain access to VIP areas. [1] Researchers speculate that these implantable RFID chips could also someday have medical applications.

2.1 Threat Model

Despite the utility of RFID automation, not everyone is happy with the proliferation of RFID tags. Privacy activists warn that pervasive RFID technology might bring unintended social consequences, much in the same way as the automobile and the television. As people start to rely on RFID technology, it will become easy to infer information about their behavior and personal tastes, by observing their use of the technology. To make matters worse, RFID transponders are also too computationally limited to support traditional security and privacy enhancing technologies. This lack of information regulation between RFID tags and RFID readers may lead to undesirable situations. One such situation is unauthorized data collection, where attackers gather illicit information by either actively issuing queries to tags or passively eavesdropping on existing tag-reader communications. So the next time that Nancy purchases an RFID-tagged bra from the department store, she may have no way of controlling which strangers with an RFID reader can read the brand and size information from the RFID tag. Other attacks include the unwanted location tracking of people and objects (by correlating RFID tag "sightings" from different RFID readers), and RFID tag traffic analysis (e.g. terrorist operatives could build a landmine that explodes upon detecting the presence of any RFID tag).

A growing number of countermeasures to these RFID security and privacy threats have been suggested, which fall into different categories: permanent tag deactivation (tag removal, destruction, or SW-initiated tag "killing"), temporary tag deactivation (Faraday cages, sleep/wake modes), on-tag cryptographic primitives (stream ciphers, reduced AES, reduced NTRU), on-tag access control

[1] Some Christian fundamentalists see these implantable RFID chips as a warning sign of the apocalypse.

(hash locks, pseudonyms), off-tag access control (blocker tags), and tag-reader authentication (lighweight protocols, adapted air interfaces). Unfortunately, this rich variety of solutions still faces a number of problems. Current on-tag cryptographic, access control, and authentication proposals require high-end RFID tags for their implementation, leaving the application scenarios that require the cheapest and simplest RFID tags unprotected (e.g. supply chain management). Access control and authentication policies are also commonly distributed across many individual RFID tags, hindering the policy updates that are necessary to protect personal security and privacy in dynamic real-world situations. Some countermeasures are also difficult to use together (e.g. blocker tags cannot provide access control for tags using pseudonyms or hash locks)[7]. This lack of integration is unfortunate because different RFID security and privacy proposals have complimentary strengths and weaknesses, that could be leveraged by using a centralized platform to tie these mechanisms together.

3 RFID Guardian

The RFID Guardian is a platform that offers centralized RFID security and privacy management for individual people. The idea is that consumers who want to enjoy the benefits of RFID-tagging, while still protecting their privacy, can carry a battery-powered mobile device that monitors and regulates their RFID usage.

The RFID Guardian is meant for *personal use*; it manages the RFID tags within physical proximity of a person (as opposed to managing RFID tags owned by the person, that are left at home). For this reason, the operating range of the RFID Guardian must extend at least from the head to toe of the user; a radius of 1-2 meters should be sufficient. This full-body coverage requires the RFID Guardian to be *portable*. It should be PDA-sized, or better yet, could be integrated into a handheld computer or cellphone. The RFID Guardian could then occupy a vacant shirt pocket, handbag, or belt loop, and thus remain close to the person that it is supposed to protect. The RFID Guardian is also *battery powered*. This is necessary to perform resource-intensive security protocols, such as authentication and access control, which would not be possible if the RFID Guardian was implemented on a passive device, like an RFID tag. The RFID Guardian also performs *2-way RFID communications*. It acts like an RFID reader, querying tags and decoding the tag responses. But far more interestingly, the RFID Guardian can also emulate an RFID tag, allowing it to perform direct in-band communications with other RFID readers. As we will see later, this tag emulation capability allows the RFID Guardian to perform security protocols directly with RFID readers.

The heart of the RFID Guardian is that it integrates four previously separate security properties into a single device:

1. Auditing (Discussed in Sect. 3.1)
2. Key management (Discussed in Sect. 3.2)

3. Access control (Discussed in Sect. 3.3)
4. Authentication (Discussed in Sect. 3.4)

Some of these security properties have never been available within the context of RFID before, and other properties have combined or extended existing mechanisms.

3.1 Auditing

Auditing is the act of recording and reviewing events that happen in the world. Just as regulatory bodies might audit corporate finances or mobile telephone usage, the RFID Guardian audits all RFID activity within radio range. RFID auditing serves multiple functions: It acts as a deterrent against abuse, it provides a means to detect illicit activity, and it provides a source of "evidence" to support later correctional measures. The RFID Guardian supports two forms of auditing, *RFID scan logging* and *RFID tag logging*, both of which are new in the context of RFID.

RFID Scan Logging. Nancy's favorite department store has recently discovered that RFID scanning is an excellent way to do targeted advertising ("You recently bought a Prada sweater – maybe you would be interested in buying our matching handbag"). Unfortunately, contrary to local privacy laws, the store manager forgot to put up a sign notifying the customers about the RFID scans.

RFID Scan Logging allows consumers to audit RFID scans in the vicinity. The RFID Guardian uses its "tag emulation" capabilities to listen to and decode the RFID scans in its environment. For each query, it records such information as: command codes, flags, parameters (e.g. RFID tag queried), passed data, and annotations (e.g. timestamp). The RFID Guardian stores this information and displays it upon request, similar to the way Internet firewalls record and display intrusion attempts. This information should ideally be filtered, based upon relevance to the user (e.g. the user's tags are specifically queried). [2] This log of RFID scans then enables the consumer to report illegal RFID scanning to the proper authorities.

RFID Tag Logging. RFID is not always desired by the general public, but its deployment is tolerated because the consumer can always choose to remove or deactivate RFID tags. The only problem is that knowledge of an RFID tag's existence is a necessary precondition for the tag's removal. A stalker could drop an RFID tag into Nancy's purse, or a well-meaning department store could forget to notify her about the RFID tag attached to her new sweater. The result is that, regardless of how it got there, Nancy is now RFID-trackable. And without knowing that the RFID tag is there, she is robbed of her liberty to deactivate it.

[2] Strict filtering and adequate storage space can help mitigate Denial of Service attacks that abuse RFID Scan Logging

RFID Tag Logging offers a solution by alerting individuals about RFID tags that appear "stuck" to them. The RFID Guardian conducts periodic RFID scans, which detect all tags within radio range. It then correlates to find the RFID tags that remain constant across time, and alerts the user of the discovery of these new tags. For example, when Nancy returns home with her sweater at the end of the day, the RFID Guardian can inform her that "one new RFID tag has been added since this morning". The frequency of scanning and tag discovery reports can be increased or decreased, but there is a tradeoff between privacy, accuracy, and battery life. Scanning too infrequently may not discover RFID tags until long after they have compromised the user's privacy. However, scanning often will place high demands on the RFID Guardian's battery, and frequent reporting increases the chance of "false positives".

3.2 Key Management

As RFID technology continues to improve, consumers find themselves with an increasing number of on-tag RFID security mechanisms. Consumers can deactivate and reactivate their RFID tags using kill, sleep, and wake operations, and can perform encryption, decryption, or authentication with crypto-enabled tags (see Sect. 2.1). Each of these on-tag security mechanisms require the use of secret authorization or cryptographic keys. Like most shared secrets, these RFID tag key values must be established, available on-demand, and periodically updated to adequately protect the security of the users.

The RFID Guardian is well suited to manage RFID tag keys for several reasons. First, the RFID Guardian's ability to perform 2-way RFID communications permits key transfer without relying upon the presence of extra non-RFID infrastructure. [3] Additionally, the RFID Guardian serves as a fully-functional RFID reader, so it can use tag keys "on-demand", to activate and deactivate security features on all RFID tags within radio range. Finally, the RFID Guardian can assist with refreshing RFID tag keys, by generating pseudorandom (or truly random) values, and assigning these new values to the appropriate tags with RFID queries. This entropy generation support is useful because some low-cost RFID tags might not be able to generate their own random key material.

3.3 Access Control

Nancy wants her RFID tagged items to work at the proper times; the RFID tag in her sweater must work with her washing machine, and the tags in her groceries must work with her smart refrigerator and microwave. However, Nancy is aware of the privacy risks inherent to RFID, and she does not want her tags to be readable by the entire world. Access control addresses Nancy's concerns by actively controlling which RFID readers can query which RFID tags under which

[3] A secure (encrypted and mutually authenticated) channel is required for RFID tag key transfer between RFID Readers and the RFID Guardian.

circumstances. The RFID Guardian provides granular access control by leveraging three main features: *coordination of security primitives*, *context-awareness*, and *tag-reader mediation*. All of these features are new in the context of RFID.

Coordination of Security Primitives. Nancy's desires reflecting the activity/inactivity of her tags are represented by a security policy, which is enforced by one or multiple access control mechanisms. In other words, Nancy has a variety of tools (e.g. hash locks, sleep/wake modes, pseudonyms) that she can use to restrict access to her RFID tags. Each access control mechanism has advantages and shortcomings that make it appropriate (or inappropriate) for specific application scenarios. Since a person's situation is constantly changing, the user should be able to leverage these mechanisms in a coordinated fashion, so they can fit application constraints at any given moment while enforcing a unified security policy. No tool currently exists that can automate this process, and people do not have the ability nor the patience to use these various mechanisms manually. The RFID Guardian fills this void by offering an integrated framework for the automated management of RFID security and privacy mechanisms.

The use of a unified security policy departs from the predominant approach of decentralized RFID security, which solely considers the security needs of individual RFID tags. Centralized policies, as used in the RFID Guardian, can manage the RFID privacy of physical entities, including that of individual users and fixed locations (e.g. protecting a supermarket from the competing grocer's RFID readers). Another benefit of centralized access control is ease of management, as it eliminates the need for the propagation and synchronization of security policy updates. The main disadvantage of centralized access control is that only RFID tags within of the operating range of the RFID Guardian will receive protection.

Context-Awareness. When Nancy leaves the protective haven of her house in the morning, the RFID tags on her person are exposed to an increased amount of risk. Accordingly, Nancy expects that RFID Guardian will then tighten the access control of these RFID tags. The RFID Guardian is specially designed to adapt access control settings to reflect the reality of a person's current situation. However, the RFID Guardian is only able to make these adjustments after it first perceives the situation itself. So a form of context-awareness is necessary.

Context is a fuzzy term that is used a lot in ubiquitous computing, which essentially refers to the situation that the user is in. There are two major ways in which the RFID Guardian can detect a person's context. First, the RFID Guardian can infer its own context information. For example, the RFID Guardian might be able to detect its location, using GPS or WiFi triangulation, or it could make note of the local time. Other kinds of context can also be detected, but the more "fuzzy" the context is, the harder it becomes to detect it, and to subsequently decide how to respond to it. Second, the RFID Guardian can receive context information from RFID readers. In this case, RFID readers send the RFID Guardian textual "context updates", which consist of an arbitrary

string of data that represents some situation. For example, the RFID reader at the front door of Nancy's house could send her RFID Guardian a message, informing it that it is leaving her property. While context updates are easier to use than context inference, there are there are still problems. Any untrusted RFID reader can send a context update, so it is necessary to use authentication to check the origin of these updates (see Sect. 3.4). Another problem of relying upon context updates is that, if the RFID Guardian is not in the vicinity of an RFID reader, it has no way of being able to determine its context.

Tag-Reader Mediation. Nancy decides that she doesn't want the department store to be able to access the RFID tags on her clothing anymore, so she modifies her preferences on the RFID Guardian. The RFID Guardian could propagate the policy updates to the RFID tags themselves (assuming that the RFID tags have their own security mechanisms, which many might not). However, another option is for the RFID Guardian to act as a "man-in-the-middle", mediating interactions between RFID readers and RFID tags. This centralizes the decision making in the RFID Guardian, and leaves the RFID tags free to perform their application-specific functions, without burning valuable power on making security decisions. Mediation can take either a constructive or destructive form, which is illustrated by the two opposing concepts of "RFID Proxy Functionality" and "Selective RFID Jamming".

RFID Proxy Functionality is an example of constructive mediation where the RFID Guardian forwards cryptographically-protected queries to RFID tags on the behalf of untrusted RFID readers. By mediating RFID tag access, RFID Proxy Functionality both enables per-usage security negotiations between the RFID Guardian and RFID readers, and also reduces the need for the revocation of cryptographic RFID tag keys (since RFID readers never have the tag keys to begin with.) Here is how RFID Proxy Functionality works: An untrusted RFID reader passes a request for a desired query to the RFID Guardian, preferably over a secure channel. Upon the successful completion of a possibly complex security negotiation, the RFID Guardian then re-issues the query in encrypted form, on the behalf of the RFID reader. The RFID Guardian then receives the encrypted tag response, decrypts it, and forwards the response to the RFID reader that requested it. Prerequisites for RFID Proxy Functionality are cryptographically-enabled RFID tags, the centralized storage of RFID tag keys (see Sect. 3.2), and 2-way RFID communications between the RFID Guardian and RFID readers (see Sect. 3). Unfortunately, RFID Proxy Functionality will not work with low-cost RFID tags that are too cheap to support the required on-tag security mechanisms.

Selective RFID Jamming is an example of destructive mediation where the RFID Guardian blocks unauthorized RFID queries on the behalf of RFID tags. By filtering RFID queries, Selective RFID Jamming provides off-tag access control for low-cost RFID tags that are not powerful enough to support their own on-tag access control mechanisms. Selective RFID Jamming is a new technique, which is inspired by the RFID Blocker Tag by Juels, Rivest, and Szydlo.[9]. Here

is how Selective RFID Jamming works: An RFID reader sends a query to an RFID tag, and the RFID Guardian captures and decodes the query in real-time. It then determines whether the query is permitted, and if the query is not allowed, the RFID Guardian sends a jamming signal that is just long enough to block the RFID tag response. Selective RFID Jamming differs from the RFID Blocker Tag in that it is implemented on battery-powered mobile devices, and that it uses Access Control Lists, source authentication (see Sect. 3.4), and a randomized jamming signal. (The paper [11] offers a detailed explanation of Selective RFID Jamming.) Selective RFID Jamming has a number of problems. First, its use is legally questionable, since it is conceivably a form of signal warfare. Secondly, the use of jamming may have an adverse affect on surrounding RFID systems, if not used sparingly. And third, malicious RFID readers can abuse Selective RFID Jamming by repeatedly performing unauthorized queries. This Denial of Service attack would cause both a flurry of jamming signals, and a major drain on the battery of the RFID Guardian. For these reasons, it is preferable to use other forms of access control, so long as the application scenario permits it.

3.4 Authentication

Access control regulates which RFID readers can access which RFID tags under which circumstances. However, this mechanism needs a reliable way to determine which reader is sending any given RFID query. Some RFID tags can perform direct authentication with RFID readers, but they cannot convey the authentication results to higher-level RFID privacy management systems. In contrast, the RFID Guardian offers "off-tag authentication" by authenticating RFID readers on the behalf of the RFID tags, and directly supporting the access control methods from the previous section.

RFID Guardian-reader authentication should be implemented over the two-way RFID communications channel (see Sect. 3), using any standard challenge-response algorithm that is widely implemented and understood. This challenge-response should support both one-way and mutual authentication, to address the risk of foreign RFID Guardians. The authentication protocol is always initiated by the RFID reader, since it requests RFID tag access asynchronously from the RFID Guardian. A key distribution scheme is also necessary to facilitate the exchange of shared keys between the RFID Guardian and RFID readers. Key pre-establishment is useful for swapping keys with RFID readers that the user plans to have a lasting relationship with (e.g. the neighborhood supermarket), and this key exchange could occur using a variety of out-of-band means. On-the-fly key distribution, on the other hand, is useful when the RFID Guardian wants to establish a temporary trust relationship with an unfamiliar RFID reader. For example, Nancy may want her RFID Guardian to perform a transaction with an RFID reader located at the supermarket that she happens to be visiting. On-the-fly key distribution could use in-band or out-of-band communications, and may even rely upon a supporting Public Key Infrastructure.

4 Related Work

Many RFID security and privacy techniques exist, but there is nothing in the state-of-the-art that provides all of the security properties of the RFID Guardian. Two-way RFID communications have been investigated by MIT's Auto-ID lab, which have designed an RFID tag emulator called the 'RFID Field Probe'. A semi-passive RFID tag is used to perform real-time diagnostics on RFID equipment, and their planned 'third generation' field probe will communicate RF field values back to the RFID Reader, using in-band RFID protocols.[10] RFID auditing has been preliminarily investigated by c't magazine, who's RFID Detektor[1] lights an LED to indicate the presence of any RFID activity. RFID tag key management hasn't been systematically addressed until this point, beyond a few suggestions to transfer RFID tag keys by printing keys on cash register receipts, saving keys on smart cards, emailing keys, sending keys to a PDA using non-RFID communications. Each of these methods are less usable than the RFID tag key management that the RFID Guardian provides.

RFID tag-reader authentication, access control, and cryptography schemes provide potentially useful tools for the RFID Guardian to leverage and coordinate. Vajda and Buttyan offer lightweight authentication protocols [12], and Weis, et. al, proposed a randomized hash lock protocol for authentication[13]. Feldhofer, et. al, proposes an extension to the ISO 18000 protocol, that would enable the in-band transmission of authentication data [3]. RFID access control mechanisms include tag deactivation, which was standardized by the EPCglobal consortium [2]. Juels also suggests the use of dynamic tag identifiers, called pseudonyms, that use a mechanism called "pseudonym throttling" to allow authenticated RFID readers to refresh the pseudonym list. [8] Juels, Rivest, and Szydlo also propose the RFID Blocker Tag, that interferes with RFID Readers by "spoofing" the RFID Reader's tree-walk singulation protocol.[9] Some cryptography is also suitable for the limited resources of RFID tags. Finkenzeller describes the use of stream ciphers, [5], and Feldhofer, et. al, describes a low-cost AES implementation, simulated to work in RFID tags. [4] Gaubatz, et. al, also describe a low cost NTRU implementation, designed for sensor networks, that brings public key cryptography closer to fitting the constraints of RFID [6].

5 Conclusion and Future Work

The RFID Guardian is a new approach for personal RFID security and privacy management. It is a compact battery-powered device, that ordinary people can carry with them in RFID-tagged environments. The RFID Guardian leverages in-band RFID communications to integrate four previously separate security properties into a single device: auditing, key management, access control, and authentication. This offers some functionality that is totally new within the realm of RFID, and facilitates the coordinated usage of existing RFID security and privacy mechanisms.

The RFID Guardian has a number of issues that require further research. The bulk of our future work includes designing the security protocols that will

hold this entire RFID personal privacy management architecture together. A big problem is that the RFID Guardian is a single point of failure. Anyone who compromises the Guardian has total control over the RFID tags, whether it is lost or taken over by a hostile entity. This can be improved by using PIN codes to lock the device, and synchronizing the information on the RFID Guardian with trusted fixed location (e.g. home-based) RFID systems. Lastly, we are currently working on an implementation of the RFID Guardian, which will be used to test and extend the ideas in this paper.

References

[1] c't magazine, *Bauanleitung fur einen simplen rfid-detektor*, (2004), no. 9.
[2] EPCglobal, *13.56 MHz ISM band class 1 radio frequency (RF) identification tag interface specification*.
[3] Martin Feldhofer, *An authentication protocol in a security layer for RFID smart tags*, The 12th IEEE Mediterranean Electrotechnical Conference, vol. 2, IEEE, May 2004, pp. 759–762.
[4] Martin Feldhofer, Sandra Dominikus, and Johannes Wolkerstorfer, *Strong authentication for RFID systems using the AES algorithm*, Workshop on Cryptographic Hardware and Embedded Systems, LNCS, vol. 3156, IACR, Springer-Verlag, Aug 2004, pp. 357–370.
[5] Klaus Finkenzeller, *RFID Handbook: Fundamentals and applications in contactless smart cards and identification*, John Wiley & Sons, Ltd., 2003.
[6] G. Gaubatz, J.-P. Kaps, E. Ozturk, and B. Sunar, *State of the art in public-key cryptography for wireless sensor networks*, Proceedings of the Second IEEE International Workshop on Pervasive Computing and Communication Security, 2005.
[7] Jan E. Hennig, Peter B. Ladkin, and Bernd Sieker, *Privacy enhancing technology concepts for RFID technology scrutinised*, Research Report RVS-RR-04-02, University of Bielefeld, D-33501 Bielefeld, Germany, Oct 2004.
[8] Ari Juels, *Minimalist cryptography for low-cost RFID tags*, The Fourth International Conference on Security in Communication Networks, LNCS, Springer-Verlag, September 2004.
[9] Ari Juels, Ronald L. Rivest, and Michael Szydlo, *The blocker tag: Selective blocking of RFID tags for consumer privacy*, Proceedings of the 10th ACM Conference on Computer and Communications Security, ACM Press, 2003.
[10] Rich Redemske, *Tools for RFID testing and measurement*, 2005.
[11] Melanie R. Rieback, Bruno Crispo, and Andrew S. Tanenbaum, *Keep on blockin' in the free world: Personal access control for low-cost RFID tags*, 13th International Workshop on Security Protocols, Apr 2005.
[12] István Vajda and Levente Buttyán, *Lightweight authentication protocols for low-cost RFID tags*, Second Workshop on Security in Ubiquitous Computing, October 2003.
[13] Stephen Weis, Sanjay Sarma, Ronald Rivest, and Daniel Engels, *Security and privacy aspects of low-cost radio frequency identification systems*, Security in Pervasive Computing, LNCS, vol. 2802, 2004, pp. 201–212.

Enhanced DES Implementation Secure Against High-Order Differential Power Analysis in Smartcards

Jiqiang Lv and Yongfei Han

ONETS Wireless&Internet Security Tech. Co., LTD.
No.29, East Chuangye Road, Shangdi Information Industry Base,
Haidian District, Beijing, China 100085
lvjiqiang@hotmail.com, yongfei_han@onets.com.cn

Abstract. Since Differential Power Analysis (DPA) on DES in smartcards was firstly published by Kocher *et al.* in 1999, many countermeasures have been proposed to protect cryptographic algorithms from the attack, of which masking is an efficient and easily implemented method. In this paper, after showing some attacks on Akkar *et al.* 's improved DES implementation from FSE'04, we list and prove some basic requirements for a DES implementation using masking methods to defense High-Order DPA attacks, then present an enhancement of Akkar *et al.* 's DES implementation, which requires only three random 32-bit masks and six additional S-Boxes to be generated every computation. Finally, we prove that three random 32-bit masks and six additional S-Boxes are the minimal cost for a DES implementation masking all the outputs of the S-Boxes of the sixteen rounds to be secure against High-Order DPA attacks.

Key words: Smart-cards; DES; Simple power analysis (SPA); (High-Order) Differential power analysis (DPA); Boolean masking

1 Introduction

Differential Power Analysis (DPA)[9,10] was introduced by Kocher *et al.* in 1998 and subsequently published in 1999. It starts from the fact that the attacker can get much more information than the knowledge of the inputs and the outputs during the execution of the algorithm, such as the electric consumption or electromagnetic radiations of the circuit devices, then tries to extract information about the secret key of a cryptographic algorithm by studying the power consumption of the electronic devices during the execution of the algorithm.

To secure cryptographic algorithms against DPA attacks, two main categories of countermeasures have been presented by now. In one direction, Goubin *et al.* [7] and Char *et al.* [4] described a generic countermeasure consisting in "splitting" all the intermediate variables using some secret sharing principle. Its drawback is that it greatly increases the computation time and the memory required, which is a weakness in some constrained environments such as smart-cards. In

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 195–206, 2005.

the other direction, Messerges [12] proposed a general method that "masks" all the intermediate data, which is possible if all the fundamental operations used in a given algorithm can be rewritten with masked input data, giving masked output data. Since the masking method is easy and efficient to be implemented in some algorithms, namely DES, it has received extensive research [1,5,6,8]. Both the two main methods have been proven secure against the initial DPA attacks, however, they do not take into consideration more elaborated attacks called High-Order DPA attacks that consist in studying correlations between the secret data and several points of the electric consumption curves [10,11]. To protect some secret-key cryptographic algorithms against High-Order DPA attacks, Akkar and Giraud introduced a new countermeasure called Unique Masking Method, and applied it to DES implementation [2]. Unfortunately, based on the fact that the output of the S-Box of the second round is unmasked, Akkar, Bévan and Goubin recently presented an enhanced DPA attack on Akkar and Giraud's DES implementation using Unique Masking Method and they finally gave an improved DES implementation using Unique Masking Method to avoid this enhanced DPA attack [3].

In this paper, after briefly describing DPA and High-Order DPA attacks in Section 2, we show in Section 3 that Akkar *et al.* 's improved DES implementation is still vulnerable to High-Order DPA attacks. Following, to achieve perfect security and performances, we list and prove some basic requirements for a DES implementation using masking methods to defense (High-Order) DPA attacks in Section 4. In Section 5, we present an enhancement of Akkar *et al.* 's DES implementation, which requires only three random 32-bit masks and six additional S-Boxes to be generated every computation. In addition, we prove that three random 32-bit masks and six additional S-Boxes are the minimal cost for a DES implementation masking all the outputs of the S-Boxes of the sixteen rounds to be secure against High-Order DPA attacks. In Section 6, we discuss the security and performance of the enhanced DES implementation. Conclusion and future works will be made in Section 7.

2 DPA and High-Order DPA

The DPA attack initially focuses was on DES[13], which can be performed as follows (cited from [7]):

Step 1: We measure the consumption on the first round, for 1000 (for example) DES computations. We denote by $M_1, \cdots, M_{1000}$ the input values of those 1000 computations and $C_1, \cdots, C_{1000}$ the 1000 electric consumption curves measured during the computations. We also compute the mean curve MC of those 1000 consumption curves.

Step 2: We focus for instance on the first output bit (as the target bit) of the first S-Box during the first round. Let b be the value of that bit. It is easy to see that b depends on only 6 bits of the secret key. We make an hypothesis on the involved 6 bits. We compute the expected (theoretical) values for b from those 6 bits and from the M_i $(i = 1, \cdots, 1000)$. This enables us to separate the 1000

inputs $M_1, \cdots, M_{1000}$ into two categories: those giving $b = 0$ and those giving $b = 1$.
Step 3: We now compute the mean MC_0 of the curves corresponding to inputs of the first category. If MC and MC_0 show an appreciable difference much greater than the standard deviation of the measured noise in a statistical meaning, we consider that the chosen values for the 6 key bits were correct. If MC and MC_0 do not show any sensible difference, we repeat step 2 with another choice for the 6 bits.
Step 4: We repeat steps 2 and 3 with a "target" bit b in the second S-Box, the third, $\cdots$, until the eighth S-Box. As a result, we finally obtain 48 bits of the secret key.
Step 5: The remaining 8 bits can be found by exhaustive search.

This attack relies on the following fundamental hypothesis [2]:
Fundamental Hypothesis (Order 1) There exists an intermediate variable, that appears during the computation of the algorithm, such that knowing a few key bits (in practice less than 32 bits) allows to decide whether two inputs (respectively two outputs) give or not the same value for a known function of this variable.

High-Order DPA attacks generalize the DPA: the attacker now compute statistical correlations between the electrical consumptions considered at several instants. More precisely, an n-th order DPA attack takes into account n values of the consumption signal, which correspond to n intermediate values occurring during the computation. These attacks now rely on the following fundamental hypothesis [2],
Fundamental Hypothesis (Order n) There exists a set of n intermediate variables, that appear during the computation of the algorithm, such that knowing a few key bits (in practice less than 32 bits) allows to decide whether two inputs (respectively two outputs) give or not the same value for a known function of these n variables.

3 Attacks on Akkar, Bévan and Goubin's Improved DES Implementation Using Unique Masking Method

In this section, we will briefly review Akkar *et al.* 's improved DES implementation using Unique Masking Method, and then show our attacks.

3.1 Akkar, Bévan and Goubin's Improved DES Implementation Using Unique Masking Method

Unique Masking Method [2] aims at providing a generic protection against any order DPA. The two principles of this method is firstly to mask only the values that depend on less than 32 bits of the key in order to prevent DPA, and secondly intermediate independent variables depending on less than 32 bits of the key should not be masked by the same value in order to thwart High-Order DPA.

After generating a 32-bit value α according to their proposed method, Akkar *et al.* firstly defined two new functions $\widehat{S}1$ and $\widehat{S}2$ based on the original DES S-Boxes function S:

$$\begin{cases} \forall x \in [0,1]^{48} : \widehat{S}_1(x) = S(x \oplus E(\alpha)) \\ \forall x \in [0,1]^{48} : \widehat{S}_2(x) = S(x) \oplus P^{-1}(\alpha) \end{cases}.$$

Then, they defined f_{K_i} to be the composition of E, the XOR with the i-th round subkey K_i the S-Box and the permutation P. Finally, they defined $\widehat{f}_{1,K_i}$ and $\widehat{f}_{2,K_i}$ by replacing S in f_{K_i} with $\widehat{S}_1$ and $\widehat{S}_2$, respectively.

Using the function f, $\widehat{f}_{1,K_i}$ and $\widehat{f}_{2,K_i}$, they obtained 5 types of different rounds using masked or unmasked values:

- A-type: The left and the right parts of the input are unmasked, and the function is f. Therefore, the left and the right parts of the output will also be unmasked.
- B-type: The left and the right parts of the input are unmasked, but the function is $\widehat{f}_2$. Therefore, the left part of the output will be unmasked, but the right part will be masked.
- C-type: The left part of the input is unmasked, but the right part is masked, and the function is $\widehat{f}_1$. Therefore, the left part of the output will be masked while the right part will be unmasked.
- D-type: The left part of the input is masked, but the right part is unmasked, and the function is f. Therefore, the left part of the output will be unmasked while the right part will be masked.
- E-type: The left part of the input is masked, but the right part is unmasked, and the function is $\widehat{f}_2$. Therefore, the left or the right part of the output will be unmasked.

To defense any order DPA attack, they gave a compatible 16 round DES implementation as follows, $IP - B_{\alpha_1}C_{\alpha_1}D_{\alpha_1}C_{\alpha_1}D_{\alpha_1}C_{\alpha_1}E_{\alpha_1}B_{\alpha_2}C_{\alpha_2}D_{\alpha_2}C_{\alpha_2}D_{\alpha_2}$ $C_{\alpha_2}D_{\alpha_2}C_{\alpha_2}E_{\alpha_2} - FP$, where FP represents the final permutation of DES without countermeasures and B_{α_1} (*et al.*) denotes that the round is a B-type with the mask α_1.

Furthermore, they pointed out that if one wants the mask never to appear several times, even on values depending on more than 36 bits of the key, one can use the following combination instead of the above one: $IP - B_{\alpha_1}C_{\alpha_1}E_{\alpha_1}AAAAA$ $AAAAAB_{\alpha_2}C_{\alpha_2}E_{\alpha_2} - FP$. It is even possible to add two new masks and to mask every values depending on less than 56 bits of the key.

However, Akkar, Bévan and Goubin [3] pointed out in FSE'04 that for all the proposed sequences of rounds above, the second round is always a "C"-type round and the output of the S-Box of this second round is

$$\begin{aligned} & S\ (E(P(S(K_1 \oplus E(IP(M)_{32-63}))) \oplus IP(M)_{0-31} \oplus \alpha_1) \oplus K_2 \oplus E(\alpha_1)) \\ & = S(E(P(S(K_1 \oplus E(IP(M)_{32-63})))) \oplus K_2 \oplus E(IP(M)_{0-31})). \end{aligned}$$

It is unmasked and stay unmasked after being XORed with the left part of the message, which will be vulnerable to the attack shown in [3].

Finally, to improve the DES implementation by masking the output of the second round, they pointed out that one can use a different mask but the use of α_1 is not forbidden since the bits that are masked by the same value depends on 42 bits of the key, so they defined one more function $\widehat{f}_{3,K_i}$ with the modified S-Boxes $\widehat{S}_3(x)$ such that $\forall x \in [0,1]^{48} : \widehat{S}_3(x \oplus E(\alpha_1)) = S(x) \oplus P^{-1}(\alpha_1)$. Hereafter, the output of the S-Boxes of the second round in the improved DES implementation will be

$$\begin{aligned} & S\ (E(P(S(K_1 \oplus E(IP(M)_{32-63}))) \oplus IP(M)_{0-31} \oplus \alpha_1) \oplus K_2) \\ & = S(E(P(S(K_1 \oplus E(IP(M)_{32-63})))) \oplus E(IP(M)_{0-31}) \oplus E(\alpha_1) \oplus K_2) \\ & = S(E(P(S(K_1 \oplus E(IP(M)_{32-63})))) \oplus K_2 \oplus E(IP(M)_{0-31})) \oplus P^{-1}(\alpha_1). \end{aligned} \quad (1)$$

Note that every encryption there will be a random and different value $P^{-1}(\alpha_1)$ that is unknown to the attacker in Eqn.(1), so the attacker cannot any longer classify correctly the messages into two groups, which disables the above attack.

3.2 Our Attacks

Our attacks are based on the fact that there is the same mask in the outputs of the S-Boxes of the first two rounds in Akkar *et al.*'s improved DES implementation using Unique Masking Method.

During Akkar *et al.*'s improved DES implementation using Unique Masking Method in Section 3.1, one can see that:

Step 1: The output of the S-Box of the first round is

$$S(K_1 \oplus E(IP(M)_{32-63})) \oplus P^{-1}(\alpha_1). \quad (2)$$

Step 2: The output of the S-Box of the second round is Eqn.(1).

Step 3: By taking XOR of the outputs of S-Boxes of the first two rounds in this DES implementation (that is the XOR of Eqn. (1) and (2)) , we can get a value $\widehat{T}$ as

$$\begin{aligned} \widehat{T} = S(\underline{E(P(S(K_1 \oplus E(IP(M)_{32-63})))) \oplus K_2} \oplus E(IP(M)_{0-31})) \oplus \\ S(K_1 \oplus E(IP(M)_{32-63})). \end{aligned} \quad (3)$$

Note that the random value $P^{-1}(\alpha_1)$ vanishes in Eqn.(3), and hereafter we have two methods to perform an attack.

The first one: By fixing the right 32 bits of each message after IP to some arbitrary value and letting the left 32 bits change to get the enough inputs, we can correctly get the underlined value in Eqn.(3), and K_1 simultaneously by performing a High-Order DPA attack similar to Akkar and Giraud's superposition attack in [2].

The second one: Note that after making an hypothesis on K_1, if $IP(M)_{32-63}$ is set to some arbitrary but fixed value, then $S(K_1 \oplus E(IP(M)_{32-63}))$ will also be fixed. Following, if we classify the 1000 electric consumption curves corresponding to some 1000 inputs (the right 32 bits of each message after IP is

fixed to a constant and the left 32 bits different) according to some target bit in $\widehat{T}$, we can also classify correctly them to the same two groups according to the corresponding bit in $S(E(P(S(K_1 \oplus E(IP(M)_{32-63})))) \oplus K_2 \oplus E(IP(M)_{0-31}))$. Therefore, after fixing the right 32 bits of each message after IP to some constant M_A and letting the left part change to get the enough inputs, we can perform a DPA attack with some chosen messages to acquire the value $\theta_A = K_2 \oplus E(P(S(K_1 \oplus E(M_A))))$. Again, after fixing the right 32 bits of each message after IP to another value M_B different from M_A, we can then perform another DPA attack with some other chosen messages to acquire a similar value $\theta_B = K_2 \oplus E(P(S(K_1 \oplus E(M_B))))$. After taking XOR of the two acquired values, θ_A and θ_B, we can finally get the equation

$$S\ (K_1 \oplus E(M_A)) \oplus S(K_1 \oplus E(M_B)) = P^{-1}(E^{-1}(\theta_A \oplus \theta_B)),$$

where E^{-1} is the inverse of E.

The differential properties of S will give us about 4 possibilities for each subkey. Since there are 8 subkeys and furthermore, we also need to find the 8 bits which are not in K_1, this gives us $4^8 \cdot 2^8 = 2^{24}$ possibilities on the key, which can be finished in several seconds on a PC.

4 Basic Requirements for DES Implementation Using Masking Methods to Be Secure Against DPA Attacks

4.1 Basic Requirements

Due to the diffusion property of E and P permutations and S-Boxes in the DES, the DPA attacks make use of the two first and the two last rounds. For a DES implementation using masking methods to defense (High-Order) DPA attacks, at least the following five requirements should be met,

- Req. 1. Every crucial intermediate value should be masked by some random integer.
- Req. 2. The XORed value of the outputs of the S-Boxes of the first and the last rounds of the DES implementation using masking method should be masked by some random integer.
- Req. 3. The XORed value of the outputs of the S-Boxes of the first two (the last two) rounds of the DES implementation using masking method should be masked by some random integer.
- Req. 4. The XORed value of the outputs of the S-Boxes of the second round and the last round (the first round and the last second round) of the DES implementation using masking method should be masked by some random integer.
- Req. 5. The XORed value of the outputs of the S-Boxes of the first two rounds and the last round (the first round and the last two rounds) of the DES implementation using masking method should be masked by some random integer.

4.2 Proof

From the existing literatures, we can learn why Req.(1) and (2) should be met. Let's just show why Req.(3)-(5) should be satisfied one by one in the following:
Req. 3: From Section 3.2, we can learn why the case of the first two rounds should be satisfied to defense high-order DPA attacks.

The similar attacks are with the case of the last two rounds, except that we should get the enough outputs that have the same right 32 bits, which may be impossible in practice, but in theory it is feasible.
Req. 4: Suppose there exists a DES implementation using masking method during which Req. (4) is not satisfied, that is, the XORed value of the outputs of the S-Boxes of the second and the last rounds (or the first and the last second rounds) is unmasked by a random integer.

Let's show the attack in the case of the second round and the last round.

We assume that C is the output corresponding to the input M in this supposed DES implementation. Then the value before the Final Permutation is $FP^{-1}(C)$, therefore we can get

$$\begin{aligned} &RoriDES16 = FP^{-1}(C)_{0-31}, \\ &LoriDES16(= RoriDES15) = FP^{-1}(C)_{32-63}, \end{aligned} \tag{4}$$

where $RoriDES(i)$ and $LoriDES(i)$ denote the right and left 32 bits of the final result of the i-th round in the DES without countermeasures, respectively.

Finally, we can deduce

$$\begin{aligned} &L\,oriDES15(RoriDES14) = P(S(K_{16} \oplus E(FP^{-1}(C)_{32-63}))) \oplus FP^{-1}(C)_{0-31}, \\ &L\,oriDES14 = P(S(E(P(S(K_{16} \oplus E(FP^{-1}(C)_{32-63})))) \oplus K_{15} \oplus \\ &\quad E(FP^{-1}(C)_{0-31}))) \oplus FP^{-1}(C)_{32-63}. \end{aligned} \tag{5}$$

By using Eqn.(4), we can get the XORed value of the outputs of S-Boxes of the second and the last rounds as follows,

$$\begin{aligned} &S\ (K_2 \oplus E(RoriDES1)) \oplus S(K_{16} \oplus E(RoriDES15)) \\ &= S(\underline{K_2 \oplus E(P(S(K_1 \oplus E(IP(M)_{32-63}))))} \oplus E(IP(M)_{0-31})) \oplus S(K_{16} \oplus \\ &\quad E(FP^{-1}(C)_{32-63})). \end{aligned} \tag{6}$$

Then, after by fixing the right 32 bits of each message after IP to some arbitrary value and letting the left 32 bits change to get the enough inputs, we can easily get the correct underlined value in Eqn.(6) and K_{16} simultaneously by performing a High-Order DPA attack similar to Akkar and Giraud's superposition attack in [2] if we could choose the inputs and get their respective outputs.

The case of the first and the last second rounds is similar except that we should get the enough ciphertexts that have the same right 32 bits, which may be impossible in practice, but in theory it is feasible.
Note: The DES implementation in [1] will be vulnerable to the corresponding attacks above, besides Akkar and Giraud's superposition attack in [2].

Req. 5: Suppose there exists a DES implementation using masking method during which Req. (5) is not satisfied, that is, the XORed value of the outputs of the S-Boxes of the first two rounds and the last round (the first round and the last two rounds) of the DES implementation is unmasked by a random integer. Then by taking XOR of the outputs of S-Boxes of the first two rounds and the last round in this DES implementation, we can get the following value,

$$S\ (\underline{E(P(S(K_1 \oplus E(IP(M)_{32-63})))) \oplus K_2} \oplus E(IP(M)_{0-31})) \oplus \\ S(K_1 \oplus E(IP(M)_{32-63})) \oplus S(K_{16} \oplus E(FP^{-1}(C)_{32-63})). \quad (7)$$

If we could choose the inputs and get their respective outputs, then, after by fixing the right 32 bits of each message after IP to some arbitrary value and letting the left 32 bits change to get the enough inputs, we can easily get the correct underlined value in Eqn.(7), K_1 and K_{16} simultaneously by performing a High-Order DPA attack similar to Akkar and Giraud's superposition attack in [2], except that here we get three 6-bit values instead of two each time.

The case of the first round and the last two rounds is similar except that we should get the enough ciphertexts that have the same right 32 bits.

5 Enhanced DES Implementation Secure Against High-Order Differential Power Analysis

5.1 Enhanced DES Implementation

The proposed enhancement requires only three random masks and six additional S-Boxes to be generated every computation, and except the S-Box in each of the sixteen rounds, it is same as the DES without countermeasures.

After generating three different random 32-bit values X_1 X_2 and X_3, we firstly define six new S-Boxes based on the original DES S-Boxes function S. For $\forall x \in [0,1]^{48}$, the S-Box $\overline{S}(x)$ of every round is as follows:

$$\begin{cases} Round\ 1,6,11,12 : \overline{S}(x) = S(x) \oplus P^{-1}(X_1) \\ Round\ 2,5,10,13 : \overline{S}(x)\ such\ that\ \overline{S}(x \oplus E(X_1)) = S(x) \oplus P^{-1}(X_2) \\ Round\ 3,4 : \overline{S}(x)\ such\ that\ \overline{S}(x \oplus E(X_2)) = S(x) \oplus P^{-1}(X_1 \oplus X_2) \\ Round\ 7,16 : \overline{S}(x) = S(x) \oplus P^{-1}(X_3) \\ Round\ 8,15 : \overline{S}(x)\ such\ that\ \overline{S}(x \oplus E(X_3)) = S(x) \oplus P^{-1}(X_2) \\ Round\ 9,14 : \overline{S}(x)\ such\ that\ \overline{S}(x \oplus E(X_2)) = S(x) \oplus P^{-1}(X_1 \oplus X_3) \end{cases}.$$

Then, we define f_{j,K_j} by replacing S in f_{K_j} with the S-Box of the j-th round, for $j = 1, \cdots, 16$.

Finally, we can get the enhanced DES implementation by replacing the original Feistel function with the new one f_{j,K_j} in the j-th round of Akkar *et al.*'s DES implementation.

It is easy to see that the enhanced DES implementation meets all the requirements in Section 4.1.

5.2 Why Three 32-Bit Random Masks and Six Additional S-Boxes Are the Minimal Cost for a Secure DES Implementation Masking All the Outputs of the S-Boxes of the Sixteen Rounds?

Theorem 1. *To defense high-order DPA attacks for a DES implementation with all the outputs of the S-Boxes of the sixteen rounds masked, the minimal number of the required random masks is 3.*

Proof: (trivial, can be easily drawn from Req.(2)-(5))

Theorem 2. *To defense high-order DPA attacks for a DES implementation with all the outputs of the S-Boxes of the sixteen rounds masked, the minimal number of the additional S-Boxes required to be generated from the S-Box of the DES without countermeasures is 6.*

The details of Theorem 2 is shown in the Appendix.

6 Discussion of Proposed Enhanced DES Implementation

6.1 Security

Needless to show in details, one can deduce that the output of the S-Box of every round is masked by some mask. Therefore, the proposed DES implementation could thwart SPA and 1-st order DPA attacks.

Let's consider the security related to High-Order DPA attacks. From the fundamental hypothesis of order n in Section 2, one can learn that, to perform a n-order DPA attack, an attacker should know n intermediate values such that knowing a few key bits (in practice less than 32 bits) allows him to decide whether two inputs (respectively two outputs) give or not the same value for a known function of these n variables.

Due to the diffusion property of E and P permutations and S-Boxes in the DES, therefore, the possible intermediate values to perform a High-Order DPA attack will be the following combinations, the outputs of S-Boxes of the first two (or the last two) rounds, the outputs of S-Boxes of the first and the last rounds, the outputs of S-Boxes of the second and the last (or the first and the fifteenth) rounds, the outputs of S-Boxes of the first two rounds and the last round (the first round and the last two rounds), which are corresponding to Req.(2)-(5), respectively. All the other combinations will violate the fundamental hypothesis (in practise less than 32 bits) in Section 2. Note that each of the XORed values of the above combinations has always some random mask, i.e. $P^{-1}(X_1 \oplus X_2)$ $(P^{-1}(X_2 \oplus X_3))$, $P^{-1}(X_1 \oplus X_3)$, $P^{-1}(X_2 \oplus X_3)$ $(P^{-1}(X_1 \oplus X_2))$, and $P^{-1}(X_1 \oplus X_2 \oplus X_3)$, respectively. Since the masks will change every encryption, so the attacker cannot correctly decide whether two inputs or outputs give or not the same value. Therefore, the enhanced DES implementation could also defense High-Order DPA attacks.

As for the other security discussion, please refer to Akkar *et al.*'s DES implementation in [2,3].

6.2 Performance

The proposed enhanced DES implementation requires six additional S-Boxes to be generated from the original S-Box in advance, which requires 288 " $\oplus$ " operations and 338 " = " operations every computation. In [2], Akkar and Giraud presented a DES implementation that has four additional S-Boxes to be generated and requires 192 " $\oplus$ " operations and 192 " = " operations. The following executions is the same as the proposed enhancement, both executing as the DES without countermeasures. Akkar and Giraud showed that the execution time of their DES implementation on an ST19 component is about 40 ms every encryption. Therefore, the proposed enhanced DES implementation would be more than 40 ms, but less than 338/192 times of 40 ms at the worst case, if it was also implemented on an ST19 component, which shows that it is applicable in a smart-card environment. But, as what we show in Section 3.2, Akkar *et al.*'s DES implementation [2] and their recent improved DES implementation [3] are both vulnerable to High-Order DPA attacks.

7 Conclusion and Future Works

Masking is an efficient and easily implemented method to counteract the DPA attack. In this paper, we firstly show some attacks on Akkar *et al.* 's improved DES implementation presented in FSE'04. Following, we list and prove some basic requirements for a DES implementation using masking methods to defense (High-Order) DPA attacks, and then present an enhancement of Akkar *et al.* 's DES implementation, which requires only three random 32-bit masks and six additional S-Boxes to be generated every computation.

However, the paper does not consider the DES implementation with some inner rounds unmasked, for it is easy to deduce some variants from the enhanced DES implementation. Though we proved that three random 32-bit masks and six additional S-Boxes are the minimal cost for a DES implementation masking all the outputs of the S-Boxes of the sixteen rounds to be secure against High-Order DPA attacks, we do not prove what is the minimal cost for a secure DES implementation, i.e., masking which of the sixteen rounds is enough to be secure against High-order DPA attacks.

Acknowledgements The authors would sincerely thank to the anonymous referees for their helpful advice to improve this paper.

References

1. M. Akkar and C. Giraud, An Implementation of DES and AES Secure against Some Attack, Proceedings of CHES'01, LNCS 2162, Springer-Verlag, 2001.
2. M. Akkar and C. Giraud, A Generic Protection against High-Order Dierential Power Analysis, Proceedings of FSE'03, LNCS 2887, Springer-Verlag, 2003.
3. M. Akkar, R. Bévan and L. Goubin, Two Power Analysis Attacks against One Mask Method, Proceedings of FSE'04, LNCS 3017, Springer-Verlag, 2004.

4. S. Char, C. Jutla, J. Rao and R. Rohatgi, Towards Sound Approaches to Counteract Power-Analysis Attacks, Proceedings of Advances in Cryptology-CRYPTO'99, LNCS 1666, Springer-Verlag,1999.
5. J. Coron and L. Goubin, On Boolean and Arithmetic Masking against Differential Power Analysis, Proceedings of CHES'00, LNCS 1965, Springer-Verlag, 2000.
6. J. Coron and A. Tchulkine, A New Algorithm for Switching from Arithmetic to Boolean Masking, Proceedings of CHES'03, LNCS 2779, Springer-Verlag, 2003.
7. L. Goubin and J. Patarin, DES and Differential Power Analysis -The Duplication Method, Proceedings of CHES'99, LNCS 1717, Springer-Verlag, 1999.
8. L. Goubin, A Sound Method for Switching between Boolean and Arithmetic Masking, Proceedings of CHES'01, LNCS 2162, Springer-Verlag, 2001.
9. P. Kocher, J. Jaffe and B. Jun, Introduction to Differential Power Analysis and Related Attacks, Technical Report, Cryptography Research Inc., 1998. Available from http://www.cryptography.com/dpa/technical/index.html
10. P. Kocher, J. Jaffe and B. Jun, Differential Power Analysis, Proceedings of of Advances in Cryptology- CRYPTO'99, LNCS 1666, Springer-Verlag, 1999.
11. T.Messerges, Using Second-Order Power Analysis to Attack DPA Resistant Software, Proceedings of CHES'2000, LNCS 1965, Springer-Verlag, 2000.
12. T. Messerges, Securing the AES Finalists Against Power Analysis Attacks, Proceedings of FSE'00, LNCS 1978, Springer-Verlag, 2001.
13. National Bereau of Standards, Data Encryption Standard, Federal Information Processing Standards Publication 46, January, 1977.

Appendix: Proof of Theorem 2

If we prove that a DES implementation with all the outputs of the S-Boxes of the sixteen rounds masked cannot be implemented with 5 or less additional S-Boxes, then since the proposed enhanced DES implementation requires six, therefore, Theorem 2 will be proved.

Assume one can implement DES masking all the outputs of the S-Boxes of the sixteen rounds with 5 or less additional S-Boxes.
Step 1. From Theorem 1, we know that the three masks in the outputs of the S-Boxes of the first two rounds and the last round should be different each other, namely X_1, X_2 and X_3. Then, we can conclude that,
1. The right 32 bits of the final result of the first round will be masked by X_1, and the left 32 bits will be unmasked;
2. The left 32 bits of the final result of the second round will be X_1, and the right 32 bits will be masked by X_2;
3. The left 32 bits of the final result of the third round will be X_2, and the mask in the left 32 bits will be undetermined;
4. The left 32 bits of the final result of the fifteenth round will be masked by X_3, and the right 32 bits will be unmasked;
5. The right 32 bits of the final result of the fourteenth round will be masked by X_3, and the mask in the right 32 bits will be undetermined.

Therefore, four different additional S-Boxes will need to be generated from the S-Box of the DES without countermeasures by now, that is,

$$First\ round: \overline{S}(x) = S(x) \oplus P^{-1}(X_1), \tag{8}$$
$$Second\ round: \overline{S}(x)\ such\ that\ \overline{S}(x \oplus E(X_1)) = S(x) \oplus P^{-1}(X_2), \tag{9}$$
$$Third\ round: \overline{S}(x)\ such\ that\ \overline{S}(x \oplus E(X_2)) = S(x) \oplus P^{-1}(X_1 \oplus Y_1), \tag{10}$$
$$Sixteenth\ round: \overline{S}(x) = S(x) \oplus P^{-1}(X_3), \tag{11}$$

where Y_1 is the mask in the right 32 bits of the final result of the third round. Y_1 cannot be X_1, otherwise, the output of the S-Box of the third round will be unmasked. Therefore, $Y_1 \in \{X_2, X_3, NULL\}$, where "$NULL$" means there is no mask or the mask is a 32-bit string of "0".

Step 2. Consequently, to implement a DES using these three masks and meet the requirements in Section 4.1 simultaneously, we can see that the mask in the output of the S-Box of the fifteenth round can only be X_2. Sequently, since the right 32 bits of the final result of the fifteenth round is unmasked, therefore, the mask in the left 32 bits of the final result of the fourteenth round can only be X_2, which means that the mask in the right 32 bits of the final result of the thirteenth round will also be X_2. The mask in the left 32 bits of the final result of the thirteenth round will be undetermined. Therefore, the S-Box $\overline{S}(x)$ of the fifteenth round is such that

$$\overline{S}(x \oplus E(X_3)) = S(x) \oplus P^{-1}(X_2), \tag{12}$$

and the S-Box of the fourteenth round is such that

$$\overline{S}(x \oplus E(X_2)) = S(x) \oplus P^{-1}(X_3 \oplus Y_2), \tag{13}$$

where Y_2 is the mask in the left 32 bits of the final result of the thirteenth round. Similarly, $Y_2 \in \{X_1, X_2, NULL\}$.

Step 3. Obviously, Eqn.(8),(9),(11) and (12) have different formats each other. Therefore, to implement a DES masking all the outputs of the S-Boxes of the sixteen rounds with 5 additional S-Boxes is to make Eqn.(10) and Eqn.(13) identical, that is, $Y_1 = X_3$ and $Y_2 = X_1$.

Following, we can determine that the 64-bit mask in the final result of every of the first to the tenth rounds is $NULL||X_1, X_1||X_2, X_2||X_3, X_3||NULL, NULL||NULL, NULL||X_1(or X_3), X_1(X_3, respectively)||X_2, X_2||X_3(X_1, respectively), X_3(X_1, respectively)||NULL, NULL||NULL$, and the 64-bit mask in the final result of every of the sixteen to the eleventh rounds is $NULL||NULL$,$X_3||NULL$, $X_2||X_3, X_1||X_2, NULL||X_1, NULL||NULL$, respectively. Therefore, there will be no mask in the final result of either the tenth round or the eleventh round, which means that the output of the S-Box of the eleventh round will be unmasked. So a DES implementation with all the outputs of the S-Boxes of the sixteen rounds masked cannot be implemented with 5 additional S-Boxes.

On the other hand, since the proposed enhanced DES implementation needs six additional S-Boxes, one can learn that Theorem 2 holds.

Improved Zero Value Attack on XTR

Régis Bevan

Oberthur Card Systems SA,
25 rue Auguste Blanche, BP 133
92800 Puteaux, France
r.bevan@oberthurcs.com

Abstract. In 2000, Lenstra and Verheul presented the XTR Public Key System which used a subgroup of the multiplicative group $GF(p^6)$ with a compact representation. In two other papers, Han *et al.* analyzed the security against power analysis of the XTR algorithms presented by Lenstra and Verheul in 2000. In particular they showed that the XTR Single Exponentiation (XTR-SE) is vulnerable to a modification of the Refined Power Analysis (MRPA) and they presented a countermeasure based on the XTR double exponentiation. In the first part of this paper, we show that this countermeasure is not efficient for some particular inputs. For these inputs, an attacker has a probability of 2/3 to retrieve the secret exponent with only one power measurement. In a second part, we show that all the inputs used by Han *et al.* for MRPA are not valid inputs for XTR. As one of these dangerous inputs can also be obtained by Fault Injection, we discuss about the different scenarios of attacks and about their respective countermeasures.

Keywords: MRPA, DFA, Power Analysis, XTR, smart cards

1 Introduction

The XTR public key system presented by Lenstra and Verheul at Crypto 2000 [12] is a traditional subgroup discrete logarithm system. Elements of a subgroup of $GF(p^6)^*$ of order dividing $p^2 - p + 1$ are represented by their trace over $GF(p^2)$. Several papers ([12,17,19]) studied the security of XTR at the mathematical level. XTR is as fast as Elliptic Curve (EC) and much faster than RSA with an assumed equivalent security. Its compact representation and its quick parameters selection make XTR well suitable for devices with few memory like smartcards. Since the introduction of the Differential Power Analysis (DPA) by Kocher *et al.* in [11], side-channels attacks are a great concern for smart cards. So Han *et al.* reviewed in [8] general power analysis attacks against the implementations of XTR presented in [12]. These implementations are shown to be immune against Simple Power Analysis (SPA) but are sensitive to Address-bit Differential Power Analysis (ADPA, [9]), to Data-bit Differential Power Analysis (DDPA) and the Doubling Attack (DA,[4]). In [8], Han *et al.* investigated classical power analysis *i.e.* with random inputs. However specific inputs have been already successfully combined to power analysis to recover secret keys [15,6]. So

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 207–217, 2005.

Han *et al.* in [7] investigated adaptations of the Refined Power Analysis [6] and of the Zero Value Analysis [1]. They presented a countermeasure called XTR-RSE. This countermeasure is based on the double exponentiation algorithm presented in [12]. They split the exponent in two random exponents then use the XTR double exponentiation which applies twice the XTR single exponentiation algorithm (XTR-SE).
In this paper, we show that specific values already pointed out by Han *et al.* in [7] create a finite state machine when they are used in the XTR single exponentiation (XTR-SE). As this state machine involves multiplications by zero all along the XTR-SE algorithm, one attacker can retrieve the exponent of the XTR-SE with only one message. Moreover as the computation between the two XTR-SE of the XTR-RSE does not change the state machine for 2 cases on 3, an attacker can retrieve the exponent used in the XTR-RSE with a very high probability.
These dangerous inputs being pointed out, we consider if they represent traces of elements of the XTR group over $GF(p^2)$. By using a membership test described in [13], we show that these values are not traces of elements of the XTR group and even all the values exhibit in [7] are not traces of the XTR group. The countermeasure can be then straightforward but we point out that some of the dangerous inputs can be obtained by Fault Attacks as mentioned in [3]. So we analyze the scenarios of attacks and the countermeasures for the different use of XTR in cryptography.
The article is organized as follows. We use Section 2 to briefly recapitulate operations involved in the XTR single exponentiation presented in [12] and recall the countermeasure XTR-RSE described in [7]. Then classes of inputs producing multiplications by zero all along the exponentiation are discussed in Section 3. Their membership to the XTR group is evaluated in Section 4. Finally the scenarios of attacks and their respective countermeasures are described in Section 5.

2 Description of XTR and of the XTR-RSE Algorithm

Let p and q be two prime numbers. Let $g \in GF(p^6)^*$ be an element of order q dividing $p^2 - p + 1$. As shown in [12], the subgroup $\langle g \rangle$ is included in $GF(p^6)^*$ but is not included in $GF(p^i)^*$ for every $i \in \{1, 2, 3\}$. Consequently, solving the discrete logarithm problem is as hard in $\langle g \rangle$ as in $GF(p^6)^*$. Instead of using the representation of the elements of $\langle g \rangle$ with six coordinates in $GF(p)$, XTR represents them by their traces over $GF(p^2)$. In this section, we remind some results established in [12]. More particulary the arithmetic operations in $GF(p^2)$ and the single exponentiation operation XTR-SE. We then recall the countermeasure proposed in [7].
We use p such that $p \equiv 2 \bmod 3$ until the end of the article.

2.1 Arithmetic Operations in $GF(p^2)$

As $p \equiv 2 \bmod 3$, the polynomial $X^2 + X + 1$ is irreducible over $GF(p^2)$. The roots α and α^p of this polynomial form an optimal normal basis for $GF(p^2)$ over $GF(p)$. Moreover, since $p \equiv 2 \bmod 3$, $\alpha^i = \alpha^{i \bmod 3}$. It follows that:

$$GF(p^2) \simeq \{x_1\alpha + x_2\alpha^2, \text{ with } \alpha^2 + \alpha + 1 = 0 \text{ and } x_1, x_2 \in GF(p)\}$$

Each element of $GF(p^2)$ can thus be represented as a couple (x_1, x_2) where $x_1, x_2 \in GF(p)$.
Let $x, y, z \in GF(p^2)$ with $p \equiv 2 \bmod 3$. The four basic operations used in XTR are the following:

- p^{th} powering:
  $x^p = (x_1\alpha + x_2\alpha^2)^p = x_1^p\alpha^p + x_2^p\alpha^{2p} = x_2\alpha + x_1\alpha^2$
- squaring:
  $x^2 = x_1\alpha^2 + 2x_1x_2\alpha^3 + x_2^2\alpha^4 = x_2(x_2 - 2x_1)\alpha + x_1(x_1 - 2x_2)\alpha^2$
- multiplication:
  $xy = (x_1y_1 + 2x_2y_2 - (x_1 + x_2)(y_1 + y_2))\alpha$
  $+(2x_1y_1 + x_2y_2 - (x_1 + x_2)(y_1 + y_2))\alpha^2$
- computing $xz - yz^p$:
  $xz - yz^p = (z_1(y_1 - x_2 - y_2) + z_2(x_2 - x_1 + y_2)))\alpha$
  $+(z_1(x_1 - x_2 + y_1) + z_2(y_2 - x_1 - y_1))\alpha^2$

2.2 XTR Single Exponentiation

XTR is a method which represents elements of a subgroup embedded in the field $GF(p^6)$ by their trace over $GF(p^2)$. We used the trace function Tr defined as follow:

Definition 1 $\forall a \in GF(p^6)$, $\mathrm{Tr}(a) = a + a^{p^2} + a^{p^4}$.

The main idea of XTR is to use the trace representation of elements of $GF(p^6)$ over $GF(p^2)$ to have a compact representation. More particulary, Lenstra and Verheul presented in [12] methods to get $c \in GF(p^2)$, such that $c = \mathrm{Tr}(g)$ with g, an element of $GF(p^6)^*$ of order q dividing $p^2 - p + 1$. In order to apply XTR to cryptographic applications, we must be able to compute $c_n = \mathrm{Tr}(g^n)$ in a fast way. The following algorithm proposed in [12] produces such a computation.
The following notations are used in the algorithm:

- $c_i = \mathrm{Tr}(g^i)$.
- $S_i(c) = (c_{i-1}, c_i, c_{i+1}) \in (GF(p^2))^3$.

Remark 1. This algorithm can be applied for all elements c of $GF(p^2)$, not only for elements $c = \mathrm{Tr}(g)$. This property can be used to mount chosen inputs attacks.

Algorithm 2.1 XTR Single Exponentiation
INPUT: $c = \mathrm{Tr}(g)$, the exponent n
OUTPUT: $S_n(c) = (\mathrm{Tr}(g^{n-1}), \mathrm{Tr}(g^n), \mathrm{Tr}(g^{n+1}))$

$c_2 = (3, c, c^2 - 2c^p)$
$c_3 = (c, c^2 - 2c^p, c * c_2 - c^p * c + 3)$
$c_4 = (c_2, c_3, c_2^2 - 2 * c_2^p)$
if n is odd,
  let $m = \dfrac{(n-1)}{2} = \sum_{i=0}^{r} m_i 2^i$ with $m_i \in \{0, 1\}$
  $k = 1$, $V = (c_2, c_3, c_4)$
  for $(j = r - 1$ to $0)$
  {
    if $m_j = 0$ then update V by computing
      $V = (V(1)^2 - V(1)^p, V(1)V(2) - c^p V(2)^p + V(3)^p, V(2)^2 - 2V(2)^p)$
    if $m_j = 1$ then update V by computing
      $V = (V(2)^2 - V(2)^p, V(3)V(2) - cV(2)^p + V((1)^p, V(3)^2 - 2V((3)^p)$
    $k = k + m_j$
  }
  After this iteration, we have $k = m$ and $S_n(c) = V$
else use the algorithm describes upper to compute $S_{n-1}(c)$. Afterwards compute $S_n(c) = (c_{n-1}, c_n, c \times c_n - c^p c_{n-1} + c_{n-2})$

2.3 XTR-RSE

After two studies of power analysis on XTR, Han *et al.* proposed in [7] a countermeasure which seemed to be efficient against all power analysis attacks (Simple Power Analysis, Differential Power Analysis, MRPA, MVZA, Doubling Attack). This countermeasure is based on the XTR double exponentiation proposed in [12]. Randomly splitting the exponent make impossible for an attacker to find information on the secret exponent. Algorithm 2.2 describes this countermeasure.

Algorithm 2.2 XTR Randomized Single Exponentiation
INPUT: $c = \mathrm{Tr}(g)$, the exponent n
OUTPUT: $S_n(c) = (\mathrm{Tr}(g^{n-1}), \mathrm{Tr}(g^n), \mathrm{Tr}(g^{n+1}))$

1. Select a random number b in $[0, q]$ and compute $a = n - b \bmod q$
2. Compute $e = a/b \bmod q$
3. Compute $S_e(c) = (c_{e-1}, c_e, c_{e+1})$ with XTR-SE
4. Use c_{e+1} to compute $S_b(c_{e+1}) = S_{b(e+1)}(c) = S_n(c)$ with XTR-SE

3 Chosen Input Attack

Multiplying by zero is noticeable when analyzing the power consumption trace of a smartcard. This fact has been successfully used against different algorithms

[5,6,1]. The power consumption is much lower with an operand equals to zero than with non-null operands. If a crypto-processor is used to perform the multiplication, the detection is even easier. So Simple Power Analysis of one power measurement can reveal all the multiplications involving a null operand.
As Han *et al.* already mentioned in [7], four classes of traces produce multiplication by zero in the next step of the algorithm XTR-SE: $(0, a)$, $(a, 0)$, $(2a + 2, a)$, $(a, 2a + 2)$. Their reasoning was based on the fact that it is possible to find some traces $c = (x, y)$ such that at the i-th step of Algorithm 2.1, one has $j = \sum_{k=0}^{i} n_k 2^k$ and $c_j = (0, a)$ or $c_j = (a, 0)$ or $c_j = (2a+2, a)$ or $c_j = (a, 2a+2)$. But they did not study the case when the input c of the algorithm is equal to one of these four specific values. In the next paragraph, we show that these values create a finite state machine with only three different states. The transitions between the states are clearly noticeable on a power trace because of the difference in the sequence of multiplications by zero.

3.1 First Case: $c = (a, 2a + 2)$

If c is equal to $(a, 2a + 2)$, then :

$$S_3(c) = ([0, -6a - 3a^2], [1 - 9a^2 - 6a - 3a^3, 1 + 9a^2 + 6a + 3a^3], [12a + 42a^2 + 36a^3 + 9a^4, 0])$$

Let X_3, Y_3, Z_3 be the following values : $X_3 = -6a - 3a^2$, $Y_3 = 1 - 9a^2 - 6a - 3a^3$, $Z_3 = 12a + 42a^2 + 36a^3 + 9a^4$. Then we have $S_3(c) = ([0, X_3], [Y_3, 2 - Y_3], [Z_3, 0])$. The important point is that V, the temporary variable used throughout Algorithm 2.1 can be considered like an internal variable of a state machine. The state of the automata is not determined by the exact value of V but by the the relationships between the different coordinates of V.
Let E_1 be the state when V is equal to $([X, X], [Y, 2Y + 2], [0, Z])$, let E_2 be the state when V is equal to $([0, X], [Y, 2 - Y], [Z, 0])$, and let E_3 be the state when V is equal to $([X, 0], [2Y + 2, Y], [Z, Z])$, for all X,Y,$Z \in GF(p)$.
Figure 1 sums up the different operations involved during the exponentiation of the value $c = (a, 2a + 2)$. This shows that the coordinates of V (*i.e.* the intermediate values of the algorithm) follow the state machine described in Figure 3. The interesting fact is that the sequence of multiplication by zero depends on the value of the secret key. (When an operand is equal to zero the sign $\times$ is used to represent the multiplication.)

The case when $c = (2a + 2, a)$ is completely symmetrical so can easily be deduced from Figure 1.

3.2 Second Case: $c = (a, 0)$

If we consider that $c = (a, 0)$, then we have:

$$S_3(c) = ([0, -2a + a^2], [-3 + 3a^2 - a^3, -3 + 3a^2 - a^3], [4a + 2a^2 - 4a^3 + a^4, 0])$$

Let X_3, Y_3, Z_3 be the following values : $X_3 = -2a + a^2$, $Y_3 = -3 + 3a^2 - a^3$, $Z_3 = 4a + 2a^2 - 4a^3 + a^4$. Then we can write that $S_3(c) = ([0, X_3], [Y_3, Y_3], [Z_3, 0])$.

Transition	New coordinates of $V(1)$	New coordinates of $V(2)$	New coordinates of $V(3)$
$E_2 \rightarrow E_3$	$X(X-2\times 0-2)$	$Y(2a+2-X-a)+0$ $+(2-Y)(X-0+a)$	$(2-Y)(2-Y-2Y-2)$
(bit $=0$)	$\mathbf{0}\times(0-2X-2)$	$Y(0-X+2a+2)+Z$ $+(2-Y)(a-0-2a-2)$	$Y(Y-2(2-Y)-2)$
$E_2 \rightarrow E_1$	$(2-Y)(2-Y-2Y-2)$	$Y(a-0-2a-2)+Z$ $+(2-Y)(0-Z+2a+2)$	$\mathbf{0}\times(0-2Z-2)$
(bit $=1$)	$Y(Y-2(2-Y)-2)$	$Y(Z-0+a)+0$ $+(2-Y)(2a+2-Z-a)$	$Z(Z-2\times 0-2)$
$E_1 \rightarrow E_1$	$X(X-2X-2)$	$Y(2a+2-X-a)+Z$ $+(2Y+2)(X-X+a)$	$(2Y+2)(2Y+2-2Y-2)$
(bit $=0$)	$X(X-2X-2)$	$Y(X-X+2a+2)+0+$ $(2Y+2)(a-X-2a-2)$	$Y(Y-2(2Y+2)-2)$
$E_1 \rightarrow E_2$	$(2Y+2)(2Y+2-2Y-2)$	$Y(a-0-2a-2)+X+$ $(2Y+2)(Z-0+2a+2)$	$Z(Z-2\times 0-2)$
(bit $=1$)	$Y(Y-2(2Y+2)-2)$	$Y(0-Z+a)+X+$ $(2Y+2)(2a+2-0-a)$	$\mathbf{0}\times(0-2Z-2)$
$E_3 \rightarrow E_2$	$\mathbf{0}\times(0-2X-2)$	$(2Y+2)(2a+2-0-a)$ $+Y(0-X+a)$	$Y(Y-2(2Y+2)-2)$
(bit $=0$)	$X(X-2\times 0-2)$	$(2Y+2)(X-0+2a+2)$ $+Y(a-X-2a-2)+Z$	$(2Y+2)(2Y+2-2Y-2)$
$E_3 \rightarrow E_3$	$Y(Y-2(2Y+2)-2)$	$(2Y+2)(a-Z-2a-2)$ $+Y(Z-Z+2a+2)+0$	$Z(Z-2Z-2)$
(bit $=1$)	$(2Y+2)(2Y+2-2Y-2)$	$(2Y+2)(Z-Z+a)$ $+Y(2a+2-Z-a)+X$	$Z(Z-2Z-2)$

Fig. 1. Operations during the exponentiation when $c=(a,2a+2)$

If we denote by E_1 the state when V is equal to $([X,X],[Y,0],[0,Z])$, by E_2 the state when V is equal to $([0,X],[Y,Y],[Z,0])$ and by E_3 the state when V is equal to $([X,0],[0,Y],[Z,Z])$, for all X,Y,$Z \in GF(p)$.
Then, Figure 2 shows that the coordinates of V follow the same state machine (Figure 3) when $c=(a,0)$ that when $c=(a,2a+2)$ but the relationships between the coordinates are different.
It is easy to see that when $c=(0,a)$ the results are identical.

3.3 General Analysis

As the attacker knows the initial state, it is not difficult for him to guess the bits of the exponent one after the other by noticing the instants when an intermediate value equal to zero is used in a multiplication. Figures 1 and 2 show that, for a given state, the instants when a zero is used in a multiplication are different if the bit is equal to zero or to one. The attacker can thus guess the bits of the key without ambiguities.
Figure 4 shows an example of the attack with an input equal to $c=(a,2a+2)$. We consider that the attacker measured the power consumption during the whole execution of the algorithm but store only the power consumption of each multiplication. The measurement is equal to H (high power consumption) if the multiplication does not involve a zero, or to L (low power consumption) if the multiplication produces a zero. We suppose that the smartcard computes successively the first then the second coordinate of $V(1)$, then the coordinates of $V(2)$, and finally the coordinates of $V(3)$.

Transition	New coordoninates of $V(1)$	New coordoninates of $V(2)$	New coordoninates of $V(3)$
$E_2 \rightarrow E_3$	$X(X - 2 \times 0 - 2)$	$Y(0 - X - a)$ $+Y(X - 0 + a) + 0$	$Y(Y - 2Y - 2)$
(bit $= 0$)	$\mathbf{0}\times(0 - 2X - 2)$	$Y(0 - X + 0)$ $+Y(a - 0 - 0) + Z$	$Y(Y - 2Y - 2)$
$E_2 \rightarrow E_1$	$Y(Y - 2Y - 2)$	$Y(a - 0 - 0)$ $+Y(0 - Z + 0) + Z$	$\mathbf{0}\times(0 - 2Z - 2)$
(bit $= 1$)	$Y(Y - 2Y - 2)$	$Y(Z - 0 + a)$ $+Y(0 - Z - a) + 0$	$Z(Z - 2 \times 0 - 2)$
$E_1 \rightarrow E_1$	$X(X - 2X - 2)$	$Y(0 - X - a)$ $+\mathbf{0}\times(X - X + a) + Z$	$\mathbf{0}\times(0 - 2Y - 2)$
(bit $= 0$)	$X(X - 2X - 2)$	$Y\times(\mathbf{X - X + 0})$ $+\mathbf{0}\times(a - X - 0) + 0$	$Y(Y - 2 \times 0 - 2)$
$E_1 \rightarrow E_2$	$\mathbf{0}\times(0 - 2Y - 2)$	$Y(a - Z - 0)$ $+\mathbf{0}\times(Z - 0 + 0) + X$	$Z(Z - 2 \times 0 - 2)$
(bit $= 1$)	$Y(Y - 2 \times 0 - 2)$	$Y(0 - Z + a)$ $+\mathbf{0}\times(0 - 0 - a) + X$	$\mathbf{0}\times(0 - 2Z - 2)$
$E_3 \rightarrow E_2$	$\mathbf{0}\times(0 - 2X - 2)$	$\mathbf{0}\times(0 - 0 - a)$ $+Y(0 - X + a) + Z$	$Y(Y - 2 \times 0 - 2)$
(bit $= 0$)	$X(X - 2 \times 0 - 2)$	$\mathbf{0}\times(X - 0 + 0)$ $+Y(a - X - 0) + Z$	$\mathbf{0}\times(0 - 2Y - 2)$
$E_3 \rightarrow E_3$	$Y(Y - 2 \times 0 - 2)$	$\mathbf{0}\times(a - Z - 0)$ $+Y\times(\mathbf{Z - Z + 0}) + 0$	$Z(Z - 2Z - 2)$
(bit $= 1$)	$\mathbf{0}\times(0 - 2Y - 2)$	$\mathbf{0}\times(Z - Z + a)$ $+Y(0 - Z - a) + X$	$Z(Z - 2Z - 2)$

Fig. 2. Operations during the exponentiation of $c = (a, 0)$

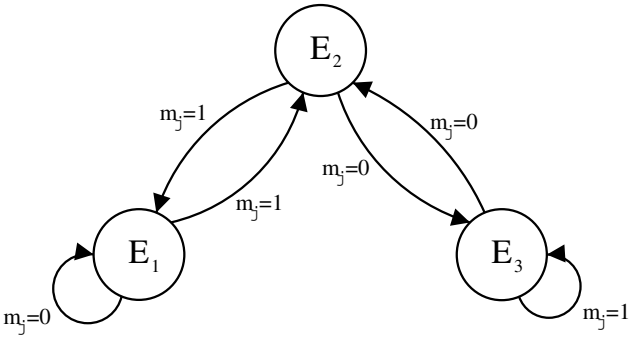


Fig. 3. State machine for a starting value c of XTR equals to $(a, 2a+2)$ or $(a, 0)$

Multiplication by 2 does not involved the use of the multiplier, since only a shift of the register is performed. As shift is not as noticeable as a multiplication on a power trace, we ignored multiplications by 2 in our power analysis.

So with only one chosen input equal to $(a, 2a + 2)$ and the complete power trace of the XTR-SE computation on this chosen input, an attacker can recover the value of the exponent (for example, the secret key for the XTR-ElGamal protocol).
The same reasoning can be used for the values of c equal to $(a, 0)$.

power consumption	$HHHHHHLH$	$HHHHHHHH$	$HHHHHHHH$	$HHHHHHHL$
exponent bits	1	0	0	1

Fig. 4. Example of an attack with $c = (a, 2a + 2)$

3.4 Impact on XTR-RSE

According to the previous section, if the attacker sends $c = (a, 0)$ as input for a smartcard using XTR-RSE (Algorithm 2.2), he can easily retrieve the exponent e used in the first XTR-SE by Simple Power Analysis. Then c_{e+1} is used as the input of the second XTR-SE. As e is random, c_{e+1} is equal to the last coordinate of one of the state describe in Figure 3. So for $c = (a, 0)$, c_{e+1} is equal to $(0, Z)$ or $(Z, 0)$ or (Z, Z). If c_{e+1} is equal to $(0, Z)$ or $(Z, 0)$, the attacker can also retrieve by Simple Power Analysis the exponent b used in the second XTR-SE of Algorithm 2.2. However, if c_{e+1} is equal to (Z, Z), one can easily check that during the second XTR-SE, V is always in the form $(u, u), (v, v), (w, w)$. Moreover there is no multiplication by zero during the exponentiation so the attacker is not able to recover b by Simple Power Analysis. Thus with a probability of $2/3$, the second XTR-SE will have an input that allows the attacker to retrieve the exponent b.
Once an attacker has recovered e and b, he can easily compute the secret key $n = (e + 1)b$. Finally, XTR-RSE does not provide enough security against chosen inputs attacks. The same conclusion can be drawn from the inputs equal to $(a, 2a + 2)$.
In the next section, we will study the membership to the XTR group of the four dangerous classes of inputs use in this attack.

4 Membership Test

According to [12], an element c of $\in GF(p^2)$ is a trace of an element g of the XTR group if and only if the polynomial $F(c, X) = X^3 - cX^2 + c^pX - 1$ is irreducible on $GF(p^2)$ and if $c_{(p^2-p+1)/q} \neq 3$. In [13], Lenstra and Verheul showed that the irreducibility of $F(c, X)$ over $GF(p^2)$ is equivalent to the irreducibility of $P(c, X) = X^3 + (c^p + c)X^2 + (c^{p+1} + c^p + c - 3)X + c^{2p} + c^2 + 2 - 2c^p - 2c$ over $GF(p)$.
For $c = (2a + 2, a)$ or $c = (a, 2a + 2)$, -1 is a root of the polynomial $P(c, X)$. And for $c = (2a + 2, a)$ or $c = (a, 2a + 2)$, 1 is a root of the polynomial $P(c, X)$. So whatever the value of a, the elements $c = (0, a)$, $c = (a, 0)$, $c = (2a + 2, a)$ and $c = (a, 2a + 2)$ are never traces of element of the XTR group.
Finally, none of the traces of the previous section is a trace of an element of the XTR group. Moreover, the fact that the polynomial $P(c, X)$ is reducible over $GF(p)$ not only implies that c is not a trace of an element of the XTR group but also implies that c is not a trace of an element of the supergroup H of order $p^2 - p + 1$. As the XTR group is cyclic, it means also that none application of the XTR-SE algorithm on the values exhibit in [7] for MRPA and MZVA will produce traces of the XTR group.

Knowing this, we can consider the different cryptographic applications involving XTR and the way to protect them against the specific inputs of Section 3.

5 Attack Scenarios and Countermeasures

5.1 Applications Where the Attacker Can Submit a Trace to the Smartcard

In applications like XTR-ElGamal, an attacker can send a wrong couple (c, E) to the smartcard with a c equal to $(a, 0)$ or $(a, 2a+2)$. To decrypt the message E, the smartcard must first perform the XTR exponentiation of c with its private key. If no test on c is done before the exponentiation then the attacker can retrieve the key by monitoring the power consumption of the smartcard, even if the XTR-RSE algorithm is used. So before computing the XTR exponentiation of an element $c = (x, y)$ coming from outside, the smartcard must check that $x(x - 2y - 2)$ and $y(y - 2x - 2)$ are different from zero.
In applications where the smartcard receives $c = (x, y)$ from the outside and must return the result of the XTR exponentiation, a membership test to the XTR group or to the supergroup H described in [13] is mandatory to avoid subgroup attacks. As shown in the previous section, the dangerous inputs belong neither to the XTR group nor to the supergroup H so no additional test is necessary to avoid chosen inputs attacks.

5.2 Applications Where the Trace Is Stored Inside the Smartcard

For applications like XTR-DSA where the trace c is stored inside the smartcard, the Zero Value Attack described in Section 3 seems not to be relevant since the XTR key generation never outputs dangerous c. However, as mentioned in [3] and [2], it is possible by Fault Induction Attack to set one coordinate of c to zero when c is read from the EEPROM of the smartcard. So by a fault attack during the reading of c and Simple Power Analysis, an attacker can retrieve the random secret exponent of one XTR-DSA and so retrieve the private key of the smartcard. In [3], they propose to randomize of the computation of the three parts of V in the Algorithm 2.1 to thwart this attack.

6 Conclusion

Four classes of inputs can reveal the XTR secret key with only one exponentiation even if the algorithm XTR-RSE is used. Nevertheless this attack can be efficiently avoided by checking the validity the input of the XTR exponentiation. Euclidean exponentiation exposed as proposed in [18] is an alternative to XTR-SE for computing XTR operations. In [16], Page and Stam already analyzed the security of Euclidean exponentiations against SPA and DPA and they proposed also an efficient countermeasure. Further work must be done to analyze the security of the Euclidian exponentiation against Zero Value Attack.

Acknowledgements

I would like to thank the people of the Cryptography and Security Group of Oberthur, in particularly Sébastien Aumonier and Christophe Giraud for their help and numerous discussions about XTR implementations.

References

1. T. Akishita and T. Takagi. Zero-value Point Attacks on Elliptic Curve Cryptosystem. In C. Boyd and W. Mao, editors, *Information Security – ISC 2003*, volume 2851 of *Lecture Notes in Computer Science*, pages 218–233. Springer-Verlag, 2003.
2. H. Bar-El, H. Choukri, D. Naccache, M. Tunstall, and C. Whelan. The Sorcerer's Apprentice Guide to Fault Attacks. In L. Breveglieri and I. Koren, editors, *Workshop on Fault Diagnosis and Tolerance in Cryptography – FDTC'04*, pages 330–342. IEEE Computer Society, 2004.
3. M. Ciet and C. Giraud. Transient Fault Induction Attacks on XTR. In López et al. [14], pages 440–451.
4. P.-A. Fouque and F. Valette. The Doubling Attack: Why Upwards is better than Downwards. In C.D. Walter, Ç.K. Koç, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2003*, volume 2779 of *Lecture Notes in Computer Science*, pages 269–280. Springer-Verlag, 2003.
5. J. Golic and C. Tymen. Multiplicative masking and power analysis of AES. In Kaliski Jr. et al. [10], pages 198–212.
6. L. Goubin. A Refined Power-Analysis Attack on Elliptic Curve Cryptosystem. In Y. Desmedt, editor, *Public Key Cryptography – PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 199–210. Springer-Verlag, 2003.
7. D.-G. Han, T. Izu, J. Lim, and K. Sakurai. Modified Power-Analysis Attacks on XTR and an Efficient Countermeasure. In López et al. [14], pages 305–317.
8. D.-G. Han, J. Lim, and K. Sakurai. On Security of XTR public key cryptosystems against Side Channel Attacks. In H. Wang, J. Pieprzyk, and V. Varadharajan, editors, *Information Security and Privacy - 9th Australasian Conference – ACISP 2004*, volume 3108 of *Lecture Notes in Computer Science*, pages 454–465. Springer-Verlag, 2004.
9. K. Itoh, T. Izu, and M. Takenak. Address-bit Differential Power Analysis of Cryptographic Schemes OK-ECDH and OK-ECDSA. In Kaliski Jr. et al. [10], pages 129–143.
10. B.S. Kaliski Jr., Ç.K. Koç, and C. Paar, editors. *Cryptographic Hardware and Embedded Systems – CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*. Springer-Verlag, 2002.
11. P. Kocher, J. Jaffe, and B. Jun. Differential Power Analysis. In M.J. Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer-Verlag, 1999.
12. A.K. Lenstra and E.R. Verheul. The XTR public key system. In M. Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 1–19. Springer-Verlag, 2000.
13. A.K. Lenstra and E.R. Verheul. Fast irreductibility and subgroup membership testing in XTR. In K. Kim, editor, *Public Key Cryptography – PKC 2001*, volume 1992 of *Lecture Notes in Computer Science*, pages 73–86. Springer-Verlag, 2001.

14. J. López, S. Qing, and E. Okamoto, editors. *International Conference on Information and Communications Security – ICICS'04*, volume 3269 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.
15. R. Novak. SPA-Based Adaptive Chosen-Ciphertext Attack on RSA Implementation. In D. Naccache and P. Paillier, editors, *Public Key Cryptography – PKC 2002*, volume 2274 of *Lecture Notes in Computer Science*, pages 252–262. Springer-Verlag, 2002.
16. D. Page and M. Stam. On XTR and Side-Channel Analysis. In M. Matsui and R. Zuccherato, editors, *Selected Areas in Cryptography – SAC 2004*, volume 3357 of *Lecture Notes in Computer Science*, pages 54–68. Springer-Verlag, 2004.
17. I.E. Shparlinski. On the Generalized Hidden Number Problem and Bit Security of XTR. In *Advances in Cryptology – the 14th Symp. on Appl. Algebra, Algebraic Algorithms, and Error-Correcting Codes*, volume 2227 of *Lecture Notes in Computer Science*, pages 268–277. Springer-Verlag, 2001.
18. M. Stam and A.K. Lenstra. Speeding up XTR. In E. Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 125–143. Springer-Verlag, 2001.
19. E.R. Verheul. Evidence that XTR Is More Secure then Supersingular Elliptic Curve Cryptosystems. In B. Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 195–210. Springer-Verlag, 2001.

Efficient Representations on Koblitz Curves with Resistance to Side Channel Attacks

Katsuyuki Okeya[1], Tsuyoshi Takagi[2], and Camille Vuillaume[1]

[1] Hitachi, Ltd., Systems Development Laboratory, Kawasaki, Japan
{camille,ka-okeya}@sdl.hitachi.co.jp
[2] Future University - Hakodate, Japan
takagi@fun.ac.jp

Abstract. Koblitz curves belong to a special class of binary curves on which the scalar multiplication can be computed very efficiently. For this reason, they are suitable candidates for implementations on low-end processors. However, such devices are often vulnerable to side channel attacks. In this paper, we propose two countermeasures against side channel attacks on Koblitz curves. Both of them utilize a fixed-pattern recoding to defeat simple power analysis. Our first technique extends a known countermeasure to the special case of Koblitz curves. In our second technique, the scalar is recoded from left to right, and can be easily stored or even randomly generated.

Keywords: *elliptic curve cryptosystems, Koblitz curves, smartcard, side channel attacks, SPA countermeasure.*

1 Introduction

Since their introduction, elliptic curve cryptosystems (ECC) have been thoroughly studied, because in contrary to RSA-type cryptosystems, they are well-suited for implementations on memory-constrained devices and low-end processors. Koblitz curves belong to a special class of curves defined over a binary field, where the primitive of ECC, namely the scalar multiplication, can be computed very efficiently. Since their efficient arithmetic has been pointed out [Kob91], no significant security flaw or practical attack has be found.

Side channel attacks are powerful attacks which use a priori innocuous information such as timings or power consumption to break implementations of cryptosystems [Koc96, KJJ99]. On light and specialized cryptodevices such as smartcards, side channel attacks are a major threat. There are two types of attack strategies based on information leakage by power consumption: simple power analysis (SPA) and differential power analysis (DPA). In the frame of SPA, the attacker uses only one power curve to guess the secret information, whereas he/she is allowed to use a statistical tool in order to extract information from several power traces in the frame of DPA [KJJ99]. On Koblitz curves, the standard DPA countermeasures can be deployed. However, SPA resistance is problematic: the known countermeasures for general curves are either based on

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 218–229, 2005.

dummy operations [Cor99, CCJ04] or scalar recoding using properties of binary representations [OT03]. First, dummy-based countermeasures may be vulnerable to fault attacks [YJ00], therefore, they should be avoided. Second, on Koblitz curves, in order to speed-up the computations, the scalar is recoded using a τ-adic expansion, where τ is the solution of a quadratic equation, instead of a binary expansion. Therefore, countermeasures based on binary representation tricks are not straight-forwardly applicable.

In this paper, we propose two new countermeasures against side channel attacks on Koblitz curves. The first technique extends the mechanisms of a countermeasure for general curves, namely the SPA-resistant NAF_w [OT03], to the special arithmetic of Koblitz curves. The original SPA-resistant NAF_w utilizes special properties of binary expansions to generate a secure representation. We show how to transpose the mechanisms of the countermeasure to Koblitz curves. The second technique utilizes a two-round recoding. First, it generates a zero-free representation using the principles of our first countermeasure. Second, it applies a windowing technique in order to take advantage of pre-computed points and consequently reduce computational costs. Then, we emphasize interesting properties of our schemes. On elliptic curves, left-to-right computations are usually faster. Thus, it is preferable to use a left-to-right recoding approach as well: the recoding and the scalar multiplication can be combined, and no memory is needed to store the scalar in multiple representations. We show practical situations where our ideas are compatible with a left-to-right recoding. First, when the scalar is fixed (e.g. EC-ElGamal decryption), using the proposed zero-free representation, the scalar can be stored once for all, and the windowing technique can be applied on the fly. Second, when a secret ephemeral is needed (in EC-DSA signature generation or EC-DH), we can generate this information on the fly while computing the scalar multiplication with the zero-free technique. Therefore, in all practical situations where SPA-resistance is needed, our countermeasures can be deployed to protect secret information against side channel attacks, providing a high security level, great efficiency, smart and small memory usage.

2 Preliminaries

In this section, we discuss known facts: we introduce Koblitz curves and discuss the feasibility of side channel attacks on them.

2.1 Koblitz Curves

Koblitz curves are binary elliptic curves which offer a very efficient arithmetic with no significant security drawback [Kob91]. They are defined over a binary field $\mathbb{F}_{2^m}$ by the equation: $\mathcal{E}_a = y^2 + xy = x^3 + ax^2 + 1$, where $a \in \{0, 1\}$. We denote by $\mathcal{E}_a(\mathbb{F}_{2^m})$ the additive group of the points of the elliptic curve over $\mathbb{F}_{2^m}$, along with the point of infinity $\mathcal{O}$, neutral element of the addition law. The main interest of Koblitz curves is that point doublings can be replaced by

the efficiently computable Frobenius automorphism $\Phi : (x, y) \mapsto (x^2, y^2)$. Since the quadratic equation $(x^4, y^4) + 2(x, y) = \mu(x^2, y^2)$ where $\mu = (-1)^{1-a}$ holds for all points, the Frobenius map can be regarded as $\tau = (\mu + \sqrt{-7})/2$, solution of the equation $\Phi^2 + 2 = \mu\Phi$. The Lucas sequence U_w is useful to compute with τ:

$$U_0 = 0, U_1 = 1 \text{ and } U_{w+1} = \mu U_w - 2U_{w-1} \text{ for } w \geq 1. \tag{1}$$

The approach for fast computations over Koblitz curves is to convert a scalar d to a radix-τ expansion such as $d = \sum_{i=0}^{j} d_i \tau^i,\ d_i \in \{0, \pm 1\}$. However, in order to fully take advantage of the Frobenius map, the τ expansion must be sparse and short. In [Sol00], Solinas proposed two efficient algorithms to satisfy these properties: partial reduction modulo $\delta = (\tau^m - 1)/(\tau - 1)$ and radix-τ NAF recoding. To generate a width w radix-τ NAF expansion (TNAF_w), one can use the following map:

$$\Phi_w : u_0 + u_1 \cdot \tau \in \mathbb{Z}[\tau] \mapsto u_0 + u_1 \cdot t_w \bmod 2^w \in \mathbb{Z}/2^w\mathbb{Z}, \tag{2}$$

where $t_w = 2U_{w-1}U_w^{-1} \bmod 2^w$ and the notation mods 2^w refers to the signed representatives modulo 2^w. More precisely, the digits of the TNAF_w are computed by iterating the following procedure [Sol00]. (1) If $\rho \in \mathbb{Z}[\tau]$ is divisible by τ, set $u \leftarrow 0$; otherwise $u \leftarrow \Phi_w(\rho)$. (2) Set $\rho \leftarrow (\rho - u)/\tau$, append u to the TNAF_w and iterate. If the scalar is first reduced modulo δ, the length of the TNAF_w is at most $m + a$ [Sol00]. Since $u \in \{\pm 1, \pm 3, \ldots, \pm(2^{w-1} - 1)\}$, only $2^{w-2} - 1$ non-trivial points must be pre-computed, with $2^{w-2} - 1$ point additions and one point doubling. Therefore, the total cost of the scalar multiplication using TNAF_w is on average:

$$\mathcal{C}_{NAF} = (m + a)\,\mathrm{ECFRB} + \left(\frac{m + a}{w + 1} + 2^{w-2} - 1\right)\mathrm{ECADD} + \mathrm{ECDBL} \tag{3}$$

where ECFRB, ECADD and ECDBL stand for the computational cost of the Frobenius map, point additions and point doublings, respectively.

2.2 Side Channel Attacks on Koblitz Curves

Side channel attacks are a serious threat for light embedded devices running cryptographic applications, which leak critical information through side channels, like timings or power consumption [Koc96, KJJ99]. One can classify side channel attacks relying on power analysis into two categories [KJJ99]. The first class of attacks is called simple power analysis (SPA): in this situation, the attacker attempts to reveal the secret parameter with one single power trace. The second class is called differential power analysis (DPA): the attacker is allowed to gather several power traces and analyzes them with the help of a statistical tool. In general, SPA on ECC utilizes the fact that point additions and doublings have different implementations, leading to different power traces [Cor99]. By recognizing the operation chain from the power consumption, the attacker can reveal the secret information. To implement Koblitz curves, a polynomial basis is generally preferred. In this case, point doublings are replaced by the Frobenius map,

whose computational cost is small but not negligible [HHM01]. In other words, it is realistic to expect that the computation of one single Frobenius map can be detected within one power trace: in this case, SPA is straight-forwardly applicable. In [Cor99], Coron extended DPA to elliptic curve cryptosystems. Since Koblitz curves offer no intrinsic DPA resistance, when necessary, the standard DPA countermeasures have to be deployed [Cor99, JT01a].

3 Proposed SPA-resistant Techniques

In the following, we describe two methods to protect the scalar multiplication on Koblitz curves against side-channel attacks.

3.1 SPA-resistant TNAF$_w$

Recall that to generate the TNAF$_w$, one computes representatives of congruent classes modulo τ^w with the map $\Phi_w : d_0 + d_1 \cdot \tau \in \mathbb{Z}[\tau] \mapsto d_0 + d_1 \cdot t_w \bmod 2^w$. Then, $d - \Phi_w(d)$ is divisible by τ^w. However, we aim at generating a fixed pattern: we look for a new set of representatives modulo τ^w, which verify that $d - u$ is divisible by τ^w, but additionally that $d - u$ is *not* divisible by τ^{w+1}.

Proposition 1. *Let the map Ψ_w be defined as:*

$$\Psi_w : d_0 + d_1 \cdot \tau \in \mathbb{Z}[\tau] \mapsto (d_0 + d_1 \cdot t_{w+1} \bmod 2^{w+1}) - 2^w \in \mathbb{Z}/2^w\mathbb{Z}. \quad (4)$$

For any $d \in \mathbb{Z}[\tau]$, $d - \Psi_w(d)$ is divisible by τ^w but not by τ^{w+1}.

Proof. Recall that d is divisible by τ^w iff $\Phi_w(d) = 0$ [Sol00]. We first prove that $d - \Psi_w(d)$ is divisible by τ^w, in other words, $\Phi_w(d - \Psi_w(d)) = 0$. Let $d = d_0 + d_1 \cdot \tau \in \mathbb{Z}[\tau]$. Then, $\Phi_w\left(d - \Psi_w(d)\right) = d_1\left(t_w - t_{w+1}\right) \bmod 2^w$. Moreover, $t_w - t_{w+1} = (U_w^2 - \mu U_{w-1}U_w + 2U_{w-1}^2)/(U_w U_{w+1})^{-1} \bmod 2^w$. We can write that $U_w^2 - \mu U_{w-1}U_w + 2U_{w-1}^2 = |U_w - U_{w-1} \cdot \tau|$ and $U_w - U_{w-1} \cdot \tau = \tau^{w-1}$. Because $\left|\tau^{w-1}\right| = 2^{w-1}$, $t_w - t_{w+1} = -2^w U_w^{-1} U_{w+1}^{-1} = 0 \bmod 2^w$. It follows that $\Phi_w\left(d - \Psi_w(d)\right) = 0$. Besides, it is trivial that: $\Phi_{w+1}(d - \Psi_w(d)) = -2^w \neq 0 \bmod 2^{w+1}$. Then, $\Phi_{w+1}(d - \Psi_w(d)) \neq 0$ and $d - \Psi_w(d)$ is not divisible by τ^{w+1}. □

As a consequence, Ψ_w can be used to generate SPA-resistant TNAF$_w$ expansions. Note that the input of the recoding algorithm is $\rho = r_0 + r_1 \cdot \tau \in \mathbb{Z}[\tau]$, corresponding to an integer d which was first reduced modulo δ in order to generate a shorter recoding. More precisely, the SPA-resistant TNAF$_w$ recoding has $\lceil (m+a)/w \rceil$ non-zero digits; with the original TNAF$_w$, thanks to the reduction modulo δ, the recoded scalar has $m + a$ digits.

Based on the recoding computed by Algorithm 1, Algorithm 2 protects the scalar multiplication on Koblitz curves against SPA.

Proposition 2 (τ-SPA resistance). *The ability to distinguish individual τ multiplications and point additions in power traces confers no advantage for finding the scalar in Algorithm 2.*

Algorithm 1: Conversion to SPA-resistant TNAF_w

INPUT: $\rho = r_0 + r_1 \cdot \tau \in \mathbb{Z}[\tau]$ with r_0 odd, width w;
OUTPUT: $(d_l^{(w)}, \ldots d_0^{(w)}) = \mathrm{TNAF}_w(\rho)$;

1. $c_0 \leftarrow r_0$; $c_1 \leftarrow r_1$, $l \leftarrow 0$;
2. **while** $c_1 \neq 0$ **or** $|c_0| > 2^w$ **do**
   (a) $u \leftarrow \Psi_w(c_0 + c_1 \cdot \tau)$; $d_l^{(w)} \leftarrow u$; $c_0 \leftarrow c_0 - u$; $l \leftarrow l + 1$;
   (b) **for** j **from** 1 **to** w **do** $(c_0, c_1) \leftarrow (c_1 + \mu c_0/2, -c_0/2)$;
3. $d_l^{(w)} \leftarrow \Psi_w(c_0 + c_1 \cdot \tau)$;
4. **return** $(d_l^{(w)}, \ldots d_0^{(0)})$;

Algorithm 2: Scalar multiplication using SPA resistant TNAF_w

INPUT: a scalar d, base point P, width w;
OUTPUT: multiplied point $Q = dP$;

1. pre-compute $3P, 5P, \ldots, (2^w - 1)P$; $\rho \leftarrow d \bmod \delta$;
2. **if** ρ is divisible by τ **then** $\rho' \leftarrow \rho + 1$; **else** $\rho' \leftarrow \rho + \tau$;
3. compute $(d_l^{(w)}, \ldots d_0^{(w)})$, from ρ' with Algorithm 1;
4. $Q \leftarrow \mathcal{O}$;
5. **for** i **from** l **down to** 0 **do**
   (a) **for** j **from** 1 **to** w **do** $Q \leftarrow \tau Q$;
   (b) **if** $d_i^{(w)} > 0$ **then** $Q \leftarrow Q + d_i^{(w)} P$; **else** $Q \leftarrow Q - (-d_i^{(w)})P$;
6. **if** ρ is divisible by τ **then** $Q \leftarrow Q - P$; **else** $Q \leftarrow Q - \tau P$;
7. **return** Q;

Proof. Since the main loop (i.e. Step 5) is τ-SPA resistant because the scalar representation has a fixed pattern, the only concern is that the scalar multiplication scheme requires an "odd" input ρ (i.e. indivisible by τ). If ρ is divisible by τ, one can add 1 to $\rho = d \bmod \delta$, and adjust the result of the scalar multiplication by subtracting P. To prevent attackers from distinguishing the cases where ρ is odd or even, one can always add τ to ρ if it is odd, and subtract τP from the result of the scalar multiplication $Q = (\rho + \tau)P$. □

It follows that our scheme is τ-SPA resistant, that is, assuming strong abilities for the attacker, who is able to distinguish individual τ multiplications. In fact, in our attack model, the information arisen from SPA is of *no* use for attackers. In terms of computational cost, Algorithm 2 computes $\lceil (m + a)/w \rceil$ point additions in the main loop. The pre-computation of $2^{w-1} - 1$ points requires $2^{w-1} - 1$ point additions and 1 point doubling. Making the cases ρ odd and ρ even indistinguishable by SPA requires one more point addition. Then, the total computational cost of Algorithm 2 is:

$$\mathcal{C}_{STNAF_w} = (m + a)\mathrm{ECFRB} + \left(2^{w-1} + \left\lceil \frac{m + a}{w} \right\rceil\right)\mathrm{ECADD} + \mathrm{ECDBL}. \quad (5)$$

Note that the computational cost of ECFRB is small but not negligible.

3.2 Zero-Free Representation

The recoding of the SPA-resistant TNAF_w utilizes a right-to-left strategy: the scalar must be first recoded and stored in its new representation, wasting memory. However, we can partially fix the problem, using a two-round recoding. In the first round, the scalar is converted to a zero-free representation, using a right-to-left approach. In the second round, a windowing technique is applied in order to take advantage of pre-computed points and reduce computational costs. We will see that in many practical situations, it is not even necessary to compute the first round in the runtime.

First, we explain how to convert the scalar to a zero-free representation. The set of digits of the SPA-resistant TNAF_w consists of $\pm 1, \pm 3, \ldots, \pm(2^w - 1)$. Especially, when $w = 1$, the representation uses only two digits, namely 1 and -1, and the scalar multiplication is carried out without pre-computations. Unfortunately, since zeros are absent from the new representation, about m point additions are necessary to compute the scalar multiplication, whereas the original TNAF needed only $m/3$ point additions. On the other hand, this zero-free representation has several interesting properties. First, it is SPA resistant. Second, since the only possible digits are 1 and -1, one can easily store the recoded scalar in memory by representing the digit 1 with the bit 0, and -1 with the bit 1, for instance. Additionally, a random representation can be easily generated from a random bit sequence.

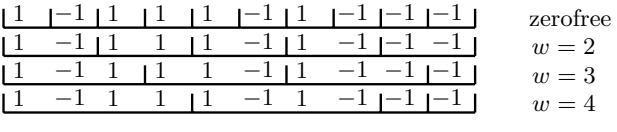


Fig. 1. Windowing technique on zero-free representation

It remains to reduce computational costs. In fact, it is easy to apply a windowing technique to the zero-free representation, while preserving the original SPA-resistance: simply divide the scalar into windows of w consecutive digits, from left to right, and if the right-most window is not full, treat each digit independently in a window $w = 1$ [3] , as shown in Fig. 1. Then, the computational cost of the scalar multiplication (excluding pre-computations) is:

$$\mathcal{C}_{ZF_w} = (m + a)\mathrm{ECFRB} + \left(\left\lfloor \frac{m+a}{w} \right\rfloor + m + a - w \left\lfloor \frac{m+a}{w} \right\rfloor \right) \mathrm{ECADD}. \quad (6)$$

In each (full) window, the possible digits are $\pm\tau^{w-1} \pm \ldots \pm \tau \pm 1$, and ± 1 in the rightmost windows. Therefore, it suffices to pre-compute the points $\tau^{w-1}P \pm$

[3] The number of windows $w = 1$ does not depend on the scalar: this information cannot be used to mount attacks.

$\ldots \pm \tau P \pm P$ and compute point additions or subtractions depending on the sign of the leftmost digit in each windows. Interestingly, the points can be pre-computed using simultaneous additions and subtractions: the computational cost of the simultaneous computation of $P+Q$ and $P-Q$ is ECAS $= 4M+I$ instead of $4M+2I$, where M and I denote the cost of field multiplications and inversions, respectively. In a naive approach based on tree exploration, one computes P, then $\tau P \pm P$, after that $\tau^2 P \pm \tau P \pm P$, and so on. This technique requires $2^{w-1}-1$ simultaneous additions-subtractions, which is slower than the $2^{w-1}-1$ additions in the case of the SPA-resistant TNAF$_w$. However, in a more efficient approach, one can re-use partial results in order to compute the next steps. Figure 1 illustrates this pre-computation technique: the partial results at a given depth in the tree are re-combined using simultaneous additions-subtractions in order to expand the tree.

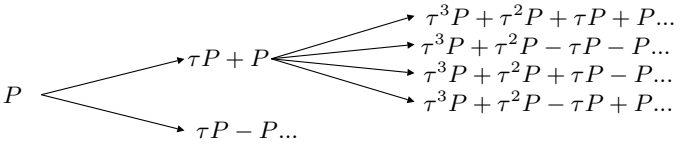


Fig. 2. Efficient pre-computations in the zero-free method

With this pre-computation approach, at the ith level of the tree (starting from P, level 0), there are 2^{2^i-1} points, and 2^{2^i-2} simultaneous point additions-subtractions are required to compute them from level $i-1$. As a consequence, the computational cost of pre-computations at level i, corresponding to a pre-computed table with width $w = 2^i$ is $1 + \sum_{j=1}^{i} 2^{2^j-2}$ECAS. This is expected to be faster than in the case of the SPA-resistant TNAF$_w$. In the general case, using a hybrid approach which combines levels of the tree at different depths (for example, $w = 5$ can be computed using $w = 2$ and $w = 3$), the pre-computations are indeed faster than those of the SPA-resistant TNAF$_w$.

4 Applications, Improvements and Comparisons

We describe applications of our techniques in the frame of standard ECC protocols, and show how they compare to other schemes.

4.1 Efficient Representation for Fixed Scalar

When the scalar is fixed (for instance the secret key in EC-ElGamal), part of the recoding work can be pre-processed off-line, that is, once for all during the initialization of the smartcard. In particular, instead of storing the integer value of the secret key, one can reduce it modulo δ and store the corresponding element of the quadratic field $c_0 + c_1\tau \in \mathbb{Z}[\tau]$, or even recode it off-line to an

SPA-resistant representation [4]. Especially, we can compute the zero-free representation (that is, the SPA-resistant TNAF_1) off-line and choose the window size w in the runtime. Storing the recoded scalar takes only $m + a$ bits.

In Algorithm 3, we consider how to use the windowed zero-free method to compute the scalar multiplication with fixed scalar. Since the zero-free method is based on a fixed-pattern recoding, the scalar multiplication is SPA-resistant. However, to generate the zero-free representation, the SPA-resistant TNAF_1 needs an input $c_0 + c_1\tau$ where c_0 is odd (i.e. the scalar is not divisible by τ). Therefore, after reducing the scalar, we check whether c_0 is divisible by τ or not; if this is the case, we add 1 and set an additional parity bit d_p to 0, and if not, we add τ and set d_p to 1. After computing the scalar multiplication, we adjust the result by subtracting P or τP depending on the parity bit. Since the first (right-to-left) zero-free recoding round has been eliminated, the second (left-to-right) windowing round can be carried out on-the-fly during the scalar multiplication, with no further memory consumption.

Algorithm 3: Computing $Q = dP$ for fixed d

INPUT: Base point P, zero-free recoding $(d_{m+a-1} \dots d_0)$, parity bit d_p, width w;
OUTPUT: $Q = dP$;

1. pre-compute $\tau^{w-1}P \pm \tau^{w-2}P \pm \dots \pm \tau P \pm P$;
2. $Q \leftarrow \mathcal{O}$; $i \leftarrow m + a - 1$;
3. **while** $i \geq w$ **do**
   (a) **for** j **from** 0 **to** $w - 1$ **do** $Q \leftarrow \tau Q$**;**
   (b) $Q \leftarrow Q + (-1)^{d_i}\tau^{w-1}P + (-1)^{d_{i-1}}\tau^{w-2}P + \dots + (-1)^{d_{i-w+1}}P$; $i \leftarrow i - w$;
4. **for** j **from** 0 **to** i **do** $Q \leftarrow \tau Q + (-1)^{d_{i-j}}P$;
5. **if** $d_p = 0$ **then** $Q \leftarrow Q - P$ **else** $Q \leftarrow Q - \tau P$**; return** Q

4.2 Random SPA-resistant Representation for Secret Ephemerals

In many cryptographic protocols, a random ephemeral is needed. Since the knowledge of the ephemeral generally allows to recover the secret key, it is important to protect scalar multiplications with the ephemeral against SPA. One can always generate a random *integer* multiplier, reduce it modulo δ, convert it to an SPA-resistant representation and finally perform the scalar multiplication. However, it would be preferable to generate a random SPA-resistant representation instead of a random integer, and even better, to generate the successive coefficients of the representation on-the-fly, from left to right. Additionally, since the integer value of the random multiplier is often needed along with the multiplied point, we should compute this integer value without excessive overhead.

[4] In this case, DPA countermeasures based on scalar blinding are not available, since the recoding is fixed. However, one can still deploy other types of countermeasures, such as randomized projective coordinates or randomized base point [Cor99].

In the following, we present a method for generating a random windowed zero-free representation along with its integer value. Our technique takes w random bits and generates the coefficients of the windowed zero-free representation one after the other, on-the-fly and from left to right. To compute the integer value of the multiplier, we simply reverse Algorithm 1. This method employs only additions, shifts, and one final integer multiplication with τ to compute the integer value of the zero-free representation. Since Algorithm 1 is right-to-left, its reversed counterpart works left-to-right, and can also be executed on the fly.

Algorithm 4: Generating a random point $Q = kP$

INPUT: Base point P;
OUTPUT: Random integer k and the corresponding point $Q = kP$;

1. pre-compute $\tau^{w-1}P \pm \tau^{w-2}P \pm \ldots \pm \tau P \pm P$;
2. $Q \leftarrow \mathcal{O}$; $c_0 \leftarrow 0$; $c_1 \leftarrow 0$; $i \leftarrow m + a - 1$;
3. **while** $i \geq w$
   (a) pick w random bits $d_i d_{i-1} \ldots d_{i-w+1}$;
   (b) **for** j **from** 0 **to** $w - 1$ **do** $Q \leftarrow \tau Q$; $(c_0, c_1) \leftarrow (-2c_1 + (-1)^{d_{i-j}}, c_0 + \mu c_1)$;
   (c) $Q \leftarrow Q + (-1)^{d_i}\tau^{w-1}P + (-1)^{d_{i-1}}\tau^{w-2}P + \ldots + (-1)^{d_{i-w+1}}P$; $i \leftarrow i - w$;
4. pick $i + 1$ random bits $d_i d_{i-1} \ldots d_0$ and a parity bit d_p;
5. **for** j **from** 0 **to** i **do** $Q \leftarrow \tau Q + (-1)^{d_{i-j}}P$; $(c_0, c_1) \leftarrow (-2c_1 + (-1)^{d_{i-j}}, c_0 + \mu c_1)$;
6. **if** $d_p = 0$ **then** $Q \leftarrow Q - P$; $c_0 \leftarrow c_0 - 1$; **else** $Q \leftarrow Q - \tau P$; $c_1 \leftarrow c_1 - 1$;
7. $k \leftarrow c_0 + c_1 \cdot \tau \bmod \#\mathcal{E}_a$; **return** k and Q;

Proposition 3. *The distribution of random zero-free chains is close to the uniform distribution. In fact, its statistical distance* $\Delta(g) = \sum_{i=0}^{\#\mathcal{E}_a - 1} |P(g = i) - \frac{1}{\#\mathcal{E}_a}|$ *to the uniform distribution is bounded by:*

$$\Delta(g) \leq \frac{1}{2}\sqrt{\sum_{i=1}^{\#\mathcal{E}_a - 1} \prod_{j=0}^{m+a-1} |\cos(\frac{2\pi \cdot i}{\#\mathcal{E}_a}(U_j\tau - 2U_{j-1}))|^2}. \tag{7}$$

Proof. To bound the statistical distance, we use the fact that for any random variable X, we have: $\Delta(X) \leq \frac{1}{2}\sqrt{\sum_{i=1}^{\#\mathcal{E}_a - 1} |E[e^{\frac{2\pi \Im i X}{\#\mathcal{E}_a}}]|^2}$, where $\Im$ denotes the complex $\sqrt{-1}$ and i is used as index. See [JT01b] for a proof. Obviously, the distribution of windowed zero-free τ expansions does not depend on the width w; more precisely, we only have to study the distribution of the "binary" zero-free representation, that is, the distribution of the SPA-resistant TNAF_1. Since all bits of the SPA-resistant TNAF_1 expansion $\sum_{j=0}^{m+a-1}(-1)^{d_j}\tau^j$ are chosen independently, we have $|E[e^{\frac{2\pi \Im i}{\#\mathcal{E}_a}(\sum_{j=0}^{m+a-1}(-1)^{d_j}\tau^j)}]| = \prod_{j=0}^{m+a-1} |E[e^{\frac{2\pi \Im \cdot i}{\#\mathcal{E}_a}(-1)^{d_j}\cdot\tau^j}]|$. Additionally, $|E[e^{\frac{2\pi \Im i}{\#\mathcal{E}_a}(-1)^{d_j}\cdot\tau^j}]| = |\frac{e^{\frac{2\pi \Im i}{\#\mathcal{E}_a}\tau^j} + e^{-\frac{2\pi \Im i}{\#\mathcal{E}_a}\tau^j}}{2}| = |\cos(\frac{2\pi \cdot i \cdot \tau^j}{\#\mathcal{E}_a})|$, and $\tau^j = U_j\tau - 2U_{j-1}$, which proves the result. □

Conjecture 1. The statistical distance of g to the uniform distribution is bounded by $\Delta(g) \leq 2^{-m/5}$.

We computed approximations of the sum by using a smaller pool of random values of $i \in \{1, 2, \ldots, \#\mathcal{E}_a - 1\}$. According to our numerical experimentations for several bitlengths m and 8192 random values of i, the experimental statistical distance is indeed smaller than $2^{-m/5}$: writing $\Delta(g) \approx 2^{-m/\alpha}$, the experimental value of α seems to decrease as m grows, which tends to show that our conjecture is reasonable.

Table 1. Approximated values of α, where $\Delta(g) \leq 2^{-m/\alpha}$ for several bitlengths m

m	109	113	131	163	233	239	277	283	359	409	571
α	4.0	4.3	4.2	3.9	3.6	3.5	3.5	3.6	3.3	3.2	3.0

4.3 Comparison with Known Methods

Fast scalar multiplication on Koblitz curves. The fastest scalar multiplication on Koblitz curves is the TNAF_w [HHM01]; although our proposed techniques do not intend to compete with the (insecure) TNAF_w, Table 2 put in evidence the overhead introduced to achieve SPA-resistance. Roughly speaking, for the same memory consumption, the TNAF_w utilizes a window size $w+2$ where our techniques have w. This is the price to pay to achieve SPA resistance.

Table 2. Compared computational costs ($m = 163$)

		$w=1$	$w=2$	$w=3$	$w=4$	$w=5$
insecure TNAF_w	memory (bytes)	-	-	42	126	294
[Sol00, HHM01]	comp. cost (M)	-	437	338	296	301
SPA-res. TNAF_w	memory (bytes)	-	42	126	294	630
(Algorithms 1, 2)	comp. cost (M)	1312	680	488	424	456
zero-free method	memory (bytes)	-	84	168	336	672
(Algorithms 3, 4)	comp. cost (M)	1312	670	490	398	442

SPA-resistant methods. SPA countermeasures on binary curves, such as the Montgomery ladder [LD99], are also applicable to Koblitz curves. However, the computational advantage introduced by τ multiplications is lost in that case. The Montgomery ladder requires only 978 multiplications for $m = 163$, protects

against SPA with no pre-computed points. On the one hand, when memory resources are extremely scarce, the Montgomery ladder performs better than our methods. On the other hand, when some memory is available for pre-computed tables, as soon as $w = 2$, our methods beat the Montgomery ladder.

Some SPA countermeasures using τ expansions were proposed in [Has01]. However, these countermeasures are not optimal in terms of memory and computational cost: our methods are more efficient. In [PSL03], the SPA-resistance properties of the TNAF_w using a change-of-basis strategy is pointed out. In a normal basis, the operation $\tau^w P$ is a simple cyclic shift of the coordinates of P. Thus, they claim the time for computing $\tau^w P$ is independent from w, and therefore, the *position* of nonzero digits is concealed in the TNAF_w. However, it is controversial whether the cyclic shift has a static implementation in software, without using dummy operations. Second, the method leaks the number of nonzero digits of the TNAF_w representation. Even though this problem can be partially fixed by introducing additional operations, some information is still leaked. Third, they did not discuss how to efficiently store or randomly generate the TNAF_w. Finally, even though the method seems to have the same computational cost as the original TNAF_w, there are several drawbacks which may practically slow down the scheme. If dummy operations are used in order to have a static implementation of cyclic shifts and change-of-bases, the latter operations will run much slower. Additionally, since point additions are inserted to conceal the number of nonzero digits, it is not clear what the average cost of the countermeasure is.

Secret ephemerals and compact encoding. Two methods for generating ephemerals on Koblitz curves have been proposed. In [CMT01], the Frobenius is utilized to increase the entropy of standard generators. While this idea leads to a very efficient scalar multiplication, our scheme has several advantages compared to [CMT01]. First, our scheme is SPA-resistant, whereas side channel attacks are not discussed in [CMT01]. Second, we can use the same (offline) pre-computation table for the known point scalar multiplications in the signature generation and verification of EC-DSA, which is impossible in the case of the generator proposed in [CMT01]. In [JT01b], a compact encoding of the NAF_2 is proposed; since their encoding can be randomly generated, they also discuss how to obtain random TNAF_2. Unfortunately, their idea seems only applicable for a width $w = 2$. On Koblitz curves, the computation cost of the scalar multiplication can be drastically reduced with window methods, therefore, this is an important drawback of their method. Besides, the straight-forward implementation of the TNAF_2 is vulnerable to SPA.

5 Conclusion

We presented two new scalar multiplication methods on Koblitz curves: the SPA-resistant TNAF_w and the windowed zero-free method. The first technique extends the mechanisms to the SPA-resistant NAF_w to the arithmetic of Koblitz

curves, whereas the second technique is specifically designed for left-to-right computations in some practical situations. Both of our schemes are efficient, SPA-resistant, allow to flexibly choose how much memory is used in order to speed-up the computations, and free from dummy operations. Therefore, we claim that our schemes achieve a high security level for an acceptable overhead compared to insecure methods. Additionally, we proposed practical applications, such as fixed-scalar and random ephemeral multiplication schemes. In this cases, the windowed zero-free method can be optimized by introducing a full left-to-right and on-the-fly recoding.

References

[CCJ04] Chevallier-Mames, B., Ciet, M., Joye, M.: Low-Cost Solutions for Preventing Simple Side-Channel Analysis: Side-Channel Atomicity. *IEEE Trans. Comput.*, **53**(6) (2004) 760–768.

[Cor99] Coron, J.-S.: Resistance against differential power analysis for elliptic curve cryptosystems. Proc. *CHES'99*, LNCS **1717** (1999) 292–302.

[CMT01] Coron, J.-S., M'Raïhi, D., Tymen, C.: Fast generation of pairs $(k, [k]P)$ for Koblitz elliptic curves. Proc. *SAC'01*, LNCS **2259** (2001) 151–164.

[Has01] Hasan, A.: Power analysis attacks and algorithmic approaches to their countermeasures for Koblitz curve cryptosystems. *IEEE Trans. Comput.*, **50**(10) (2001) 1071–1083.

[HHM01] Hankerson, D., López, J., Menezes, A.: Software implementation of elliptic curve cryptography over binary fields. Proc. *CHES'00*, LNCS **1965** (2001) 1–24.

[JT01a] Joye, M., Tymen, C.: Protections against differential analysis for elliptic curve cryptography. Proc. *CHES'01*, LNCS **2162** (2001) 377–390.

[JT01b] Joye, M., Tymen, C.: Compact encoding of non-adjacent forms with applications to elliptic curve cryptography. Proc. *PKC'01*, LNCS **1992** (2001) 353–364.

[KJJ99] Kocher, P., Jaffe, J., Jun, B.: Differential power analysis. Proc. *Crypto'99*, LNCS **1666** (1999) 388–397.

[Kob91] Koblitz, N.: CM-curves with good cryptographic properties. Proc. *Crypto'91*, LNCS **576** (1992) 279–287.

[Koc96] Kocher, P.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. Proc. *Crypto'96*, LNCS **1109** (1996) 104–113

[LD99] López, J., Dahab. R.: Fast multiplication on elliptic curves over GF(2^m) without precomputations. Proc. *CHES'99*, LNCS **1717** (1999) 316–327.

[OT03] Okeya, K., Takagi, T.: The width-w NAF method provides small memory and fast elliptic scalar multiplications secure against side channel attacks. Proc. *CT-RSA'03*, LNCS **2612** (2003) 328–342.

[PSL03] Park, D. J., Sim, S. G., Lee, P. J.: Fast scalar multiplication method using change-of-basis matrix to prevent power analysis attacks on Koblitz curves. Proc. *WISA 2003*, LNCS **2908** (2003) 474–488.

[Sol00] Solinas, J.: Efficient arithmetic on Koblitz curves. *Designs, Codes, and Cryptography*, **19**(2–3) (2000) 195–249.

[YJ00] Yen, S.-M., Joye, M.: Checking before output may not be enough against fault-based cryptanalysis. *IEEE Trans. Comput.*, **49**(9), (2000) 967–970.

SIFA: A Tool for Evaluation of High-Grade Security Devices

Tim McComb and Luke Wildman

School of Information Technology and Electrical Engineering
The University of Queensland, 4072, Australia
{tjm, luke}@itee.uq.edu.au

Abstract. We describe a tool for analysing information flow in security hardware. It identifies both sub-circuits critical to the preservation of security as well as the potential for information flow due to hardware failure. The tool allows for the composition of both logical and physical views of circuit designs. An example based on a cryptographic device is provided.

1 Introduction

Secure electronic communications networks such as those employed in government or the military may involve the use of *domain separation* devices to control the flow of information between high- and low-security domains. Cryptographic devices, data-diodes, context filters and peripheral sharing devices are all examples of such devices. A principal concern in the design of such devices is that they must preserve data security in the face of hardware faults and security threats.

The *Common Criteria for Information Technology Security Evaluation* [6] provides high-level guidance for describing the specification and evaluation of such security requirements. However, it gives very little detailed guidance on techniques useful for analysing the designs of the devices mentioned above, particularly at higher levels of security.

Evaluating devices for potential security failures is difficult because it involves a detailed analysis of every possible way that information may flow through the device in both intended and unintended modes of operation, and additionally in the presence of hardware failures. It is therefore highly labour intensive.

In this paper we present tool support for the evaluation of security hardware. The tool implements two evaluation techniques: identification of the *security critical region* [7], and failure analysis [8]. The evaluation is based on schematics of the security hardware. However, schematic diagrams invariably focus on some details and ignore others. For instance, a schematic generally focuses on the logical connections in a circuit but may ignore physical associations such as the fact that certain components may all reside on the same chip. However, the interactions between these device characteristics cannot be ignored when evaluating security devices at the highest level. We present a new approach that supports multiple views [5] of a device. In particular, logical and physical views

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 230–241, 2005.

describing the device at different levels of detail may be composed and analysed together.

1.1 Overview

Section 2 introduces the Secure Information Flow Analyser (SIFA) and discusses the tool's method for composing different views of the model to allow for a unified analysis. Section 3 goes further to illustrate the view-based approach, using a model of an experimental cryptographic device as a running example. Following that we discuss the use of SIFA for evaluation. Specifically, Section 4 illustrates the evaluation of the device for systematic design faults, and Section 5 discusses the use of SIFA to evaluate the device for operational failures. A summary and a discussion of further and related work is presented in Section 6.

2 SIFA

The Secure Information Flow Analyser (SIFA) enables models of electronic circuits to be built and analysed. Electronic circuits are modelled in SIFA using block diagrams consisting of *blocks*, *ports*, and *arcs*. At the most basic level, each component in the schematic diagram for the device under analysis is represented by a block in the tool's graphical editor. These blocks have a number of ports which allow them to be connected to other blocks via arcs. Conceptually, arcs represent information flow, which normally corresponds to the actual wiring in the hardware circuit.

Nesting may also be used for describing devices at different levels of abstraction. That is, a block may represent either a single atomic component in the device for the purpose of analysing the effect of complicit failures relating to the whole block, or it may represent a collection of components for the purpose of analysing the effects arising from independent failures of the sub-components.

A distinguishing feature of SIFA is the ability to combine multiple views of a circuit. In particular, this feature allows both logical and physical views to be analysed concurrently. Multiple views are provided by treating ports that appear in different diagrams, but share a common name, as the same port. In fact this mechanism allows diagrams to be structured both vertically (giving rise to hierarchical block diagrams) and horizontally (giving rise to alternate views). Hierarchical and view-based structuring can be interleaved to provide multiple views of nested components. This is particularly convenient as each nested or alternate block diagram may be saved or loaded separately (as XML) allowing component views/hierarchies to be reused.

SIFA supports analysis of the circuit in two ways. First, it may analyse the block connectivity of the device in order to determine the *Security Critical Region* (SCR) [7]. That is, given the network comprised of components and connections the tool can identify components that could not possibly contribute to information flow. Elimination of these components allows further analysis to be focused upon the remaining SCR.

Secondly, the tool can perform analysis [8] based on the different operating states of the device. That is, when given additional information about the information flow inside a component, both under normal operating circumstances as well as in the presence of failures, the tool can establish whether the device maintains security domain separation in all modes of operation. If the device does not maintain domain separation the tool reports the operating modes of each component that contributed to the security violation. The method is able to detect whether a security failure may arise as a consequence of either an individual component fault or a global fault, and generates the pattern of faults leading to the failure.

3 Example

To illustrate SIFA and the issues involved with information flow analysis, an experimental cryptographic device [4] will be used. The device encrypts plaintext received from a computer and sends the resulting encrypted text to a communications network. It also decrypts encrypted text received from the network and forwards the resulting plaintext to the computer. The device has several modes of encryption and decryption (including non-encrypting bypass modes in both directions) that are selectable by special control characters received from the computer and the network. The modes are indicated by LEDs on the front panel of the unit. A pair of such devices must be used in order to construct an encrypted channel over a network.

The cryptographic device features a fail-safe mechanism in the form of a redundant processor that is used to check the correct encryption of the input. If the device detects a problem it shuts itself down and illuminates a 'fault' LED. It also features a 'fault insertion' button and a 'fault reset' switch enabling the fault detection circuitry to be checked and reset.

The cryptographic device is modelled by a logical view and a physical view. In the logical view, the components correspond to functional units of the circuit. In the physical view, the components correspond to the actual hardware appearing on the circuit board. The logical and physical views are now presented in detail.

3.1 Logical View

A block diagram representing the logical view of the device [4] is displayed in the screenshot of the tool shown in Figure 1. The screenshot shows a top-level block diagram for the cryptographic device containing five major blocks[1].

The *Processors* block contains the redundant cryptographic microprocessors. The *Comparator* circuit contains the logic for comparing the output of the two redundant processors, and is responsible for enabling and disabling the output

[1] Much of the detail of the wiring for the LEDs, the RS232 serial data sockets, and the power circuitry, etc, has been removed from this diagram in order to focus on the security related components.

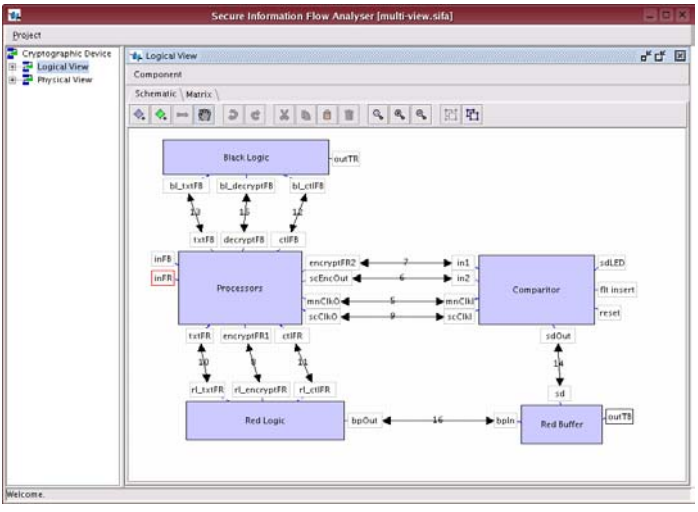


Fig. 1. Cryptographic device displayed by the tool

of the cryptographic chips through the *Red Buffer* gate[2]. The *Red Logic* block contains circuitry that controls whether encrypted information or plaintext is passed to the network, depending upon whether or not it is in bypass mode from high- to low-security domains. The *Black Logic* controls whether decrypted information or plaintext is passed to the host, depending upon whether or not it is in bypass mode from low- to high-security domains.

The cryptographic device has four external ports: two for connection to the high-security computer *inFR* and *outTR*, and two for connection to the low security network *inFB* and *outTB*. Ports *inFR* and *inFB* are on the central block labelled *Processors*, port *outTR* is associated with the *Black Logic* block, and port *outTB* is attached to the *Red Buffer*.

The *Processors* block is connected to the *Comparator* by four arcs, each representing a possible flow of information. The arcs from ports *encryptFR2* and *scEncOut* represent the encrypted outputs of the primary and secondary processors respectively. The arcs from ports *mnClkO* and *scClkO* represent clock signals from the processors to indicate to the comparator that the data has been set and is ready to be compared. In addition, the *Processors* block is connected to the Red and Black logic blocks by three arcs to each. These arcs represent the high- and low-security bypasses (from ports *txtFB* and *txtFR*), the encrypted text from the red side (*encryptFR1*), the decrypted text from the black side (*decryptFB*), and the control lines (*ctlFR*, *ctlFB*) enabling the bypass modes from the high- and low-security domains respectively. Finally, the *Comparator* block controls the output of information from high- to low-security domains via

[2] Red components lie on the security critical path from high- to low-security domains. Black components lie on the reverse path.

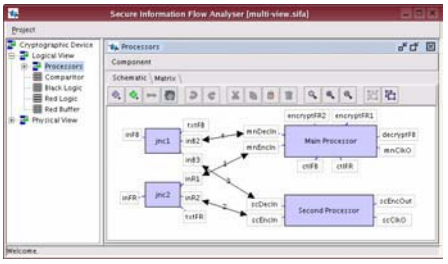


Fig. 2. Nested processors

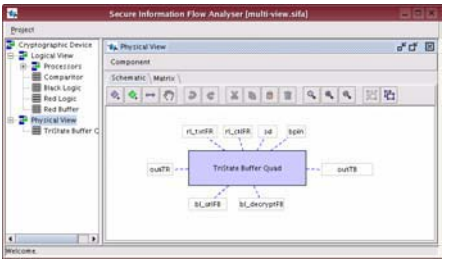


Fig. 3. Physical view

the shutdown line from the *sdOut* port to the *sd* port on the red buffer. There is no need for a similar mechanism for decryption from low- to high-security domains.

3.2 Hierarchical Block Diagrams

It is usually the case that a single block in a block diagram represents an aggregate of sub-components contained in nested block-diagrams. This hierarchical abstraction mechanism provides a powerful means for structuring large diagrams. For example, Figure 2 displays the block diagram of components nested in the *Processors* block.

To allow external ports associated with the parent component to be used as sources for internal components, the tool automatically unifies ports defined in the nested level with those defined in a superior level when they have the same name. For example, the *Processors* diagram (Figure 2) contains eight ports that are local to the block diagram: *inB2*, *inB3*, *inR1*, *inR2*, *mnDecIn*, *mnEncIn*, *scDecIn*, and *scEncIn*; these are connected internally. All the other ports are mentioned externally to the displayed block diagram (see Figure 1) for connection to external blocks. In particular, ports *txtFR* and *txtFB*, representing the bypass lines at junctions *jnc2* and *jnc1* respectively, are unified with the ports sharing the same name in the parent diagram.

3.3 Physical View

Electronic component manufacturers often supply multiple functional units on a single chip. However, schematic diagrams tend to represent each unit separately on the circuit digram ignoring physical relationships between them. For example, in the cryptographic device the *Red Logic*, *Black Logic* and *Red Buffer* blocks (referring to the logical view) all use tri-state buffer components provided by a single quad chip. Such packaging introduces interesting fault modes that may connect seemingly unrelated components. In particular they introduce the possibility of short circuits by component failure or poor soldering.

The physical view of the tri-state buffer quad chip is presented in Figure 3. The diagram contains a single block representing the quad chip, and eight ports

representing some of the pins connecting the chip to the surrounding circuitry. The connections represent *rl_txtFR* and *rl_ctlFR* from the *Red Logic* circuit which controls the output from the high-security to the low-security domains (including the bypass mode); *bpIn*, *sd* and *outTB* from the *Red Buffer* gate which controls the shutdown function; and *bl_ctlFB*, *bl_decryptFB* and *outTR* from the *Black Logic* circuit which controls the output from the low-security to the high-security domain. The diagram only represents a *partial* view of the physical componentry, as no other circuitry is represented in the diagram. Multi-view approaches generally allow individual views to be partial, allowing the particular aspect described by the view to be focused on independently. The impact of packaging all the tri-state buffers onto one chip will be discussed in Section 5.

4 Security Critical Region Analysis

Two analysis techniques are useful for directing the evaluation process and for prioritising components. First, the evaluator may eliminate components that do not need to be evaluated at all, because they do not lie on a critical information-flow path [7]. Secondly, the evaluator may choose components for detailed evaluation based upon whether or not they form crucial links between the high- and low-security domains [7].

SIFA uses well-known principles from graph-theory to automate these tasks, which are described below in detail. However, before analysis of the connectivity between the high- and low-security domains can proceed, the ports constituting the external interfaces to these domains need to be identified. Identification of these ports is achieved by tagging the respective ports as being either *red* or *black*. For example, in the top-level block diagram presented in Figure 1, the external source of high-security information to the cryptographic device is constituted by the red port *inFR*, and the external sink for low-security (or encrypted) information is constituted by the black port *outTB*.

4.1 Flow Analysis

Extraneous components that cannot possibly contribute to the maintenance of domain separation are readily identified in the tool using *flow* analysis [7]. Flow analysis identifies components that lie on a *directed walk* from the red source to the black sink. The set of components lying on directed walks from red to black is the largest set of components that needs to be evaluated in the worst case (assuming that the arc connectivity and arc directions are correct). The analysis is based on a simple walk enumeration algorithm.

Analysis is conducted at the level of abstraction corresponding to the block diagram currently displayed in the main panel. This allows different levels of abstraction to be analysed independently, which may be of assistance when analysing large and complex schematic diagrams.

For example, in the top-level block diagram presented in Figure 1 the analysis identifies that the *Black Logic* component does not lie on a directed walk from

the red source to the black sink. In the case that the RS232 sockets, power circuits, LEDs, and fault switches had been included in our original model, these components would have been eliminated at this stage as well.

This analysis assumes that the given arc directions correctly capture the possibility of information flow. It may be the case that, when beginning an evaluation, nothing is known about the direction of information flow. Under these circumstances all arcs should be assumed to be bi-directional and therefore no components can be eliminated. If, during more detailed analysis, additional assumptions about information flow may be justified, these should be added to the model so that flow analysis may be repeated.

4.2 Cutset Analysis

Component connectivity is also used to perform a *minimal cutsets* analysis [7]. The minimal cutsets of a network of components are the smallest sets (there may be more than one minimal cutset) of components that, if removed, would disconnect the network. If it can be proved that all the cutset members enforce domain separation, then the security of the total circuit is guaranteed. Therefore, by starting the in-depth analysis at these components, and iteratively eliminating components that do not enforce domain separation and regenerating the cutsets, one can quickly determine whether the circuit is secure. When calculating the cutsets, the blocks associated with red source and black sink ports – which are trivial cutsets in themselves – are removed from the analysis.

For example, the *Processors* block and the *Red Buffer* block displayed in Figure 1 are the source and sink components respectively. The result of the minimal cutsets analysis for the displayed diagram includes the single cutset: $\{Red\,Logic, Comparator\}$. The source component *Processors* and the sink component *Red Buffer* are also displayed as a reminder that these components are singleton cutsets of the circuit.

4.3 Iterative Analysis

The results of the flow analysis and cutset analysis indicate to the user the important components with respect to the failure analysis (Section 5). Failure analysis requires the entry of operational data for each component, and this can be a time-consuming process. The flow analysis and the minimal cutsets analysis are valuable tools because they can be used before this data entry stage to limit the amount of work that needs to be done and focus the evaluator's efforts. For instance, the three cutsets identified in the above example: $\{Processors\}$, $\{\ Red\ Buffer\ \}$ and $\{Red\,Logic, Comparator\}$ help prioritise the components for failure analysis; the *Processors* and *Red Buffer* may be analysed separately, but the *Red Logic* and the *Comparator* must be analysed together.

Following failure analysis, it is often useful to re-evaluate the topological information flow in light of the new analysis. For instance, as will be seen in Section 5, the *Red Buffer* is analysed first (being the easiest). It is found that this component does not prevent information flow unless it has been shutdown by

the *Comparator*. In addition, it is reasonable to assume that the shutdown signal coming from the *Comparator* does not carry red information itself. Therefore we can move the black information sink to the *Red Logic* block (*bpOut*) without changing the information flow structure. Running the flow analysis again shows that the *Comparator* may also be disregarded as it no longer lies on a path to the black data sink. This leaves the *Red Logic* and *Processors* blocks to consider.

5 Failure Analysis

The flow and minimal cutsets analyses only consider the external connectivity of the block diagram used to model the device, not the connectivity between ports within individual components. These techniques assumed total port connectivity across an individual component (the worst-case) in order to provide a direction for the user to concentrate the analysis. When performing a failure analysis, the best-case is assumed: that the components provide no connectivity across their ports unless it is otherwise specified.

In order to perform a failure analysis, every component identified by the flow analysis as participating in the information flow between high- and low-security domains, especially those mandatory for maintaining domain separation (as identified by the minimal cutsets analysis), needs to be annotated with a set of possible operation modes. These operation modes define connectivity, with respect to information flow, across each individual component, in both normal and failure modes. When the operational and static connectivity are composed, the tool offers the ability to analyse the entire system rooted below a particular diagram (combining all nested views) to identify aspects that impact upon information security. Failure analysis may be performed in two ways: as either a complete transitive analysis that calculates all possibilities of information flow between *every* port in a block diagram, or as a directed analysis which calculates only the possibilities for information flow from the port(s) tagged as the red data source to the port(s) tagged as the black data sink.

Details of the transitive and directed analyses are given in Sections 5.2 and 5.3 respectively. However, before proceeding to these sections, the details of the operational data and the method for entry are described.

5.1 Operational Data

The set of operation modes is normally divided into two disjoint subsets: those operation modes that describe connectivity under *normal* circumstances; and hypothetical operation modes that describe connectivity in the presence of *failure*. If desired, normal modes and failure modes may be analysed independently. To illustrate, consider the Red Buffer component with the set of operation modes *rbNorm* (normal), *rbShutdown* (normal with shutdown active), *sdStuckOff* (shutdown input stuck off), and *rbOpen* (short circuit). The Red Buffer has two modes that model its operation under normal circumstances, and a further two operation modes that model the component's connectivity under failure scenarios.

The intended operation of the Red Buffer is to allow information to flow from the input *bpIn* to the output *outTB* unless it is 'shutdown'. The signal to shutdown the buffer is received through the input *sd*.

Table 1. Adjacency matrix for Red Buffer

	outTB	**sd**	**bpIn**
outTB	*all*		
sd		*all*	
bpIn	{*rbNorm, sdStuckOff, rbOpen*}		*all*

The component's port connectivity is related to the operation modes through an adjacency matrix. For the Red Buffer example, the operation modes could be assigned as illustrated in Table 1 (where *all* is the set of all possible operation modes for that component).

Each cell in the matrix represents a directional information flow, from *row* to *column*, between ports attached to the component. The cells each contain a set of operation modes under which the connection is active. In this way, a single operation mode may specify a simple or elaborate pattern of connectivity across the component.

Every port is connected to itself under every operating state. These cells, forming the diagonal from left to right, cannot be modified by the user in SIFA as it does not make sense to disconnect a port from itself. The *outTB* port is the only port that allows information to flow out of the component. For input *sd*, output is possible only in operation mode *rbOpen*, whilst for input port *bpIn* this connection is made in all operation modes except for *rbShutDown*.

5.2 Transitive Analysis

A transitive analysis calculates a complete information flow matrix for a block diagram. The matrix describes all possible information flow from every port to every other port. Each cell of the fault matrix contains the combinations of operational modes that are necessary for information to flow. If no information flow is possible then this is indicated by the empty set ($\emptyset$) symbol.

For example, after invoking the transitive analysis on the top-level of the cryptographic device, the information flow matrix is populated with sets of operating modes. The cell corresponding to the connection from *inFR* to *outTB* contains the paths identified by the tool. When the contents of that cell are viewed, a tree is displayed that expands to reveal a subset of every path (from a topological perspective) connecting *inFR* to *outTB* (see Figure 4). Annotated on each step is either the component with the set of operation modes that allowed information to flow through it in that direction, or the name of the arc that joins the next component. It is a subset of all possible topological paths because some components may prohibit information flow between certain ports via their specification of operation modes. In Figure 4, three paths connecting the red source

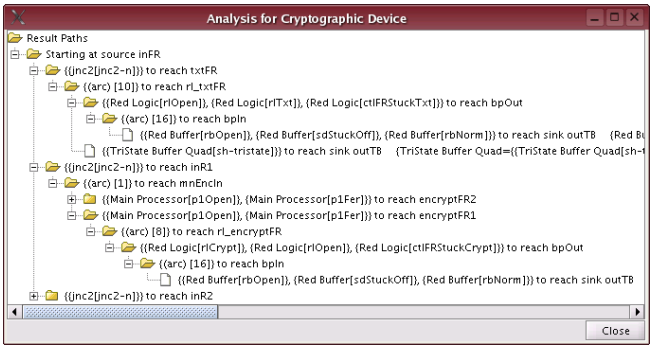


Fig. 4. Results for source inFR to sink outTB

to the black sink are expanded and shown. The third fully expanded path shows that, if the following conditions are met, there is a compromise in information security:

- *jnc2* is in operating mode *jnc2-n* (normal)
- *Main Processor* is in modes *p1Open* or *p1Fer* (faults)
- *Red Logic* is either in mode *rlCrypt* (normal), *rlOpen* (fault) or *ctlFRStuck-Crypt* (fault)
- *Red Buffer* is either in mode *sdStuckOff* (fault), *rbOpen* (fault) or *rbNorm* (normal)

This path involves a main processor fault that causes it to pass unencrypted text out to the low security domain. While this failure is protected against by the incorporation of the second processor and shutdown logic, an additional failure of the comparator logic or the red buffer may cause unwanted information flow. However, the designers may believe that such combinations of faults are unlikely. The first path is more worrying as it demonstrates that even if the main processor is functioning correctly, a failure may occur if the bypass control pin is stuck on (*ctlFRStuckTxt*), allowing the encryption processor to be bypassed continuously. The second path involves a failure of the tri-state buffer quad chip in which the unencrypted red text is connected with the low-security domain directly by a single short across the chip! From this information, the operator of the tool gains the knowledge that there are many potential single points of failure in this model of the system.

5.3 Directed Fault Analysis

Given the red and black tagging of ports, it is possible to establish every combination of operation modes that result in a connection between the high- and low-security domains. This analysis is a restricted form of the transitive analysis and consequently requires less execution time. The results of this analysis are displayed immediately in the form of the tree view as described above. The tool

provides an option to display the connectivity between multiple red and black ports simultaneously, combining all results in the one window.

6 Future and Related Work

The transitive failure analysis and the directed failure analysis are both computationally intensive. An alternative approach is under investigation which involves the generation of a state machine for the SAL model-checker [2]. With this, the tool provides the accompanying Linear Temporal Logic (LTL) theorem stating that it is never possible to reach a black port starting from a red port. The generated counter-examples produce the same results as the current tool, but with the model-checker it is possible to be much more specific when analysing the results. This line of work follows other model-checker-based approaches [3,1] and would allow us to increase the expressiveness of the model (for example, to include information about control flow through the system, and to place guards upon selected operating modes depending upon the states of these control flows).

VHDL [10] is a circuit description language which is extensively used in industry. This language enables simulations to be written which explore the circuit for failures arising from inserted faults. However, simulations are based on a functional description of the circuit whereas ours is focused specifically on information flow. Our approach enables a more abstract analysis and the provision of dedicated algorithms supporting secure information flow analysis. SIFA is capable of directly importing netlist data from VHDL, and provides for a component library where components may be defined once and instantiated arbitrarily throughout the model. Imported models are intended to be abstracted by the evaluator to create a more meaningful model based on information flow.

7 Conclusion

Performing extensive analysis of information flow through domain separation devices is an expensive, time-consuming process. The tool and architecture discussed in this paper serve to ease and augment the analysis of such devices by readily modelling possible paths of information flow in different operating states, and by allowing the exploration of the ramifications of random hardware failures throughout the system.

Our approach incorporates abstraction of system components as both hierarchies and (possibly partial) views. Our use of multi-view models, now quite common in the specification of software/system requirements [9], appears to be novel in the domain of information flow analysis through electronic circuits.

Both flow and cutset analysis are performed on a single diagram (view) allowing the security critical region to be defined incrementally. This approach yields a practical method for focussing the evaluator's efforts when large systems are being modelled.

Once the security critical region is defined, and critical operating modes and corresponding information flows have been added, failure analysis may be

performed at any point in the component hierarchy (incorporating all nested views). The tool presents the composition of individual component operating modes in a fashion that exposes critical elements. Such critical elements include the existence of connectivity between domains (by viewing the calculated matrix as a whole), and the detailed presentation of paths leading from high-security sources to low-security sinks (by viewing particular cells in the matrix). The latter presentation highlights possible single points of failure that would lead to a compromise in information security; whether that be by design, or by random failure.

Acknowledgements

This research is funded by an Australian Research Council Linkage grant, LP0347620: *Formally-Based Security Evaluation Procedures* conducted in conjuction with the Defence Signals Directorate. This article has greatly benefited from proof-reading by, and discussion with Colin Fidge, Andrew Rae, and Andrew Matthews.

References

1. Antonio Cerone and George J. Milne. A methodology for the formal analysis of asynchronous micropipelines. In *Proceedings of the Third International Conference on Formal Methods in Computer-Aided Design*, pages 246–262. Springer-Verlag, 2000.
2. Leonardo de Moura, Sam Owre, Harald Rueß, John Rushby, N. Shankar, Maria Sorea, and Ashish Tiwari. SAL 2. In Rajeev Alur and Doron Peled, editors, *Computer-Aided Verification, CAV 2004*, volume 3114 of *Lecture Notes in Computer Science*, pages 496–500, Boston, MA, July 2004. Springer-Verlag.
3. Z. Furedi and R. P. Kurshan. Minimal length test vectors for multiple-fault detection. *Theor. Comput. Sci.*, 315(1):191–208, 2004.
4. J. Graves. Cryptographic device. Technical report, Defence Signals Directorate, 2003.
5. Daniel Jackson. Structuring Z specifications with views. *ACM Trans. Softw. Eng. Methodol.*, 4(4):365–389, 1995.
6. The Common Criteria Project Sponsoring Organisations. Common criteria for information technology security evaluation. Technical Report Standard 15408, 2.1 edition, ISO/IEC, 1999.
7. A. J. Rae and C. J. Fidge. Identifying critical components during information security evaluations. *Journal of Research and Practice in Information Technology*, 2005. Accepted for publication.
8. A. J. Rae and C. J. Fidge. Information flow analysis for fail-secure devices. *The Computer Journal*, 48(1):17–26, January 2005.
9. George Spanoudakis, Anthony Finkelstein, and Wolfgang Emmerich. Viewpoints 96: international workshop on multiple perspectives in software development (sigsoft 96) workshop report. *SIGSOFT Softw. Eng. Notes*, 22(1):39–41, 1997.
10. Mark Zwolinski. *Digital system design with VHDL*. Pearson Education, 2nd edition, 2004.

Cancelable Key-Based Fingerprint Templates

Russell Ang, Rei Safavi–Naini, and Luke McAven

School of Information Technology and Computer Science, University of Wollongong, Northfields Avenue, NSW 2522, Australia
[rja02,rei,lukemc]@uow.edu.au

Abstract. Biometric based authentication can provide strong security guarantee about the identity of users. Security of biometric data is particularly important as compromise of the data will be permanent. Cancelable biometrics store a non–invertible transformed version of the biometric data and so if the storage is compromised the biometric data remains safe. Cancelable biometrics also provide a higher level of privacy by allowing many templates for the same biometric data and hence non-linkability of user's data stored in different databases. We define how to measure the success of a particular transformation and matching algorithm for fingerprints. We consider a key–dependent geometric transform that is applied to the features extracted from a fingerprint, to generate a key–dependent cancelable template for the fingerprint. We investigate performance of an authentication system that uses this cancelable fingerprint when a fingerprint matching algorithm is used for detection. We evaluate performance of the system and show the challenges of achieving good performance if the matching algorithm is not modified.

1 Introduction

User authentication is becoming increasingly important. Integrity of data and transactions in various applications relies on authenticity of participants' identities.

The three basic forms of user authentication that can be used independently or in combination with others, are *knowledge based* which rely on a secret such as a password held by the user, *token based* that rely on possession of a 'token', such as a physical key or smartcard and *biometric based* that uses unique characteristics of individuals, such as fingerprints or voice prints.

While knowledge can be forgotten, and tokens stolen or lost, biometrics do not suffer from these deficiencies, and can provide the security of long passwords without sacrificing the ease of memorizing short ones.

Biometric authentication has two phases, *enrolment* and *authentication* (or *verification*). Enrolment involves measuring an individuals biometric data to construct a *template* for storage. Authentication involves a measurement of the same data and comparison with the stored template.

Biometric readings are rarely identical and readings are environment and device dependent. A template provides an approximate version of the biometric data, and verification succeeds if the second reading is close to the stored template.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 242–252, 2005.

Biometric characteristics are largely immutable and as such their compromise is permanent. For fingerprints, template compromise may allow the construction of artificial fingerprints [5]. Fingerprint images can be artificially generated [1], and there exists commercial software which is able to generate synthetic fingerprints [9].

Even without database compromise, biometric possession by government or medical organisations provides the potential for information misuse as the data belonging to the same user can be easily linked. Cancelable biometrics store a transformed version of the biometric data. The transformation is one way and so knowledge of a transformed biometric does not leak information about the actual biometric data. Moreover, by using different cancelable templates, data belonging to the same user cannot be linked. In this paper, we consider cancelable biometrics that are generated through a keyed transformation and investigate performance of the system when the verification method stays the same as the one used for the original fingerprint template.

Related Work

Encrypting the template prior to storage can make template compromise harder, but introduces key management problems. Furthermore, key compromise results in the full template being revealed, compromising the biometric. A *fingerprint vault* [3], is a fuzzy vault [7] that uses the feature set of a fingerprint to encode a secret in a polynomial. Biometric readings close to the feature set allows the polynomial to be reconstructed and the secret to be extracted. Implementation of this system using real biometric data requires an appropriate representation for this data. It is shown [3] that finding the secret in the fingerprint vault requires 2^{69} trials, to each valid users trial. However, the system has a high false rejection rate of about 30%, which is unacceptable in practice.

Another approach is to store a "hash" of the biometric data, rather than the biometric data itself. Cryptographic hashes are bit sensitive and not suitable for matching two readings that are slightly different. Biometric hashes have been described which allow 'close' biometric readings to be hashed into the same hash value. For example, [13] describe a biometric hash for handwriting.

Fuzzy commitment schemes [8] commit a secret using fuzzy data x, such as biometric data. An approximate version of x can recover the committed secret. The scheme uses error correcting codes and decommitment requires decoding operations which could be computationally inefficient.

Cancelable biometrics [11] apply non–invertible transforms to the biometric template and store the result. For verification, a biometric reading undergoes the same transformation before comparison with the stored (transformed) template. In a well designed system, two transformed outputs will 'match' if the initial templates 'matched' under the template matching algorithm. That is, the transform will not affect the matching result or the outcome of verification. The biometrics are cancelable in the sense that one cannot derive the original fingerprint from the transformed template, and comprising the stored template doesn't compromise the biometric characteristic of the user. Different cancelable biometrics can

be given to different collectors, for example government and health bodies, to ensure that misuse minimises relationship leaking between databases.

1.1 Our Contribution

We consider a fingerprint authentication system that uses a key dependent non–invertible transformation applied to readings before storage. Using a key dependent transformation allows us to have different stored template for different applications (databases) so reducing the chance of linkability of information related to a person. The system can be seen as a key–dependent cancelable biometric system [11]. The key–dependent transformation we use is applied on a minutiae based representation of fingerprint data. We use the fingerprint template matching algorithm of [6], to match transformed templates. Although the transformation affects the structure of the template we justify why using the same matching algorithm is meaningful. The algorithm performs well for untransformed fingerprint templates and the aim of our experiment is to investigate suitability of the matching algorithm, or a simple modification thereof, for the transformed template. We find the matching score is affected by the transform, so the resulting system has increased false acceptance and false rejection rates, and so cannot be effectively used in practice. Another shortcoming of the proposed transform is its insensitivity to small changes to the key resulting in a reduced key space. Our work will hopefully motivate further work in this important area.

The rest of this paper is structured as follows. In Section 2 we describe fingerprint structure. Section 3 contains the user authentication procedures, and a description of the matching algorithm used [6].

In Section 4 we describe a generic process for implementing key–based transforms for cancelable key–based biometrics. We define a measure of the success of a transform and matching algorithm.

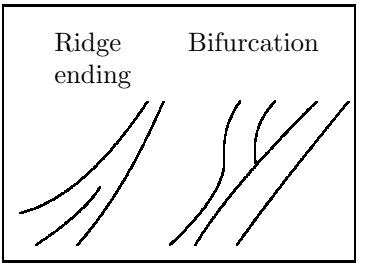
Section 5 contains the transform we adopt as an illustration. In section 6 we analyse the results of implementing this transform. We summarise our results in Section 7.

2 Fingerprints

Fingerprint recognition is probably the oldest method of biometric identification applicable to all persons, apart from the obvious facial recognition used in everyday life. A fingerprint consists of *ridges*, lines across fingerprints, and *valleys*, spaces between ridges. The ridge and valley pattern is unique to each individual. The three basic macroscopic features in ridge flow are arches, loops and whorls. Discontinuities in the ridges are referred to as *minutiae*[9]. Most fingerprints have between 70 and 150 minutiae.

The two major methods of fingerprint matching are *minutiae matching* and *global pattern matching*. The former analyses relative positions and structure of minutiae, while the latter considers macroscopic ridge flows. Most automated systems use minutiae matching. With suitable resources this method is more

accurate and can be faster. Performance is very dependent on image quality and the resolution of scanned fingerprints though.
It can also be expensive in processor and monetary terms, and potentially requires a large database. We use minutiae based matching, applied locally and globally following [6]. While about 18 minutiae types are recorded, the 2 most frequently occurring, and most frequently used in matching, are bifurcation and ridge ending. These types are formally defined in [2], where a bifurcation is thought of as a valley ending.



One cannot be sure even of the type being the same in different readings, since the distinction between bifurcation and ridge ending is not as clear in practice as in theory [2]. Pressure on the finger during a reading can squash adjacent ridges so that a ridge ending appears as a bifurcation.

3 Authentication Using Fingerprints

There are two phases in authentication using fingerprints.

1. A user U *enrols* through a trusted service that uses the measurement of a fingerprint to generate a template $\mathcal{T}$. The identity and template, $(U, \mathcal{T})$, are stored in a database.
2. To *authenticate* a user U produces a fingerprint reading $\mathcal{R}$. If $\mathcal{T}$ and $\mathcal{R}$ are close the user is accepted as being U.

Two important parameters in determining the quality of an authentication system are the *false rejection rate* (FFR) and the *false acceptance rate* (FAR). FFR records how often authentication fails when it should succeed, while FAR records how often authentication succeeds when it should fail. An overall measure is given by the point where FFR equals FAR. The error at this level is referred to as the *equal error rate* (EER).

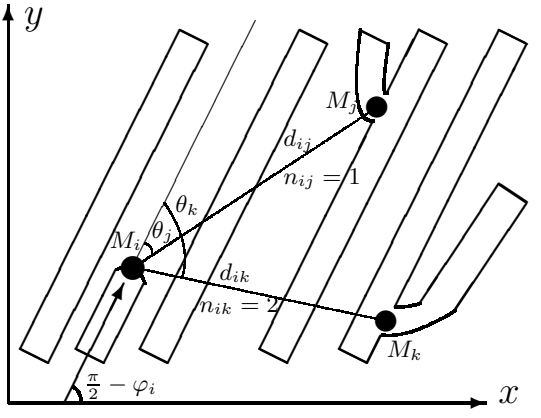
3.1 Enrolment: Template Generation, Processing and Storage

Enrolment uses a fingerprint capture device [10] to generate a fingerprint image. Processing tools, such as in the VeriFinger toolkit [12], are used to tidy up the image. Edge detection is used, for example. After processing, features such as minutiae can be readily identified. Statistical techniques are often used to choose statically reliable features [3]. We use a single reading to form a template, since this simply preserves the ridge structure.

We represent the fingerprint template $\mathcal{T}$ by a minutiae set, referred to as the *feature set*, identified on the fingerprint. The set is obtained using `MINDTCT` [4], and aligned according to the `R92` algorithm [4]. The template $\mathcal{T}$ containing the minutiae is stored in an encrypted database.

For each minutiae M_i we have a vector containing position (x_i, y_i), local ridge orientation φ_i, type t_i, and a local feature vector F_i. Euclidean d_{ij} and ridge n_{ij} distances between M_i and M_j are given, the latter being the number of ridges between M_i and M_j in a straight line. It determines which minutiae are closer.

The angle θ_j between ridges at M_i and M_j is used. For neighbours M_j we record type and the angle φ_{ij} of the ridge at M_j relative to the ridge angle at M_i. The vector F_i includes $(d_{ij}\theta_j\varphi_{ij}n_{ij}t_j)$ for each of L neighbours M_j. The local minutiae geometry is shown here for $L = 2$.



3.2 Verification: Authentication and the Matching Process

Verification requires that a user present their finger for reading. Processing applied in the template generating process is also applied here, and again the fingerprint can be recorded as a set of minutiae, $\mathcal{R}$. Comparing $\mathcal{T}$ and $\mathcal{R}$, using a matching algorithm, produces a matching score. If the score is above a threshold, the user is accepted. We adopt a matching algorithm [6] with elements of local and global matching.

Local matching uses $F_i^{\mathcal{T}}$ and $F_j^{\mathcal{R}}$ for template and reading minutiae, respectively. Local matching is less reliable but is rotation and translation invariant, since it depends only on local feature vectors which have those invariances. We define [6] a weight vector W, containing a weight for each element of the F_i. This allows us to vary tolerance distribution. We use the same, empirically chosen, weight parameters as [6].

We specify a threshold parameter $t_p = 6(5L + 1)$, for L nearest neighbours. We also define a local structure similarity function $s_l(i, j)$;

$$s_l(i,j) = \begin{cases} \frac{t_p - W.|F_i^{\mathcal{T}} - F_j^{\mathcal{R}}|}{t_p} & \text{if } W.|F_i^{\mathcal{T}} - F_j^{\mathcal{R}}| < t_p \\ 0 & \text{Otherwise} \end{cases}$$

so that $s_l(i, j) = 1$ for a perfect match and 0 for a mismatch.

One calculates $s_l(i, j)$ for all pairs of minutiae M_i, M_j from $\mathcal{T}$ and $\mathcal{R}$, respectively. One adopts the structure supported by matching the local neighbourhoods given corresponding i to j for $\max_{i,j} s_l(i, j)$. Within this structure one considers matching the global parameters of the minutiae; distance, angle and ridge angel relative to the reference minutiae.

The reference minutiae in $\mathcal{T}$ and $\mathcal{R}$ are M_i and M_j, respectively, since they are taken to be in correspondence. Each minutiae in $\mathcal{R}$ is matched against each

in $\mathcal{T}$, and a non–zero score $s_g(i,j)$ is given, $\frac{1}{2} + \frac{1}{2}s_l(i,j)$, if the points i and j are within some bounding box of each other in the global parameter space. One avoids double matching by setting $s_g(i,j)$ to zero if there any exists a k such that $s_g(i,k) \geq s_g(i,j)$ or $s_g(k,j) \geq s_g(i,j)$. The overall matching score is expressed as

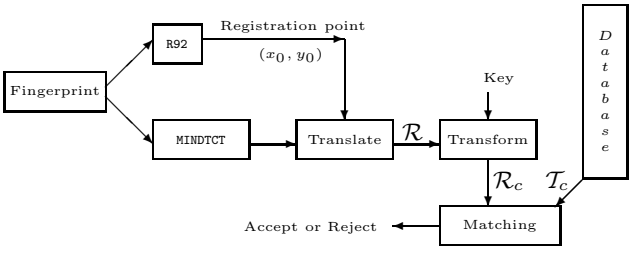
$$M_s = 100\frac{\sum_{i,j} s_g(i,j)}{\max\{M,N\}} \ .$$

A reading $\mathcal{R}$ is accepted as validating the user if the matching score between $\mathcal{R}$ and $\mathcal{T}$ is higher than some threshold. We note, in particular, that spurious minutiae tend to count negatively to the matching score in that they increase M and N without adding any $s_g(i,j)$.

4 Enrolment and Authentication for Key-Based Templates

We add an extra step to the system in Section 3. Prior to storage a non–invertible mapping, our example of which is described in Section 5, is applied to the template. For us this mapping is a global transformation which acts on the global minutiae parameters and reflects some minutiae across a line. The local feature vectors are calculated after this transformation, using the ridge structure at their new location.

Before a comparison with the stored template, the same mapping is applied against the reading. We show the modified verification process here.



We give two scenarios for key storage. Let $\mathcal{T}$ be a biometric template and $\mathcal{T}_c = C_k(\mathcal{T})$ the keyed cancelable biometric.

Scenario One: User key only storage.

The user *enrols* through a trusted service that uses a key k to generate a cancelable biometric $\mathcal{T}_c$ from k and the template $\mathcal{T}$ generated from the user's biodata. The user receives a smart card containing k, and the database stores $(\texttt{User}, \mathcal{T}_c)$.

For *verification*, a user presents $(\texttt{User}, \mathcal{R}, k)$ to the server. The reading $\mathcal{R}$ and key k are used, by the server, to calculate a cancelable biometric $\mathcal{R}_c$. The server compares $\mathcal{R}_c$ and $\mathcal{T}_c$ and, if close, the user is accepted.

Scenario Two: Database key and template storage.

The user *enrols* through a trusted service that uses a key k to generate a cancelable biometric $\mathcal{T}_c$ from k and the template $\mathcal{T}$ generated from the user's biodata. The database stores $(\texttt{User}, \mathcal{T}_c, k)$.

For *verification*, a user presents (`User`,$\mathcal{R}$) to the server. The reading $\mathcal{R}$ and key k are used, by the server, to calculate a cancelable biometric $\mathcal{R}_c$. The server compares $\mathcal{R}_c$ and $\mathcal{T}_c$ and, if close, the user is accepted.

4.1 Matching After the Transformation

Let $\mathcal{X}$ be the set of all possible images, and let the matching algorithm M act on any two images x_1, x_2 to give a matching score $M(x_1, x_2)$. Let $\mathcal{C}$ be a set of keyed transformations, $\mathcal{C} : \mathcal{X} \to \mathcal{X}_\mathcal{C}$. Let C_k denote the particular transformation associated with $k \in \mathcal{K}$, where $\mathcal{K}$ is the key set, and the matching algorithm M' act on any two transformed images $C_k(x_1), C_k(x_2)$ to give a matching score $M'(C_k(x_1), C_k(x_2))$.

We consider compatibility of M and M' with the transformation.

Definition 1 *We say a transform $\mathcal{C} : \mathcal{X} \to \mathcal{X}_\mathcal{C}$ is ϵ–match preserving, for the pair (M, M') of matching algorithms, if, for any pair of images $x_1, x_2 \in \mathcal{X}$ and for every $k \in \mathcal{K}$, $|M'(C_k(x_1), C_k(x_2)) - M(x_1, x_2)| \leq \epsilon$.*

In practice only a subset of images and keys would be considered.

The 'effective' number of keys is an indication of the level of security of system. Two different keys may produce similar matching results and so one key can replace the other.

Consider two keys k_1 and k_2. For an image x let the transformed values under keys k_1 and k_2 be $C_{k_1}(x)$ and $C_{k_2}(x)$.

Definition 2 *We say k_1 and k_2 are κ–distinct if, for any not necessarily distinct image pair x, x',*

$$|M'(C_{k_2}(x), C_{k_1}(x') - M'(C_{k_1}(x), C_{k_1}(x'))| \geq \kappa \quad .$$

If κ is small, false acceptance can result. That is, a new reading can be accepted under an incorrect key. We identify a subset of keys that can be reliably distinguished. The size of this set must be sufficiently large to provide the required security.

5 A Key Based Transformation for Fingerprints

Here we define a key dependent transformation of the minutiae. The key $0 \leq \phi \leq \pi$ is an angle specifying a line $ax + b$ through the core point (o_x, o_y). Our transformation is illustrated in Figure 1 and specified as;

`Transformation`	
Input: $\phi, (o_x, o_y), (x_i, y_i)$	
$p_i = o_x - x_i, q_i = o_y - y_i$.	
If $p_i \tan(\phi) > q_i$	
$\omega = 2(\theta - \tan^{-1}(q_i/p_i))$	
$p'_i \longleftarrow \cos(\omega)p_i - \sin(\omega)q_i$	$q'_i \longleftarrow \sin(\omega)p_i + \cos(\omega)q_i$
$x_i \longleftarrow p'_i + o_x$	$y_i \longleftarrow q'_i + o_y$.
Output (x_i, y_i)	

This algorithm means minutiae under the line specified by the key are reflected to above the line, while those above are unchanged.

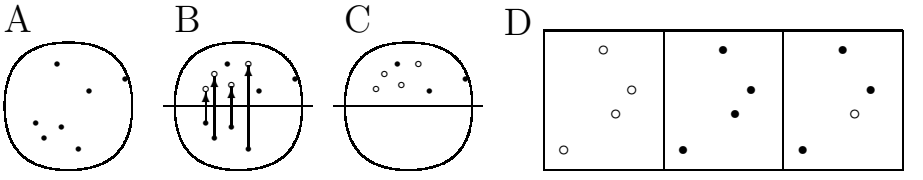


Fig. 1. Diagrams A–C respectively show the minutiae as in the fingerprint reading, the transformation with the hollow circles being the new positions, and the positions after the transformation. We refer to the region all the minutia lie in at the end as the "upper region". The three possible neighbourhood structures in the upper region are shown in Diagram D; all–reflected, none reflected and a mixture.

As noted in Section 4 the local feature vectors are calculated using the new minutiae positions and the ridge structure from the original reading. Without reference to the ridge structure the distribution of minutiae is effectively the same as fingerprints. There are, however, some relative minutiae positions that are at best unlikely. In particular, one of two very close minutiae in the transformed space is likely to be have been transformed, so the resulting cancelable biometric leaks information about the original fingerprint.

5.1 The Effect of the Transformation on Matching

In general applying transform to templates requires a new matching algorithm be used for the transformed data. Here we argue that for our proposed transform, it is plausible to use the same matching algorithm.

Local structures were discussed in Section 3.2 and are dependent upon the local feature vectors of each minutiae. The ridge structure used to generate the local feature vectors for the cancelable template $\mathcal{T}_c$ belongs to the "upper region" of the image. As such, the local structure vectors are unlikely to be similar to those generated prior to the transformation.

The local structure matching algorithm has two purposes. It identifies a pair of minutiae, one from $\mathcal{T}$, the other from $\mathcal{R}$, to put into correspondence. Global matching takes place relative to this pair. The algorithm also calculates local matching scores $s_l(i, j)$ used in global matching.

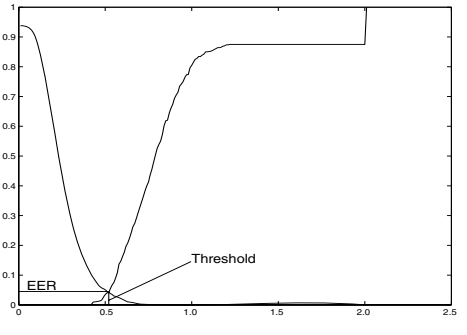
Consider two similar readings, $\mathcal{R}$ and $\mathcal{R}'$. Following Definition 1 we want to consider if the variances tolerated between $\mathcal{R}$ and $\mathcal{R}'$ are still tolerable between the transformed $\mathcal{R}_C$ and $\mathcal{R}'_C$, that is, is the tolerance preserved. We consider the effect on the different parameters, recalling that the local structure depends on Euclidean distance and direction angle between minutiae, relative ridge angle, ridge distance and type.

Tolerable variances in position, and thus Euclidean distance, are still tolerable after the transform. In general it is likely the neighbourhoods will change, however it is still possible there is a primary preserved closest matching neighbourhood. Local matching only needs one neighbourhood. However the neighbourhoods are based on ridge distance, which changes substantially in the transform. The ridge structures are all in the upper region, so tolerable variations on the original ridge structure cannot easily guarantee tolerable variations on the final ridge structure. In general there will be little correlation between the ridge structure in the original location and that in the final location. Thus ridge distance and angles could (and do) cause significant problems in tolerance preservation under the transform. While the ridge structure is problematic, the tolerance in Euclidean structures suggests the transformation is worth investigating.

6 Results and Analysis

We used a batch of 80 fingerprints from [9]. Those images consist of 8 images each of 10 fingerprints. NIST Fingerprint Image Software [4] was used to identify the `R92` core point and extract the minutiae.

We implemented the transformation algorithm to transform minutiae positions. We also implemented the matching algorithm [6], which returns a similarity level, between 0 and 2, for two feature sets. We considered matching the fingerprints before the transforms. The false rejection rates and false acceptance rates are shown here for $L = 5$. The axes are percentage of cases (the y–axis) and matching score. For pre–transformation we have an approximate Equal Error Rate (EER) of 4%, occuring at a threshold of about 0.52. To the right we tabulate $L = 2 \ldots 6$ results.



L	Equal error rate	Threshold
2	2%	0.53
3	2.5%	0.53
4	3.5%	0.51
5	4%	0.52
6	$31 - 50\%$	$0.03 - 0.035$

Having analysed the pre–transformation matching, we applied the transforms for twenty keys, spaced by $\pi/10$, to each original template, and consider matching on the results. There are three types of results.

1: We calculate the FRR, FAR and EER rates post–transform. We illustrate a typical $L = 5$ example in Figure 2A. There is a significant increase in FRR and FAR. In particular, errors in aligning transformed templates reduces matching accuracy. EER post–transform was found to be $\approx 16.8\%$, at a threshold of ≈ 0.54. While the thresholds are similar, which is good, the EER is significantly higher. This result supports the use of the transform since other systems [3] have rates as high as 30%.

2: Following Definition 2 we compare matching between images transformed under different keys. For a useful algorithm the probability of matching an image transformed under different keys should be low. In Figure 2B we illustrate how the matching score between transforms under different keys, of the same typical image, are still high. For example, at a threshold of about 0.52, about 70% of the images were still accepted, even though only 5% should be. This very high error rate is a significant problem, at best meaning a small key space.

3: We examine how the transformation and matching algorithms perform under the measure in Definition 1. Though small variance in FFR and FAR curves is good, the reality is shown in the shifts table to the right. The variance in matching score is rather large. Since the definition of ϵ–invariance requires ϵ to cover all transformations, ϵ is too large to be of use. The proportion of such large shifts is also high .

L	Average	Maximum
2	0.147	0.935
3	0.148	0.746
4	0.152	0.849
5	0.161	0.877
6	0.076	1.955

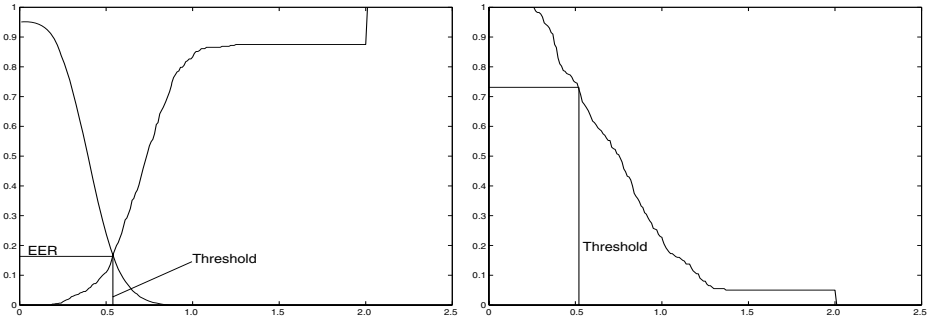


Fig. 2. On the left we show a typical $L = 5$ post–transform result. The graphs show the proportion of cases where false results are accepted (the higher curve at the beginning) and where true results are rejected. The axes are percentage of cases (the y–axis) and matching score. The EER occurs at a matching score of about 0.54. The proportion of errors at the threshold is about 17%. On the right we compare we compare the matching scores of images resulting from transformations under different keys, for a typical fingerprint image. The graph shows the proportion of cases having a matching score of at least that on the bottom axis. We see, for example, about 70% of the images are matched with a threshold of 0.52 suggested by the pre–transformation results.

Results for $L = 2$ to $L = 5$ appear similar. For $L = 6$ the correlation required means either the image is very close and can match, or is not very close and won't match, even if by eye the images are similar. This suggests using neighbourhoods of at most five in this context.

7 Conclusion

The use of biometrics in user authentication systems is very promising. However, without adequate security considerations, the compromise of such biometrics may result in them being useless for the user forever. We consider the cancelable biometrics of Ratha *et al* [11] and describe some measures of how good a cancelable fingerprint system, primarily consisting of a key–based transformation and a matching algorithm, is.

We consider a specific transformation and matching algorithm, and use the measures to suggest the pair is not useful in it's current form. We consider that good key–based template generators and matching algorithms probably need to be developed together, rather than explicitly using existing matching algorithms.

References

1. J. Araque, M. Baena, and P. Vizcaya 'Synthesis of fingerprint images', *Proceedings of the 16th International Conference on Pattern Recognition*, **II** (2002) 422–425.
2. R. M. Bolle, A. W. Senior, N. K. Ratha and S. Pankanti 'Fingerprint minutiae: A constructive definition.' *Biometric Authentication* (2002) 58–62.
3. T. Clancy, D. Lin and N. Kiyavash, 'Secure smartcard–based fingerprint Authentication', *ACM Workshop on Biometric Methods and Applications*, (2003) 45–52.
4. M. Garris, C. Watson, R. McCabe and C. Wilson 'Users guide to NIST fingerprint image software.' NIST (2001).
5. C. Hill 'Risk of masquerade arising from the storage of biometrics'. Honours thesis, ANU, (2001). `http://chris.fornax.net/download/thesis/thesis.pdf`
6. X. Jiang and W. Yau 'Fingerprint Minutiae Matching Based on the Local And Global Structures' *Proceedings of the 15th International Conference on Pattern Recognition* **II** (2000) 6038–6041.
7. A. Juels and M. Sudan 'A fuzzy vault scheme', *Proceedings of the 2002 IEEE International Symposium on Information Theory* (2002) 408–.
8. A. Juels and M. Wattenberg 'A fuzzy commitment scheme.' *ACM Conference on Computer and Communications Security* (1999) 28–36.
9. D. Maltoni, D. Maio, A. Jain, and S. Prabhakar **Handbook of Fingerprint Recognition**, (Springer–Verlag, New York, 2003).
10. L. O'Gorman 'Practical systems for personal fingerprint authentication.' *IEEE Computer* **33**(2) (2000) 58–60.
11. N. K. Ratha, J. H. Connell and R.M. Bolle, 'Enhancing security and privacy in biometrics–based authentication systems' *IBM Systems Journal* **40**(3) (2001) 614–634.
12. VeriFinger. Neurotechnologija Ltd. `http://www.neurotechnologija.com`.
13. C. Vielhauer, R. Steinmetz and A. Mayerhofer, 'Biometric Hash based on Statistical Features of Online Signatures' *Proceedings of the 16th International Conference on Pattern Recognition* **I** (2002) 123–126.

Hybrid Signcryption Schemes with Insider Security
(Extended Abstract)

Alexander W. Dent

Royal Holloway, University of London
Egham Hill, Egham, Surrey, TW20 0EX, U.K.
a.dent@rhul.ac.uk
http://www.isg.rhul.ac.uk/~alex/

Abstract. The question of constructing a hybrid signcryption scheme with outside security was considered by Dent [7]. That paper also demonstrated that the basic hybrid construction formalised by Cramer and Shoup [5,9] is incapable of producing a signcryption scheme with insider security. This paper provides a paradigm for constructing signcryption schemes with insider security based on the ideas of hybrid cryptography.

1 Introduction

Hybrid cryptography is concerned with the combination of keyed symmetric and asymmetric schemes in order to produce schemes that are more advantageous than those constructed using "pure" symmetric or asymmetric techniques alone. Typically, this takes the form of an asymmetric cryptosystem making use of a generic keyed symmetric cryptosystem with certain security properties as a subroutine. This enables the construction of asymmetric schemes in which some of the computational load is taken by the more efficient symmetric cryptosystems without compromising the security of the overall cryptosystem. Traditionally, hybrid cryptography is used to create asymmetric encryption schemes where the actual encryption of the message is provided by a symmetric encryption scheme (for example, AES in CBC mode) under a randomly generated symmetric key. The asymmetric encryption scheme is then used to encrypt this randomly generated symmetric key. This allows the asymmetric encryption scheme to handle long messages, a problem with some "pure" asymmetric encryption schemes.

Cramer and Shoup [5,9] proposed a paradigm whereby the asymmetric and symmetric parts of the cryptosystem are formally separated into an asymmetric KEM and a symmetric DEM. The authors proposed separate security criteria for the KEM and the DEM that would, if fulfilled, guarantee that the overall encryption scheme was secure. Dent [7] extended this paradigm to the signcryption setting by proposing new security criteria for the KEM and the DEM, although the model only covers the case where the signcryption scheme was attacked by third parties (known as *outsiders* by An *et al.* [2]).

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 253–266, 2005.

This paper extends earlier work by proposing a hybrid paradigm for signcryption schemes secure against attacks made by *insiders*, i.e. the resultant schemes should be secure against attacks against the confidentiality of the message made by any third party and from forgery attacks made by any person except the sender. We also note the infeasibility of producing efficient hybrid signature schemes.

2 KEMs, DEMs and the Impossibility of Hybrid Signature Schemes

A KEM–DEM encryption scheme is composed of two parts: an asymmetric *key encapsulation mechanism* (KEM) and a symmetric *data encapsulation mechanism* (DEM). To encrypt a message m, the KEM is first used, with the public key, to produce both a symmetric key K and an asymmetric encryption (or "encapsulation") of that key C_1. The message m is then encrypted symmetrically using the DEM and the randomly generated symmetric key K to give a ciphertext C_2. The encryption ciphertext is the pair (C_1, C_2). An encryption ciphertext (C_1, C_2) is decrypted by first decapsulating C_1 with the KEM and the private key to recover the symmetric key K, and then using the DEM and the symmetric key K to recover the message m from C_2.

As a step towards building a hybrid signcryption scheme, we consider the problem of building a hybrid signature scheme. A signature scheme needs to provide integrity, data origin authentication and non-repudiation services. Since a symmetric MAC scheme can provide both a integrity and an origin authentication service, we may have some hope that we can build a hybrid signature scheme[1,2].

Naively we may try and build a hybrid signature scheme (that uses a public verification key pk and a private signing key sk) for a message m as follows. To sign a message m:

1. The KEM is executed on the private key sk to produce a symmetric key K and an encapsulation of that key C_1.
2. The DEM is executed on the message m and the symmetric key K, and produces a cryptographic check value C_2.

[1] Of course, no symmetric scheme can provide a non-repudiation service without the use of a trusted third party. Hence, the KEM must act in such a way as to make sure that the overall scheme provides a non-repudiation service.

[2] It is unlikely that a hybrid signature scheme is likely to be of much practical use. It is likely that any KEM, due to its asymmetric nature, is likely to contain at least one "slow" operation (such as modular exponentiation or scalar multiplication of an elliptic curve point). Since there exist fast signature algorithm that only makes use of one "slow" operation, such as RSA-PSS [4] and Schnorr [8], any hybrid signature scheme is likely to be slower than its "pure" counterpart.

The signature is the pair (C_1, C_2). To verify such a signature:

1. The KEM is executed using the public key pk and the first part of the signature C_1, and outputs either a symmetric key K or the error symbol $\perp$. If the KEM outputs $\perp$, then the verification algorithm outputs *invalid* and terminates.
2. The DEM is executed using the message m, the symmetric key K and the second part of the ciphertext C_2. The DEM outputs either *valid* or *invalid*. The verification algorithm outputs either *valid* or *invalid* depending on the DEM's output.

It is easy to see that no hybrid signature scheme of this form can ever be secure. An attacker can always forge a signature for any message m by requesting the signature (C_1, C_2) of a message m' from the signer, recovering the symmetric key K used to create the signature (from C_1 and pk), and creating a new cryptographic check value C_2' by executing the DEM on the message m using the symmetric key K. The pair (C_1, C_2') is a valid signature for the message m. If we are to avoid this problem then we are forced to alter the KEM–DEM paradigm.

Definition 1. *A signature KEM is a triple* $(Gen, Encap, Decap)$ *where*

- *Gen is a probabilistic algorithm that take a security parameter* 1^k *and outputs a public/private key pair* (pk, sk)*, where sk is a private signing key and pk is the corresponding public verification key.*
- *Encap is a probabilistic key encapsulation algorithm that takes as input a private key sk and a message m, and outputs a symmetric key K and an encapsulation of that key* C_1*.*
- *Decap is a deterministic key decapsulation algorithm that takes as input a public key pk, a message m and an encapsulation* C_1*, and outputs either a symmetric key K or the error symbol* $\perp$*.*

We require that a signature KEM is sound, i.e. if (pk, sk) *is a public/private key pair, m is a message, and* $(K, C_1) = Encap(sk, m)$ *then* $K = Decap(pk, m, C_1)$*.*

Thus, the KEM produces symmetric keys that depend on the message being signed, as well as the public key.

Definition 2. *A signature DEM is a pair* (Compute, Check) *where*

- Compute *is a deterministic algorithm that takes as input a message m and a symmetric key K, and outputs a cryptographic check value* C_2*.*
- Check *is a deterministic algorithm that takes as input a message m, a symmetric key K and a check value* C_2 *and outputs* valid *if* $\textsc{Compute}(m, K) = C_2$ *and* invalid *otherwise.*

We construct a signing algorithm for a message m as follows:

1. Set $(K, C_1) = Encap(sk, m)$.
2. Set $C_2 = \textsc{Compute}(m, K)$.
3. Output (C_1, C_2).

The corresponding verification algorithm for a message m and a signature (C_1, C_2) is:

1. Set $K = \mathit{Decap}(pk, m, C_1)$. If $K = \perp$ then output *invalid* and terminate the algorithm.
2. Output $\textsc{Check}(m, K, C_2)$.

Now, to defeat the earlier simple attack, we require that no attacker be able to compute an encapsulation C_1 for a symmetric key K used to encrypt a message m except by querying a signature oracle. If an attacker can find such an encapsulation for a message m, then they can recover K using the public verification algorithm and compute $C_2 = \textsc{Compute}_K(m)$ using the DEM. The pair (C_1, C_2) would be a valid signature for the message m. However, if we insist upon the KEM having this security property then we can construct a signature scheme from the KEM alone as follows. To sign a message m under a private key sk:

1. Set $(K, C_1) = \mathit{Encap}(sk, m)$.
2. Output C_1.

To verify a signature C_1 of a message m under a public key pk:

1. Set $K = \mathit{Decap}(pk, m, C_1)$.
2. If $K = \perp$ then output *invalid*. Otherwise output *valid*.

Hence, whenever we have a secure hybrid signature scheme, we can construct a more efficient secure signature scheme by simply removing the DEM.

It is not surprising that we cannot construct an efficient secure hybrid signature scheme. Since the DEM can only provide integrity and data origin authentication services, the responsibility of providing the non-repudiation property falls to the KEM. However, a KEM that is providing a non-repudiation service also provides integrity and data origin authentication services.

3 Insider Secure Signcryption Schemes

The notion of signcryption was first introduced by Zheng [10]. For our purposes a signcryption scheme will consist of five algorithms:

1. A probabilistic polynomial-time common key generation algorithm, $\mathcal{G}_c$. It takes as input a security parameter 1^k and returns some global information (parameters) I.
2. A probabilistic polynomial-time sender key generation algorithm, $\mathcal{G}_s$. It takes as input the global information I and outputs a public/private key pair (pk_s, sk_s) for a party who wishes to send signcrypted messages.
3. A probabilistic polynomial-time receiver key generation algorithm, $\mathcal{G}_r$. It takes as input the global information I and outputs a public/private key pair (pk_r, sk_r) for a party who wishes to be able to receive signcrypted messages. Hence, a party who wishes to be able to both send and receive signcrypted messages will require two key-pairs: one for use when sending messages and one for use when receiving them.

4. A probabilistic polynomial-time generation-encryption algorithm, $\mathcal{E}$. It takes as input a message m from some message space $\mathcal{M}$, the private key of the sender sk_s and the public key of the receiver pk_r; and outputs a signcryption $C = \mathcal{E}(sk_s, pk_r, m)$ in some ciphertext space $\mathcal{C}$.
5. A deterministic polynomial-time verification-decryption algorithm, $\mathcal{D}$. It takes as input a signcryption $C \in \mathcal{C}$, the public key of the sender pk_s and the private key of the receiver sk_r; and outputs either a message $m = \mathcal{D}(pk_s, sk_r, C)$ or the error symbol $\perp$.

We require that any signcryption scheme is *sound*, i.e. that for almost all sender key pairs (pk_s, sk_s) and receiver key pairs (pk_r, sk_r) we have $m = \mathcal{D}(pk_s, sk_r, C)$ for almost all ciphertexts $C = \mathcal{E}(sk_s, pk_r, m)$. This definition of a signcryption scheme is essentially adapted from An [1].

We take our security notion for a signcryption scheme from An, Dodis and Rabin [2]. When we consider attacks against a signcryption scheme we have to consider two different types of attack. We have to consider attacks against the confidentiality of the scheme made by any third party (i.e. any party who is not the sender or the receiver); and we have to consider attacks made against the integrity of the scheme made by any party except the sender. This is known as *insider* security[3].

Both attacks against the confidentiality and attacks against the integrity are described in terms of a game played between an attacker and a hypothetical challenger. In each case the system is secure if the attacker's success probability or advantage is negligible as a function of the security parameter.

Definition 3. *A function f is negligible if, for all polynomials p, there exists an integer N_p such that $|f(x)| \leq 1/|p(x)|$ for all $x \geq N_p$.*

Confidentiality

The notion of confidentiality for a signcryption scheme is similar to that of an asymmetric encryption scheme. The attack model is defined in terms of a game played between a hypothetical challenger and a two-stage attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. For a given security parameter k:

1. The challenger generates some global information I by running the common key generation algorithm $\mathcal{G}_c(1^k)$; a valid sender key pair (pk_s, sk_s) by running the sender key generation algorithm $\mathcal{G}_s(I)$; and a valid receiver key pair (pk_r, sk_r) by running the receiver key generation algorithm $\mathcal{G}_r(I)$.
2. The attacker runs $\mathcal{A}_1$ on the input (pk_r, pk_s). This algorithm outputs two equal length messages, m_0 and m_1, and some state information *state*. During its execution, $\mathcal{A}_1$ can query a generation-encryption oracle that will, if given a message $m \in \mathcal{M}$, return $\mathcal{E}(sk_s, pk_r, m)$; and a verification-decryption oracle that will, if given a signcryption $C \in \mathcal{C}$, return $\mathcal{D}(pk_s, sk_r, C)$.

[3] The weaker notion of *outsider* security protects against attacks against the confidentiality or integrity made by any third party, but does not protect against attacks against the integrity made by the receiver.

3. The challenger picks a bit $b \in \{0,1\}$ uniformly at random, and computes the challenge signcryption $C^* = \mathcal{E}(sk_s, pk_r, m_b)$.
4. The attacker runs $\mathcal{A}_2$ on the input $(C^*, state)$. The algorithm outputs a guess b' for b. During its execution, $\mathcal{A}_2$ can query a generation-encryption oracle and a verification-decryption oracle as above, but with the restriction that $\mathcal{A}_2$ is not allowed to query the verification-decryption oracle on the challenge ciphertext C^*.

The attacker wins the game if $b' = b$. The attacker's advantage is defined to be:

$$|Pr[b = b'] - 1/2| \,. \tag{1}$$

Definition 4 (IND-CCA security). *A signcryption scheme is said to IND-CCA secure if, for all polynomial polynomial-time attackers $\mathcal{A}$, the advantage that $\mathcal{A}$ has in winning the above game is negligible as a function of the security parameter k.*

Integrity/Authenticity

The notion of integrity for a signcryption scheme is similar to that of a digital signature scheme. The attack model is defined in terms of a game played between a hypothetical challenger and an attacker $\mathcal{A}$. For a given security parameter k:

1. The challenger generates some global information I by running the common key generation algorithm $\mathcal{G}_c(1^k)$; a valid sender key pair (pk_s, sk_s) by running the sender key generation algorithm $\mathcal{G}_s(I)$; and a valid receiver key pair (pk_r, sk_r) by running the receiver key generation algorithm $\mathcal{G}_r(I)$.
2. The attacker runs $\mathcal{A}$ on the input (pk_s, pk_r, sk_r). This algorithm outputs a possible signcryption C^*. During its execution, $\mathcal{A}$ can query a generation-encryption oracle that will, if given a message $m \in \mathcal{M}$, return $\mathcal{E}(sk_s, pk_r, m)$.

The attacker wins the game if $\mathcal{D}(pk_s, sk_r, C^*) = m \neq \perp$ and $\mathcal{A}$ never received C^* as a response from generation-encryption oracle.[4]

Definition 5 (INT-CCA security). *A signcryption scheme is said to be INT-CCA secure if, for all polynomial-time attackers $\mathcal{A}$, the probability that $\mathcal{A}$ wins the above game is negligible as a function of the security parameter k.*

It is easy to see that a signcryption scheme that is both IND-CCA secure and INT-CCA secure maintains both the confidentiality and the integrity/authenticity of a message in the face of any attack. Therefore, we define:

Definition 6 (Insider security). *A signcryption scheme is said to be insider secure if it is IND-CCA secure and INT-CCA secure.*

[4] This is sometimes known "strong unforgeability" in order to differentiate it from "weak unforgeability", where an attacker is only deemed to have won if $\mathcal{D}(pk_s, sk_r, C^*) = m \neq \perp$ and $\mathcal{A}$ never submitted m to the generation-encryption oracle. So, with strong unforgeability, an attacker is deemed to have won if it can find a new signcryption of a message that has previously been signcrypted or if it can generate a signcryption of a new message.

4 Hybrid Signcryption Schemes

If we attempt to apply the standard hybrid encryption paradigm to the problem of creating a signcryption scheme with insider security, then we run into the same problem as we encounter when we attempt to create a hybrid signature scheme. In other words, suppose that we assume that we can separate a hybrid signcryption scheme into a KEM and a DEM, where the generation-encryption algorithm for a message m runs as follows:

1. Execute the KEM on the input (sk_s, pk_r). It outputs a random symmetric key K and an encapsulation of that key C_1.
2. Encrypt the message m under the key K to produce a ciphertext C_2 using the DEM.
3. Output the signcryption (C_1, C_2).

In this case, an inside attacker who is able to obtain a valid signcryption (C_1, C_2) can forge a signcryption on any message m by recovering a symmetric key K from C_1 (using pk_s and sk_r), and encrypting the message m using the DEM and the symmetric key K.

For a hybrid signature scheme, the solution was to provide the KEM's encapsulation and decapsulation algorithm with the message as an extra input. However, for a hybrid signcryption scheme, we cannot provide the KEM's decapsulation oracle with the message as input, because the message has not yet been recovered at the time that we execute the decapsulation algorithm. On the other hand, it is necessary to make sure that the symmetric key used for decryption is related to the message being decrypted or we may still apply the simple forgery attack described above. We therefore define an insider secure KEM and DEM as follows.

Definition 7 (Signcryption KEM). *An (insider secure) signcryption KEM is a 6-tuple of algorithms:*

1. *A probabilistic common key generation algorithm, Gen_c. It takes as input a security parameter 1^k and returns some global information (parameters) I.*
2. *A probabilistic sender key generation algorithm, Gen_s. It takes as input the global information I and outputs a public/private key pair (pk_s, sk_s) for a party who wishes to send a signcrypted message.*
3. *A probabilistic receiver key generation algorithm, Gen_r. It takes as input the global information I and outputs a public/private key pair (pk_r, sk_r) for a party who wishes to be able to receive signcrypted messages.*
4. *A probabilistic key encapsulation algorithm, Encap. It takes as input a sender's private key sk_s, a receiver's public key pk_r and a message m; and outputs a symmetric key K and an encapsulation of that key C. We denote this as $(K, C) = Encap(sk_s, pk_r, m)$.*
5. *A deterministic key decapsulation algorithm, Decap. It takes as input a sender's public key pk_s, a receiver's private key sk_r and an encapsulation of a key C; and outputs either a symmetric key K or the error symbol $\perp$. We denote this as $K = Decap(pk_s, sk_r, C)$.*

6. *A deterministic verification algorithm, Ver. It takes as input a sender's public key pk_s, a receiver's private key sk_r, a message m, and an encapsulation C; and outputs either* `valid` *or* `invalid`*. We denote this algorithm as $Ver(pk_s, sk_r, m, C)$. Note that the verification algorithm does not need to take the symmetric key K as input as it can be easily computed from the encapsulation C using the deterministic decapsulation algorithm.*

We require that the decapsulation algorithm is sound, i.e. for almost all valid sender key-pairs (pk_s, sk_s), receiver key-pairs (pk_r, sk_r) and messages m then $K = Decap(pk_s, sk_r, C)$ for almost all pairs $(K, C) = Encap(sk_s, pk_r, m)$. We also require that the verification algorithm is sound, i.e. for almost all sender key-pairs (pk_s, sk_s), receiver key-pairs (pk_r, sk_r), messages m and encapsulations $(K, C) = Encap(sk_s, pk_r, m)$ then $Ver(pk_s, sk_r, m, C) =$ `valid`*.*

Definition 8 (Signcryption DEM). *A signcryption DEM consists of two polynomial-time algorithms:*

1. *A deterministic encryption algorithm,* Enc*, which takes as input a message $m \in \{0,1\}^*$ of any length and a symmetric key K of some pre-determined length, and outputs an encryption $C = \text{Enc}_K(m)$ of that message.*
2. *A deterministic decryption algorithm,* Dec*, which takes as input a ciphertext $C \in \{0,1\}^*$ of any length and a symmetric key K of some pre-determined length, and outputs either a message $m = \text{Dec}_K(C)$ or the error symbol $\perp$.*

We require that any signcryption DEM be sound in the sense that, for every key K of the correct length, $m = \text{Dec}_K(\text{Enc}_K(m))$.

We define a hybrid signcryption algorithm composed of a signcryption KEM and DEM in the following way.

Definition 9 (KEM-DEM hybrid signcryption scheme). *Suppose that $(Gen_c, Gen_s, Gen_r, Encap, Decap, Ver)$ is a signcryption KEM,* (Enc, Dec) *is a signcryption DEM, and that, for all security parameters k, the keys produced by the signcryption KEM are of the correct length to be used by the signcryption DEM. We may then construct a signcryption scheme $(\mathcal{G}_c, \mathcal{G}_s, \mathcal{G}_r, \mathcal{E}, \mathcal{D})$ as follows.*

- *The key generation algorithms $(\mathcal{G}_c, \mathcal{G}_s, \mathcal{G}_r)$ are given by the key generation algorithms for the signcryption KEM (Gen_c, Gen_s, Gen_r).*
- *The action of a generation-encryption algorithm $\mathcal{E}$ on a message m, a sender's private key sk_s and a receiver's public key pk_r is given by:*
  1. *Set $(K, C_1) = Encap(sk_s, pk_r, m)$.*
  2. *Set $C_2 = \text{Enc}_K(m)$.*
  3. *Output (C_1, C_2).*
- *The action of a verification-decryption algorithm $\mathcal{D}$ on a signcryption (C_1, C_2), a sender's public key pk_s and a receiver's private key sk_r is given by:*
  1. *Set $K = Decap(pk_s, sk_r, C_1)$. If $K = \perp$ then output $\perp$ and stop.*
  2. *Set $m = \text{Dec}_K(C_2)$. If $m = \perp$ then output $\perp$ and stop.*
  3. *If $Ver(pk_s, sk_r, m, C_1) =$* `valid` *then output m. Otherwise output $\perp$.*

This construction is sound due to the soundness of the signcryption KEM and DEM.

5 The Security Criteria for a Signcryption KEM

In this section we will develop independent security criteria for a signcryption KEM with insider security.

Confidentiality

A signcryption KEM must satisfy a similar condition to that satisfied by an encryption KEM [5,9]. We define the IND-CCA2 game as a game played between a hypothetical challenger and a two stage attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. For a given security parameter k, the game is played as follows.

1. The challenger generates some public parameters $I = Gen_c(1^k)$, a sender key-pair $(pk_s, sk_s) = \mathcal{G}_s(I)$ and a receiver key-pair $(pk_r, sk_r) = \mathcal{G}_r(I)$.
2. The attacker runs $\mathcal{A}_1$ on the input (pk_s, pk_r). During its execution $\mathcal{A}_1$ can query an encapsulation oracle that will, when given a message m, return $Encap(sk_s, pk_r, m)$; a decapsulation oracle that will, when given an encapsulation C, return $Decap(pk_s, sk_r, C)$; and a verification oracle that will, when given an encapsulation C and a message m, return $Ver(pk_s, sk_r, m, C)$. $\mathcal{A}_1$ terminates by outputting a message m^* and some state information *state*.
3. The challenger computes the challenge signcryption as follows.
   (a) Set $(K_0, C^*) = Encap(sk_s, pk_r, m^*)$.
   (b) Randomly generate a symmetric K_1 of the same length as K_0.
   (c) Randomly generate a bit $b \in \{0, 1\}$.
   (d) Return (K_b, C^*) to the attacker.
4. The attacker executes $\mathcal{A}_2$ on the input (K^*, C^*) and *state*. During its execution $\mathcal{A}_2$ can query an encapsulation, decapsulation and verification oracle as above, with the exception that $\mathcal{A}_2$ cannot query the decapsulation oracle on the input C^*. $\mathcal{A}_2$ terminates by outputting a guess b' for b.

The attacker wins the game if $b = b'$. $\mathcal{A}$'s advantage in winning the IND-CCA2 game is defined to be:

$$|Pr[b = b'] - 1/2| \,. \tag{2}$$

Definition 10. *A signcryption KEM with insider security is IND-CCA2 secure if, for all polynomial-time attackers $\mathcal{A}$, that attacker's advantage in winning the IND-CCA2 game is negligible as a function of the security parameter k.*

However, now, along with making sure that the keys that the signcryption KEM produces are suitably random, we must also now protect against the threat that a signcryption KEM leaks information about the message directly. To do this, we define a new game, the INP-CCA2 game[5] which states that an attacker cannot, given an encapsulation, distinguish between two messages that may have been used to produce it.

Formally, we define the INP-CCA2 game as a game played between a hypothetical challenger and a two-stage attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. For a given security parameter k, the game is played as follows.

[5] Here INP stands for "input".

1. The challenger generates some public parameters $I = Gen_c(1^k)$, a sender key-pair $(pk_s, sk_s) = \mathcal{G}_s(I)$ and a receiver key-pair $(pk_r, sk_r) = \mathcal{G}_r(I)$.
2. The attacker runs $\mathcal{A}_1$ on the input (pk_s, pk_r). During its execution $\mathcal{A}_1$ can query an encapsulation oracle that will, when given a message m, return $Encap(sk_s, pk_r, m)$; a decapsulation oracle that will, when given an encapsulation C, return $Decap(pk_s, sk_r, C)$; and a verification oracle that will, when given an encapsulation C and a message m, return $Ver(pk_s, sk_r, m, C)$. $\mathcal{A}_1$ terminates by outputting two messages m_0 and m_1, and some state information *state*.
3. The challenger computes the challenge signcryption as follows.
   (a) Randomly generate a bit $b \in \{0, 1\}$.
   (b) Set $(K_b, C_b) = Encap(sk_s, pk_r, m_b)$.
   (c) Return C_b to the attacker.
4. The attacker executes $\mathcal{A}_2$ on the input C^* and *state*. During its execution $\mathcal{A}_2$ can query an encapsulation, decapsulation and verification oracle as above, with the exception that $\mathcal{A}_2$ cannot query the decapsulation oracle on the input C^* or verification oracle on the inputs (m_0, C^*) or (m_1, C^*).

The attacker wins the game if $b = b'$. $\mathcal{A}$'s advantage in winning the INP-CCA2 game is defined to be:

$$|Pr[b = b'] - 1/2| \,. \tag{3}$$

Definition 11. *A signcryption KEM with insider security is INP-CCA2 secure if, for all polynomial-time attackers $\mathcal{A}$, that attacker's advantage in winning the INP-CCA2 game is negligible as a function of the security parameter k.*

Integrity/Authenticity

It is clear that if an attacker, equipped with knowledge of pk_s, pk_r and sk_r, can determine a KEM encapsulation C_1 and a message m such that

- $Decap(pk_s, sk_r, C_1) = K \neq \perp$,
- $Ver(pk_s, sk_r, m, C_1) = \texttt{valid}$, and
- C_1 was never the response from the KEM encapsulation oracle queried on the message m,

then that attacker can use the encapsulation C_1 to forge a new signcryption (C_1, C_2) of the message m by computing $C_2 = \textsc{Enc}_K(m)$. However, if we insist that a scheme is only secure if an attacker cannot find such a message/encapsulation pair, then we can deduce that the KEM encapsulation algorithm must be acting as a signature scheme, where the component algorithms if the signature scheme are as follows.

- Key generation is performed as follows.
  1. Set $I = Gen_c(1^k)$.
  2. Set $(pk_s, sk_s) = Gen_s(I)$.
  3. Set $(pk_r, sk_r) = Gen_r(I)$.

4. Output the private signing key $sk = (sk_s, pk_r)$ and the public verification key (pk_s, sk_r).

– The signature σ of a message m computed using a private signing key (sk_s, pk_r) is given by setting $\sigma = C$ where $(K, C) = Encap(sk_s, pk_r, m)$.
– A signature σ of a message m is verified using a public verification key (pk_s, sk_r) as follows.
  1. Set $K = Decap(pk_s, sk_r, C)$. If $K = \perp$ then output `invalid` and halt.
  2. Output $Ver(pk_s, sk_r, m, C)$.

Hence, any hybrid signcryption scheme with insider security must be using some kind of combination of a signature scheme (from which we somehow manage to derive a symmetric key) and a symmetric encryption scheme. As a by-product we note that if the KEM is acting as a signature scheme then it is implicitly providing an integrity/authentication service for the message m; therefore, the DEM is only required to provide a confidentiality service for the message.

We define the integrity security criterion for a KEM in terms of a game played between an attacker $\mathcal{A}$ and a hypothetical challenger. This game is identical to the game that would define the security of the KEM acting as a signature scheme. For a given security parameter k, the game runs as follows.

1. The challenger generates some valid parameters I by running $Gen_c(1^k)$; a valid sender key pair (pk_s, sk_s) by running the sender key generation algorithm $Gen_s(I)$; and a valid receiver key pair (pk_r, sk_r) by running the receiver key generation algorithm $Gen_r(I)$.
2. The attacker executes $\mathcal{A}$ on the input (pk_s, pk_r, sk_r). During its execution $\mathcal{A}$ can query an encapsulation oracle that will, when given a message m, output an encapsulation $(K, C) = Encap(sk_s, pk_r, m)$. $\mathcal{A}$ terminates by outputting a pair (m^*, C^*).

The attacker wins the game if $Decap(pk_s, sk_r, C^*) = K \neq \perp$, $Ver(pk_s, sk_r, m^*, C^*)$ outputs `valid`, and C^* was never the response from the encapsulation oracle queried on the message m. Note that we do not have to give the attacker explicit access to a decapsulation or verification oracle because they already know sk_r and can therefore compute these functions themselves.

Definition 12. *A signcryption KEM is INT-CCA2 secure if, for all polynomial-time attackers $\mathcal{A}$, the probability that $\mathcal{A}$ wins the INT-CCA2 game is negligible as a function of the security parameter k.*

Putting this all together we have:

Definition 13. *A signcryption KEM is said to be insider secure if it is IND-CCA2, INP-CCA2 and INT-CCA2 secure.*

6 The Security Criterion for a Signcryption DEM

As we have already noted, in a hybrid signcryption scheme with insider security, the KEM will be providing the integrity, origin authentication and non-repudiation services, so the DEM will only be required to provide a simple

confidentiality service. Indeed, the notion of security against passive attacks developed by Cramer and Shoup [5] is sufficient to provide security, with one slight exception.

Security in this model is phrased in terms of a game between a challenger and a two-stage attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. It runs as follows:

1. The challenger randomly generates a symmetric key K of the appropriate length for use with the symmetric encryption scheme.
2. The attacker runs $\mathcal{A}_1$ on the input 1^k. The algorithm $\mathcal{A}_1$ terminates by outputting a pair of (equal length) messages (m_0, m_1), as well as some state information *state*.
3. The challenger chooses a bit $b \in \{0, 1\}$ uniformly at random, and forms the challenge ciphertext $C^* = \text{ENC}_K(m_b)$.
4. The attacker runs $\mathcal{A}_2$ on the input $(C^*, state)$. This algorithm outputs a guess b' for b.

The attacker wins the game if $b = b'$. The attacker's advantage in winning this game is defined to be:

$$|Pr[b = b'] - 1/2| \,. \tag{4}$$

Definition 14. *A DEM is said to be secure against passive attacks if, for all polynomial time attackers $\mathcal{A}$, the advantage that $\mathcal{A}$ has in winning the above game is negligible as a function of the security parameter k.*

Since we require a signcryption scheme to have strong unforgeability, we actually require another property from the DEM. We require that the decryption algorithm is one-to-one[6], i.e. we require that, for any symmetric key K,

$$\text{DEC}_K(C_2) = \text{DEC}_K(C_2') \text{ if and only if } C_2 = C_2' \,. \tag{5}$$

This prevents an attacker from creating a forgery (C_1, C_2') from a signcryption (C_1, C_2) by finding another DEM encryption C_2' from the ciphertext C_2.

We can now state our main results.

Theorem 1 (Confidentiality). *Suppose a hybrid signcryption scheme is composed of a signcryption KEM and a signcryption DEM. If the signcryption KEM is insider secure and the DEM is secure against passive attacks and has a one-to-one decryption function, then the overall signcryption scheme is IND-CCA2 secure.*

Theorem 2 (Integrity/Authenticity). *Suppose a hybrid signcryption scheme is composed of a signcryption KEM and a signcryption DEM. If the signcryption KEM is INT-CCA2 secure and the DEM has a one-to-one decryption function, then the overall signcryption scheme is INT-CCA secure.*

The proofs of both of these theorems can be found in the full version of this paper [6].

[6] Technically, we actually only require a weaker condition: that the decryption algorithm is computationally one-to-one. I.e., that no polynomial-time attacker can find a second ciphertext C_2' given C_2.

7 An Example of a Signcryption KEM

In order to provide an example of a signcryption KEM with insider security, we come full circle back to the original signcryption scheme proposed by Zheng [10]. We present the provably secure variant of Zheng's scheme proposed by Baek, Steinfeld and Zheng [3] as a KEM-DEM signcryption scheme.

- *Common key generation algorithm.* This algorithm takes as input the security parameter 1^k and outputs a triple (p, q, g) where p is a large prime, q is a large prime that divides $p-1$ and g is an element of $\mathbb{Z}_p^*$ of order q.
- *Sender key generation algorithm.* This algorithm chooses an integer $1 \leq s \leq q$ uniformly at random, sets $P_s = g^s \bmod p$ then outputs the public key (p, q, g, P_s) and the private key (p, q, g, s).
- *Receiver key generation algorithm.* This algorithm chooses an integer $1 \leq r \leq q$ uniformly at random, sets $P_r = g^r \bmod p$ then outputs the public key (p, q, g, P_r) and the private key (p, q, g, r).
- *Encapsulation algorithm.* This algorithm works as follows.
  1. Choose an element $1 \leq t \leq q$ uniformly at random.
  2. Set $X = P_r^t \bmod p$.
  3. Set $R = Hash_1(m||X)$.
  4. Set $S = t/(R+s) \bmod q$.
  5. Set $K = Hash_2(X)$.
  6. Set $C = (R, S)$.
  7. Output (K, C).
- *Decapsulation algorithm.* This algorithm works as follows.
  1. Parse C as (R, S).
  2. Set $X = (P_s \cdot g^R)^{Sr} \bmod p$.
  3. Output $K = Hash_2(X)$.
- *Verification algorithm.* This algorithm works as follows.
  1. Parse C as (R, S).
  2. Set $X = (P_s \cdot g^R)^{Sr} \bmod p$.
  3. Check that $Hash_1(m||X) = R$. If not, output `invalid` and halt.
  4. Otherwise output `valid`.

Of course, in a real implementation of this algorithm, there is no advantage in computing X in both the decapsulation and verification algorithm. A real implementation would merely store the value of X computed by the decapsulation algorithm and use it again in the verification algorithm. Such an implementation would be functionally identical to the above algorithm and would therefore be just as secure. We choose to separate the decapsulation and verification algorithms so that they can be studied independently.

The proofs of security for this algorithm can be adapted from those presented by Baek, Steinfeld and Zheng [3]. The scheme is secure in the random oracle model, under the Gap Diffie-Hellman assumption.

8 Conclusion

We have demonstrated that a signcryption scheme with insider security can be constructed using a hybrid approach, although this construction is significantly more complex than the "standard" hybrid construction. Furthermore, we have shown that the original signcryption scheme proposed by Zheng can be thought of as a hybrid signcryption scheme. Indeed, an examination of the literature shows that most signcryption schemes with insider security can be thought of as hybrid schemes in this form. This poses an interesting question: is it possible to construct a hybrid signcryption scheme that does not conform to the general model presented?

Acknowledgements

The author would like to thank John Malone-Lee, Liqun Chen, Fred Piper, Bodo Möller, Yevgeniy Dodis and Stéphanie Alt for their helpful comments. The author would also like to thank the anonymous reviewers for helpful comments. The author gratefully acknowledges the financial support of the EPSRC.

References

1. J. H. An. Authenticated encryption in the public-key setting: Security notions and analyses. Available from `http://eprint.iacr.org/2001/079`, 2001.
2. J. H. An, Y. Dodis, and T. Rabin. On the security of joint signature and encryption. In L. Knudsen, editor, *Advances in Cryptology – Eurocrypt 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 83–107. Springer-Verlag, 2002.
3. J. Baek, R. Steinfeld, and Y. Zheng. Formal proofs for the security of signcryption. In D. Naccache and P. Pallier, editors, *Public Key Crytpography 2002 (PKC 2002)*, volume 2274 of *Lecture Notes in Computer Science*, pages 80–98. Springer-Verlag, 2002.
4. M. Bellare and P. Rogaway. The exact security of digital signatures — how to sign with RSA and Rabin. In U. Maurer, editor, *Advances in Cryptology – Eurocrypt '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 399–416. Springer-Verlag, 1996.
5. R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2004.
6. A. W. Dent. Hybrid cryptography. Available from `http://eprint.iacr.org/2004/210/`, 2004.
7. A. W. Dent. Hybrid signcryption schemes with outsider security (extended abstract). Available from `http://www.isg.rhul.ac.uk/~alex/`, 2004.
8. C. P. Schnorr. Efficient signature generation for smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.
9. V. Shoup. Using hash functions as a hedge against chosen ciphertext attack. In B. Preneel, editor, *Advances in Cryptology – Eurocrypt 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 275–288. Springer-Verlag, 2000.
10. Y. Zheng. Digital signcryption or how to achieve cost(signature & encryption) << cost(signature) + cost(encryption). In B. Kaliski, editor, *Advances in Cryptology – Crypto '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 165–179. Springer-Verlag, 1997.

On the Possibility of Constructing Meaningful Hash Collisions for Public Keys

Arjen Lenstra[1,2] and Benne de Weger[2]

[1] Lucent Technologies, Bell Laboratories, Room 2T-504
600 Mountain Avenue, P.O.Box 636, Murray Hill, NJ 07974-0636, USA
[2] Technische Universiteit Eindhoven
P.O.Box 513, 5600 MB Eindhoven, The Netherlands

Abstract. It is sometimes argued that finding meaningful hash collisions might prove difficult. We show that for several common public key systems it is easy to construct pairs of meaningful and secure public key data that either collide or share other characteristics with the hash collisions as quickly constructed by Wang et al. We present some simple results, investigate what we can and cannot (yet) achieve, and formulate some open problems of independent interest. We are not yet aware of truly interesting practical implications. Nevertheless, our results may be relevant for the practical assessment of the recent hash collision results. For instance, we show how to construct two different X.509 certificates that contain identical signatures.

1 Introduction

Based on the birthday paradox a random collision for any n-bit hash function can be constructed after an effort proportional to $2^{n/2}$ hash applications, no matter how good the hash function is. From the results presented at the Crypto 2004 rump session (cf. [14]), and since then described in more detail in [15], [16], [17], and [18], it follows that for many well known hash functions the effort required to find random collisions is considerably lower. Indeed, in some cases the ease with which collisions can be found is disconcerting. Their application for integrity protection of binary data and in digital certificates, however, is still rather common. In particular MD5, one of the affected hash functions, is still being used by Certification Authorities to generate new certificates.

We sketch the commonly used arguments why such applications are not affected by the lack of random collision resistance. In this note we concentrate on applications in the area of public key cryptography, see [4] and [9] for interesting ideas about the application of hash collisions in other areas.

A successful attack on an existing certificate requires *second preimage resistance* of one message: given a pre-specified value and its hash, it must be practically infeasible to find another value with the same hash. As far as we are aware, the results announced in [14] do not imply that second preimages are essentially easier to find than they should, namely with an effort proportional

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 267–279, 2005.

to 2^n for an n-bit hash function. Therefore certificates that existed well before the results from [14] were obtained should be fine.

For newly to be constructed certificates the argument goes that random collisions do not suffice because the values to be hashed are *meaningful* (cf. [3] and [11]). Dobbertin's cryptanalytic work on MD4 was so strong that meaningful collisions could be found easily, cf. [2]. The recent results of [14] seem not (yet) to have similar strength, so revisiting the concept of meaningfulness is of interest.

A certificate, such as an X.509 or PGP certificate, is a highly structured document. Nevertheless, it contains pieces of data that look random, and may have been constructed to fit a hash collision. In particular there will be random looking binary data related to public keys. Also the Diffie-Hellman group size may be related to a random-looking large prime, which is a system parameter that could be hard-coded into a binary executable. As was shown in [5], given any hash collision it is trivial to construct a 'real' Diffie-Hellman prime and a 'fake' one that hash to the same value. One may ask whether the mathematical requirements that lie behind public key constructions enforce so much meaningful structure that it may be expected to be incompatible with the collision requirement. We show that this is not the case.

The collisions found by [14] all have a special structure: two inputs are found that hash to the same value, and that differ in a few spread-out and precisely specified bit positions only. This leads us to the following question. Suppose the value to be hashed contains an RSA modulus, i.e., a hard to factor composite integer, or an element g^v for a (sub)group generator g and secret exponent v. How would one come up with two different RSA moduli or two different powers of g that have the subtle differences that seem to be required in [14]?

Having the right type of difference structure does not, as far as we know, imply a hash collision. Presently, specially crafted data blocks seem to be required for collisions. But colliding data blocks can be used to generate more collisions as follows. All affected hash functions are based on the *Merkle-Damgård construction*, where a *compression function* is iteratively applied to a changing *chaining variable*. New collisions can therefore be constructed by appending arbitrary, but identical, data to any existing pair of colliding data consisting of the same number of blocks. We show how this may be used to construct well-formed and secure public keys with identical hash values.

Apparently, colliding data blocks can be found for the compression function with an arbitrary value of the chaining variable. This implies that identical data can also be prepended to colliding pairs if the resulting data have the same length and the colliding pairs have been specifically crafted to work with the chaining variable value that results from the prepended data.

In this paper we investigate the various problems and possibilities. We show how we can generate public keys with prescribed differences but with a priori unknown most significant parts. Even though the resulting public keys will, in general, not collide, it cannot be excluded, and it can indeed be expected, that in the future new collision methods will be found that have different, less severe restrictions. Therefore it is relevant to know if the two requirements—being

meaningful and having the proper difference structure—are mutually exclusive or not, and if not if examples can be constructed in a reasonable amount of time. We address this question both for RSA and for discrete logarithm systems. We explicitly restrict ourselves to *known* and *secure* private keys as the construction of unknown or non-secure private keys is hardly challenging (cf. [12]).

Furthermore, using the appending trick, we show how we can generate actually colliding pairs consisting of proper public RSA keys. Combining this construction with the prepending idea, we show how different X.509 certificates can be constructed that have identical signatures. It is conceivable that such certificate 'pairs' may be used for ulterior purposes.

We are not aware yet of real life practical implications of our results. Our sole goal is to point out that the 'meaningful message' argument against hash collisions in certification applications may be weaker than it appears at first sight.

A summary of our results is as follows. It is straightforward to generate secure pairs of RSA moduli with any small difference structure. Furthermore, in Section 2 it is shown how any actual Merkle-Damgård based hash collision can be used to construct colliding pairs consisting of two hard to factor moduli, and how such moduli can be embedded in X.509 certificates with identical signatures. For discrete logarithm systems there is a much greater variety of results, and even some interesting open questions. Briefly, one can do almost anything one desires if one may pick any generator of the full multiplicative group, but if a prescribed generator, or a subgroup generator, has to be used, then we cannot say much yet. Our observations are presented in Section 3. Some attack scenarios and applications that use our constructions are sketched in Section 4. The full version [7] of this paper contains some more detail, an additional section on the practicality of generating colliding DL system parameters, à la Kelsey and Laurie [5], and an appendix giving full details of our construction of colliding X.509 certificates.

2 Generating Pairs of Hard to Factor Moduli

The first problem we address in this section is constructing pairs of RSA public key values that differ in a prescribed small number of bit positions. The second problem is constructing pairs of colliding hard to factor moduli, with an application to the construction of pairs of X.509 certificates.

An RSA public key value ordinarily consists of an RSA modulus and a public exponent. A single RSA modulus with two different public exponents that differ in the prescribed way is in principle a solution to the first problem. But in practice one often fixes the public exponent, and selecting two proper public exponents that differ in the right way is trivial.

The first problem: RSA moduli with prescribed difference. We address the more interesting problem where the public exponent is fixed and where the two RSA moduli differ in the prescribed bit positions. The latter is the case if the XOR of the regular binary representations of the moduli consists of the

prescribed bits. Unfortunately, the XOR of two integers is not a convenient representation-independent mathematical operation. This slightly complicates matters. If the hamming weight of the prescribed XOR is small, however, the XOR corresponds often enough to the regular, representation-independent integer difference. Therefore a probabilistic method to generate moduli with a prescribed difference may be expected to eventually produce a pair with the right XOR.

Algorithm to generate moduli with prescribed difference. Let $N \in \mathbf{Z}_{>0}$ be an integer indicating the bitlength of the RSA moduli we wish to construct, and let δ be a positive even integer of at most N bits containing the desired difference. We describe a fast probabilistic method to construct two secure N-bit RSA moduli m and n such that $m - n = \delta$. Informally, pick primes p and q at random, use the Chinese Remainder Theorem to find m with $m \equiv 0 \bmod p$ and $m \equiv \delta \bmod q$, and add pq to m until both cofactors m/p and $(m-\delta)/q$ are prime. More formally:

- Let ℓ be a small positive integer that is about $2\log_2(N)$.
- Pick distinct primes p and q of bitlength $N/2 - \ell$, calculate integers $r = \delta/p \bmod q$ and $s = (rp - \delta)/q$, then for any k

$$p(r + kq) - q(s + kp) = \delta.$$

- Search for the smallest integer k such that $r + kq$ and $s + kp$ are both prime and such that $p(r + kq)$ and $q(s + kp)$ both have bitlength N.
- For the resulting k let $m = p(r + kq)$ and $n = q(s + kp)$.
- If k cannot be found, pick another random p or q (or both), recalculate r and s, and repeat the search for k.

Runtime analysis. Because the more or less independent $(N/2 + \ell)$-bit numbers $r + kq$ and $s + kp$ have to be simultaneously prime, one may expect that the number of k's to be searched is close to $(N/2)^2$. Thus, a single choice of p and q should suffice if 2^ℓ is somewhat bigger than $(N/2)^2$, which is the case if $\ell \approx 2\log_2(N)$. The algorithm can be expected to require $O(N^2)$ tests for primality. Depending on the underlying arithmetic and how the primality tests are implemented – usually by means of trial division combined with a probabilistic compositeness test – the overall runtime should be between $O(N^4)$ and $O(N^5)$.

A larger ℓ leads to fewer choices for p and q and thus a faster algorithm, but it also leads to larger size differences in the factors of the resulting RSA moduli m and n. The algorithm can be forced to produce balanced primes (i.e., having the same bitlength) by taking $\ell = 0$, and for instance allowing only $k = 0$, but then it can also be expected to run $O(N)$ times slower.

From prescribed difference to prescribed XOR. If required, and as discussed above, the method presented above may be repeated until the resulting m and n satisfy m XOR $n = \delta$ (where, strictly speaking, m and n in the last equation should be replaced by one's favorite binary representation of m and n). The number of executions may be expected to increase exponentially with the hamming weight $H(\delta)$ of δ. If $H(\delta)$ is small, as apparently required for the type of collisions constructed in [14], this works satisfactorily.

It is much faster, however, to include the test for the XOR condition directly in the algorithm before $r + kq$ and $s + kp$ are subjected to a primality test. In that case ℓ may be chosen about $H(\delta)$ larger to minimize the number of p and q choices, but that also leads to an even larger size difference between the factors. It turns out that the overhead caused by the XOR condition compared to the difference is quite small.

Security considerations. Given two regular RSA moduli m and n, their difference $\delta = |m - n|$ can obviously be calculated. But knowledge of δ and the factorization of one of the moduli, does, with the present state of the art in integer factorization, not make it easier to factor the other modulus, irrespective of any special properties that δ may have. Indeed, if the other modulus could be factored, the RSA cryptosystem would not be worth much. If m is the product of randomly selected primes p and r of the same size, as is the case in regular RSA, then $r = \delta/p \bmod q$ for any other RSA modulus n with prime factor q and $\delta = m - n$. Thus, the randomly selected prime factor r satisfies the same identity that was used to determine r in our algorithm above (given p, q, and δ), but as argued that does not make r easier to calculate given just q and δ (but not p). This shows that the '$\ell = 0$ and allow only $k = 0$' case of our algorithm produces RSA moduli pairs that are as hard to factor as regular RSA moduli, and that knowledge of the factorization of one of them does not reveal any information about the factors of the other.

The same argument and conclusion applies in the case of regular RSA moduli with unbalanced factors: with the present state of the art such factors are not easier to find than others (avoiding factors that are so small that the elliptic curve factoring method would become applicable), also not if the difference with another similarly unbalanced RSA modulus is known. If an N-bit RSA modulus m has an $(N/2 - \ell)$-bit factor p with $(N/2 + \ell)$-bit cofactor $\tilde{r}$, both randomly selected, then $\tilde{r} \bmod q = \delta/p \bmod q$ for any other RSA modulus n with $(N/2-\ell)$-bit prime factor q and $\delta = m - n$. The randomly selected prime factor $\tilde{r}$ when taken modulo q satisfies the same identity that was used to determine r in our algorithm and the cofactor $\tilde{s}$ of q in n, when taken modulo p, satisfies the same identity, with r replaced by $\tilde{r} \bmod q$, that was used to determine s in our algorithm. Because $m - n = \delta$ the integers $\tilde{r}$, r, $\tilde{s}$, and s satisfy $\tilde{r} - r = kq$ and $\tilde{s} - s = kp$ for the same integer valued k. This means that the allegedly hard to find $\tilde{r}$ equals the prime factor $r + kq$ as determined by our algorithm.

Remark on simultaneous versus consecutive construction. The method presented in this section simultaneously constructs two moduli with a prescribed difference. One may wonder if the moduli have to be constructed simultaneously and whether consecutive construction is possible: given a difference δ and an RSA modulus m (either with known or unknown factorization), efficiently find a secure RSA modulus n (and its factorization) such that m XOR $n = \delta$. But if this were possible, any modulus could be efficiently factored given its (easy to calculate) difference δ with m. Thus, it is highly unlikely that moduli with prescribed differences can be constructed both efficiently and consecutively.

The second problem: actually colliding hard to factor moduli. The object of our investigation so far has been to find out if the requirement to be meaningful (i.e., proper RSA moduli) excludes the apparent requirement of a prescribed difference structure. As shown above, that is not the case: proper RSA moduli with any prescribed difference can easily be constructed. A much stronger result would be to construct RSA moduli that actually *do* have the same hash value. We don't know yet how to do this if the two moduli must have factors of approximately equal size, a customary property of RSA moduli. We can, however, construct actually colliding composite moduli that are, with the proper parameter choices, as hard to factor as regular RSA moduli but for which, in a typical application, the largest prime factor is about three times longer than the smallest factor. Unbalanced moduli for instance occur in [13]. Our method combines the ideas mentioned in the introduction and earlier in this section with the construction from [6].

Algorithm to generate actually colliding hard to factor moduli. Let b_1 and b_2 be two bitstrings of equal bitlength B that collide under a Merkle-Damgård based hash function. Following [14], B could be 512 if b_1 and b_2 collide under MD4, or 1024 if they collide under MD5. It is a consequence of the Merkle-Damgård construction that for any bitstring b the concatenations $b_1||b$ and $b_2||b$ also collide. Denoting by $N > B$ the desired bitlength of the resulting moduli, we are thus looking for a bitstring b of length $N - B$ such that the integers m_1 and m_2 represented by $b_1||b$ and $b_2||b$, respectively, are hard to factor composites. Assuming that $N - B$ is sufficiently large, let p_1 and p_2 be two independently chosen random primes such that p_1p_2 has bitlength somewhat smaller than $N - B$. Two primes of bitlength $(N - B)/2 - \log_2(B)$ should do in practice. Using the Chinese Remainder Theorem, find an integer b_0, $0 \le b_0 < p_1p_2$ such that p_i divides $b_i 2^{N-B} + b_0$ for $i = 1, 2$. Finally, look for the smallest integer $k \ge 0$ with $b_0 + kp_1p_2 < 2^{N-B}$ and such that the integers $q_i = (b_i 2^{N-B} + b_0 + kp_1p_2)/p_i$ are prime for $i = 1, 2$. If such an integer k does not exist, select new p_1 and p_2 and try again. The resulting moduli are $m_i = p_iq_i = b_i||b$ for $i = 1, 2$, where $b = b_0 + kp_1p_2$ is to be interpreted as $(N - B)$-bit integer. The security of each modulus constructed in this fashion, though unproven, is argued in [6]; since then no weaknesses in this construction have been published. Since p_1 and p_2 are independent, knowledge of the factorization of one of the moduli does not reveal information about the factorization of the other one. The argument follows the lines of the security argument presented earlier in this section. We do not elaborate.

Remark. Given the restrictions of the MD5-collisions as found by the methods from [14] and [15], our method does not allow us to target 1024-bit moduli that collide under MD5, only substantially larger ones. Asymptotically, with growing modulus size but fixed collision size, the prime factors in the moduli ultimately become balanced. The above method can easily be changed to produce a colliding pair of balanced N-bit RSA modulus and N-bit prime. A variation of our construction leads to moduli $b||b_1$ and $b||b_2$, which may be useful for collision purposes if moduli are represented from least to most significant bit.

Colliding X.509 Certificates. Based on the ideas presented above we have constructed a pair of X.509 certificates that are different only in the hard to factor RSA moduli, but that have the same CA signature. A detailed description of our approach is given in the full version of this note, cf. [7]. Briefly, it works as follows. Based on the initial part of the data to be certified, a value of the MD5 chaining variable is determined. Using this value as initialization vector, a pair of 1024-bit values that collide under MD5 is calculated using the methods from [15]. This collision is used as described above to produce two colliding hard to factor 2048-bit moduli, which then enables the construction of two X.509 certificates with identical signatures. Given the current limitations of the MD5-collision methods from [14] and [15], new MD5-based X.509 certificates for 2048-bit RSA moduli should be regarded with more suspicion than X.509 certificates for 1024-bit RSA moduli.

3 Generating DL Public Keys with Prescribed Difference

The problem. In the previous section RSA moduli were constructed with a prescribed XOR of small hamming weight by looking for sufficiently many pairs of moduli with a particular integer difference. Thus, the XOR-requirement was translated into a regular integer difference because the latter is something that makes arithmetic sense. In this section we want to generate discrete logarithm related public key values with a prescribed small XOR: for a generator g of some multiplicatively written group of known finite order, we want integers a_1 and a_2 (the secret keys) such that g^{a_1} and g^{a_2} (the public values) have a prescribed small XOR. Obviously, g^{a_1} XOR g^{a_2} depends on the way group elements are represented. For most common representations that we are aware of the XOR operation does not correspond to a mathematical operation that we can work with. Elements of binary fields are an exception: there XOR is the same as addition.

Representation of elements of multiplicative groups of finite fields. If $\langle g \rangle$ lives in a multiplicative group of a prime field of characteristic p, the group elements can be represented as non-zero integers modulo p, and the XOR can, probabilistically if $p > 2$ and deterministically if $p = 2$, be replaced by the regular integer difference modulo p, similar to what was done in Section 2. In this case the resulting requirement $g^{a_1} - g^{a_2} = \delta$ even has the advantage that it makes sense mathematically speaking, since the underlying field allows both multiplication and addition. Because of this convenience, multiplicative groups of prime fields is the case we concentrate on in this section. Multiplicative groups of extension fields have the same advantage, and most of what is presented below applies to that case as well.

Representation issues for elements of other types of groups. Other cryptographically popular groups are groups of elliptic curves over finite fields. In this case the group element g^{a_1} to be hashed[3] is represented as some number

[3] Note that we keep using multiplicative notation for the group operation, and that our "g^{a_1}" would more commonly be denoted "$a_1 g$" in the elliptic curve cryptoworld.

of finite field elements that represent the coordinates of certain 'points', either projectively or affinely represented, or in some cases even trickier as just a single coordinate, possibly with an additional sign bit. Given such a representation, it is not always immediately clear how the XOR operation should be translated into an integer subtraction that is meaningful in elliptic curve groups. It is conceivable that, for instance, the integer difference of the x-coordinates allows a meaningful interpretation, again with characteristic 2 fields as a possibly more convenient special case. We leave this topic, and the possibility of yet other groups, for future research.

Restriction to multiplicative groups of prime fields. Unless specified otherwise, in the remainder of this section we are working in the finite field $\mathbf{Z}/p\mathbf{Z}$ with, as usual, multiplication and addition the same as integer multiplication and addition modulo p. The problem we are mostly interested in is: given $\delta \in \mathbf{Z}/p\mathbf{Z}$ find non-trivial solutions to $g^{a_1} - g^{a_2} = \delta$ with $g \in (\mathbf{Z}/p\mathbf{Z})^*$ and integers a_1 and a_2. Several different cases and variants can be distinguished, depending on the assumptions one is willing to make.

Variant I: Prescribed generator g of $(\mathbf{Z}/p\mathbf{Z})^*$ and $\delta \neq 0$. Assume that g is a fixed prescribed generator of $(\mathbf{Z}/p\mathbf{Z})^*$ and that $\delta \neq 0$. Obviously, if the discrete logarithm problem in $\langle g \rangle = (\mathbf{Z}/p\mathbf{Z})^*$ can be solved, $g^{a_1} - g^{a_2} = \delta$ can be solved as well: a solution with any desired non-zero value $z = a_1 - a_2$ can be targeted by finding the discrete logarithm a_2 with respect to g of $\delta/(g^z - 1)$, i.e., a_2 such that $g^{a_2} = \delta/(g^z - 1)$. It follows that there are about p different solutions to $g^{a_1} - g^{a_2} = \delta$.

The other way around, however, is unclear: if $g^{a_1} - g^{a_2} = \delta$ can be solved for a_1 and a_2, can the discrete logarithm problem in $\langle g \rangle = (\mathbf{Z}/p\mathbf{Z})^*$ be solved? Annoyingly, we don't know. Intuitively, the sheer number of solutions to $g^{a_1} - g^{a_2} = \delta$ for fixed δ and g seems to obstruct all attempts to reduce the discrete logarithm problem to it. This is illustrated by the fact that if the $g^{a_1} - g^{a_2} = \delta$ oracle would produce solutions a_1, a_2 with fixed $z = a_1 - a_2$, the reduction to the discrete logarithm problem becomes straightforward: to solve $g^y = x$ for y (i.e., given g and x), apply the $g^{a_1} - g^{a_2} = \delta$ oracle to $\delta = (g^z - 1)x$ and set y equal to the resulting a_2.

Lacking a reduction for the general case (i.e., non-fixed $a_1 - a_2$) from the discrete logarithm problem, neither do we know if, given δ and g, solving $g^{a_1} - g^{a_2} = \delta$ for a_1 and a_2 is easy. We conjecture that the problem is hard, and pose the reduction from the regular discrete logarithm problem to it as an interesting open question.

Summarizing, if $\delta \neq 0$ and g is a given generator of the full multiplicative group modulo p, the problem of finding a_1, a_2 with $g^{a_1} - g^{a_2} = \delta$ is equivalent to the discrete logarithm problem in $\langle g \rangle$ if $a_1 - a_2$ is fixed, and the problem is open (but at most as hard as the discrete logarithm problem) if $a_1 - a_2$ is not pre-specified.

Variant II: Prescribed generator g of a true subgroup of $(\mathbf{Z}/p\mathbf{Z})^*$ and $\delta \neq 0$. Let again $\delta \neq 0$, but now let g be a fixed prescribed generator of a true subgroup of $(\mathbf{Z}/p\mathbf{Z})^*$. For instance, g could have order q for a sufficiently large

prime divisor q of $p-1$, in our opinion the most interesting case for the hash collision application that we have in mind. If $z = a_1 - a_2$ is pre-specified, not much is different: a solution to $g^{a_1} - g^{a_2} = \delta$ exists if $\delta/(g^z - 1) \in \langle g \rangle$ and if so, it can be found by solving a discrete logarithm problem in $\langle g \rangle$, and the discrete logarithm problem $g^y = x$ given an $x \in \langle g \rangle$ can be solved by finding a fixed $z = a_1 - a_2$ solution to $g^{a_1} - g^{a_2} = (g^z - 1)x$.

But the situation is unclear if a_1 and a_2 may vary independently: we do not even know how to establish whether or not a solution exists. We observe that for the cryptographically reasonable case where g has prime order q, with q a 160-bit prime dividing a 1024-bit $p-1$, the element $g^{a_1} - g^{a_2}$ of $\mathbf{Z}/p\mathbf{Z}$ can assume at most $q^2 \approx 2^{320}$ different values. This means that the vast majority of unrestricted choices for δ is infeasible and that a δ for which a solution would exist would have to be constructed with care. However, the δ's that we are interested in have low hamming weight. This makes it exceedingly unlikely that a solution exists at all. For instance, for $H(\delta) = 6$ there are fewer than 2^{51} different δ's. For each of these δ we may assume that it is of the form $g^{a_1} - g^{a_2}$ with probability at most $\approx 2^{320}/2^{1024}$. Thus, with overwhelming probability, none of the δ's will be of the form $g^{a_1} - g^{a_2}$. And, even if one of them has the proper form, we don't know how to find out.

Variant III: Free choice of generator of $(\mathbf{Z}/p\mathbf{Z})^*$ and $\delta \neq 0$. Now suppose that just $\delta \neq 0$ is given, but that one is free to determine a generator g of $(\mathbf{Z}/p\mathbf{Z})^*$, with p either given or to be determined to one's liking. Thus, the problem is solving $g^{a_1} - g^{a_2} = \delta$ for integers a_1 and a_2 and a generator g of the multiplicative group $(\mathbf{Z}/p\mathbf{Z})^*$ of a prime field $\mathbf{Z}/p\mathbf{Z}$. Not surprisingly, this makes finding solutions much easier. For instance, one could look for a prime p and small integers u and v such that the polynomial $X^u - X^v - \delta \in (\mathbf{Z}/p\mathbf{Z})[X]$ has a root $h \in (\mathbf{Z}/p\mathbf{Z})^*$ (for instance, by fixing $u = 2$ and $v = 1$ and varying p until a root exists). Next, one picks a random integer w coprime to $p-1$ and calculates $g = h^{1/w}$, $a_1 = uw \bmod (p-1)$, and $a_2 = vw \bmod (p-1)$. As a result $g^{a_1} - g^{a_2} = \delta$. With appropriately chosen p it can quickly be verified if g is indeed a generator; if not, one tries again with a different w or p, whatever is appropriate.

Obviously, this works extremely quickly, and solutions to $g^{a_1} - g^{a_2} = \delta$ can be generated on the fly. The disadvantage of the solution is, however, that any party that knows a_1 (or a_2) can easily derive a_2 (or a_1) because $va_1 = ua_2 \bmod (p-1)$ for small u and v. In our 'application' this is not a problem if one wants to spoof one's own certificate. Also, suspicious parties that do not know either a_1 or a_2 may nevertheless find out that g^{a_1} and g^{a_2} have matching small powers. It would be much nicer if the secrets (a_1 and a_2) are truly independent, as is the case for our RSA solution. We don't know how to do this. Similarly, we do not know how to efficiently force g into a sufficiently large but relatively small (compared to p) subgroup.

Variant IV: Two different generators, any δ. In our final variant we take g again as a generator of $(\mathbf{Z}/p\mathbf{Z})^*$, take any $\delta \in \mathbf{Z}/p\mathbf{Z}$ including $\delta = 0$, and ask for a solution h, a_1, a_2 to $g^{a_1} - h^{a_2} = \delta$. Obviously, this is trivial, even if a_1

is fixed or kept secret by hiding it in g^{a_1}: for an appropriate a_2 of one's choice compute h as the a_2th root of $g^{a_1} - \delta$. For subgroups the case $\delta \neq 0$ cannot be expected to work, as argued above.

The most interesting application of this simple method is the case $\delta = 0$. Not only does $\delta = 0$ guarantee a hash collision, it can be made to work in any group or subgroup, not just the simple case $(\mathbf{Z}/p\mathbf{Z})^*$ we are mostly considering here, and g and h may generate entirely different (sub)groups, as long as the representations of the group elements is sufficiently 'similar': for instance, an element of $(\mathbf{Z}/p\mathbf{Z})^*$ can be interpreted as an element of $(\mathbf{Z}/p'\mathbf{Z})^*$ for any $p' > p$, and most of the time vice versa as long as $p' - p$ is relatively small. Because, furthermore, just g^{a_1} but not a_1 itself is required, coming up with one's own secret exponent and generator (possibly of another group) seems to be the perfect way to spoof someone else's certificate on g^{a_1}. It follows that in practical cases of discrete logarithm related public keys, information about the generator and (sub)group (the *system parameters*) must be included in the certificate or that the system parameters must be properly authenticated in some other way.

This illustrates once more that one should never trust a generator whose construction method is not specified, since it may have been concocted to collide, for some exponents, with a 'standard' or otherwise prescribed generator. This has been known for a long time, cf. [10] and [1], and, according to [19], this issue came up in the P1363 standards group from time to time. Nevertheless it still seems to escape the attention of many implementors and practitioners.

Remark on actually colliding powers of a fixed g. As shown above, $\delta = 0$ and the freedom to select a generator makes it trivial to generate actually colliding powers. One may wonder if less straightforward examples with a fixed generator g can be constructed in a way similar to the construction shown at the end of Section 2. Let N be such that the elements of $\langle g \rangle$ can be represented as bitstrings of length N, and let (b_1, b_2) be a pair of B-bit values that collide under a Merkle-Damgård hash. The question is if an $(N-B)$-bit value b and integers a_1 and a_2 can be found such that the colliding values $b_1||b$ and $b_2||b$ satisfy $b_1||b = g^{a_1}$ and $b_2||b = g^{a_2}$. We don't know how to do this – except that it can be done in any number of ways if discrete logarithms with respect to g can be computed. The ability to solve Variant I, however, makes it possible to solve the related problem of finding b such that $b_1 2^{N-B} + b = g^{a_1}$ and $b_2 2^{N-B} + b = g^{a_2}$: simply take $\delta = (b_1 - b_2)2^{N-B}$, apply Variant I to find a_1 and a_2 with $g^{a_1} - g^{a_2} = \delta$ and define $b = g^{a_1} - b_1 2^{N-B}$, which equals $g^{a_2} - b_2 2^{N-B}$. Unfortunately, the resulting b will in general not be an $(N - B)$-bit value, so that the '+' cannot be interpreted as '||', and the resulting pair (g^{a_1}, g^{a_2}) will most likely no longer collide.

4 Attack Scenarios and Applications

We describe some possible (ab)uses of colliding public keys. None of our examples is truly convincing, and we welcome more realistic scenarios.

One possible scenario is that Alice generates colliding public keys for her own use. We assume that it is possible to manufacture certificates for these

public keys in such a way that the parts of the certificates that are signed by a Certification Authority (CA) also collide, so that the signatures are in fact identical. For RSA we have shown how this goal can actually be achieved for X.509 certificates. Then Alice can ask the CA for certification of one of her public keys, and obtain a valid certificate. By replacing the public key with the other one, she can craft a second certificate that is equally valid as the first one. If so desired this can be done without any involvement of the CA, in which case she obtains two valid certificates for the price of only one. The resulting certificates differ in only a few bit positions in random looking data, and are therefore hard to distinguish by a cursory glance of the human eye. For standard certificate validating software both certificates will be acceptable, as the signature can be verified with the CA's public key.

A 'positive' application of the pairs of X.509 certificates would be that it enables Alice to distribute two RSA certificates, one for encryption and the other for signature purposes, for the transmission cost of just one certificate plus the few positions where the RSA moduli differ (similar ideas will be worked out in [8]). Indeed, the CA may knowingly participate in this application and verify that Alice knows both factorizations. However, if that is not done and the CA is tricked into signing one of the keys without being aware of the other one, the principle underlying Public Key Infrastructure that a CA guarantees the binding between an identity and a public key, has been violated. A CA usually requires its customers to provide proof of possession of the corresponding private key, to prevent key substitution attacks in which somebody tries to certify another person's public key in his own name. Although the way our certificates have been constructed makes it highly improbable that somebody could come up with either of them independent of Alice, it should be clear that the proof of possession principle has been violated. It would be more interesting to be able to produce two colliding certificates that have differences in the subject name, but at present this seems infeasible because it requires finding a second preimage.

Alice can also, maliciously, spread her two certificates in different user groups (different in space or time). When Bob sends Alice an encrypted message that has been encrypted by means of the wrong certificate, Alice may deny to be able to read it. When however the dispute is seriously investigated, it will be revealed that Alice has two colliding certificates. Alice may claim that she does not know how this is possible, but as finding second preimages still is prohibitively expensive, it is clear that either Alice is lying, or she has been misled by the key pair generating software.

Alice can produce digital signatures with one key pair, that are considered perfectly valid in one user group, and invalid in the other. This may be convenient for Alice, when she wants to convince one person of something, and to deny it to another person. Again, on serious investigation the colliding certificates will be revealed.

Another possible scenario is that Alice does not generate key pairs herself, but obtains her key pair(s) from a Key Generation Centre (KGC). This KGC may maliciously produce colliding public keys, of which one is sold to Alice, and

the other one kept for the KGC's own use, without Alice's consent. The KGC can distribute Alice's false certificate to Bob, and then Bob, when he thinks he is sending a message that only Alice can decrypt, ends up sending a message that only the KGC or a party collaborating with it can decrypt. Furthermore, when Alice sends a signed message to Bob, Bob will not accept her signature. So this constitutes a small denial of service attack. Note that a KGC in principle *always* has the possibility to eavesdrop on encrypted messages to Alice, and to spoof her signature. Our ability to construct colliding certificate does not add much value to this malicious application.

In all the above cases, when the colliding public keys are both secure keys, it cannot be detected from one key (or one certificate) that it has a twin sister. When e.g. one of the colliding public keys is intentionally weak, e.g. a prime as opposed to a composite modulus, this can be in principle detected by compositeness testing. Unless there is a concrete suspicion such tests are not carried out in practice, since they would make the public operation substantially more costly.

In conclusion it seems that possibilities for abuse seem not abundant, as the two public keys are very much related, and generated at the same time by the same person. Nevertheless, the principle of Public Key Infrastructure, being a certified binding between an identity and a public key, is violated by some of the scenarios we have described, based on random collisions for (a.o.) the hash function MD5, which is still popular and in use by certificate generating institutions. Particularly worrying is that any person, including the certificate owner, the Certification Authority, and any other party trusting a certificate, cannot tell from the information in one certificate whether or not there exists a second public key or certificate with the same hash or digital signature on it. In particular, the relying party (the one that does the public key operation with somebody else's public key) cannot be sure anymore of the Certification Authority's guarantee that the certificate owner indeed is in possession of the corresponding private key.

5 Conclusion

We demonstrated that on the basis of the existence of random hash collisions, in particular those for MD5 as shown by Wang et al. in [14], one can craft public keys and even valid certificates that violate one of the principles underlying Public Key Infrastructures. We feel that this is an important reason why hash functions that have been subject to collision attacks should no longer be allowed in certificate generation.

Acknowledgments. Acknowledgments are due to Hendrik W. Lenstra, Berry Schoenmakers, Xiaoyun Wang, Mike Wiener and an anonymous referee for helpful remarks and fruitful discussions.

References

1. D. Bleichenbacher, *Generating ElGamal signatures without knowing the secret key*, EuroCrypt '96, LNCS vol. 1070, Springer Verlag, pp. 10-18, 1996.
2. Hans Dobbertin, *Alf swindles Ann*, Cryptobytes 1(3) (1995), p. 5.
3. *Recent collision attacks on hash functions: ECRYPT position paper*, revision 1.1, February 2005, `http://www.ecrypt.eu.org/documents/STVL-ERICS-2-HASH_STMT-1.1.pdf`.
4. D. Kaminsky, *MD5 to be considered harmful someday*, preprint, december 2004, `http://www.doxpara.com/md5_someday.pdf`.
5. J. Kelsey, B. Laurie, Contributions to the mailing list "cryptography@metzdowd.com", December 22, 2004, available at `http://diswww.mit.edu/bloom-picayune/crypto/16587`.
6. A.K. Lenstra, *Generating RSA moduli with a predetermined portion*, Asiacrypt'98, Springer-Verlag LNCS 1514 (1998), 1–10.
7. A.K. Lenstra and B. de Weger, *On the possibility of constructing meaningful hash collisions for public keys, full version, with an appendix on colliding X.509 certificates*, April 2005, `http://www.win.tue.nl/~bdeweger/CollidingCertificates/ddl-full.pdf`.
8. A.K. Lenstra and B. de Weger, *Twin RSA*, submitted for publication, April 2005.
9. O. Mikle, *Practical Attacks on Digital Signatures Using MD5 Message Digest*, eprint archive 2004/356, `http://eprint.iacr.org/2004/356`.
10. NIST, *Digital Signature Standard*, NIST FIPS PUB 186, 1994.
11. J. Randall, M. Szydlo, *Collisions for SHA0, MD5, HAVAL, MD4, and RIPEMD, but SHA1 still secure*, RSA Laboratories technical notes, `http://www.rsasecurity.com/rsalabs/node.asp?id=2738`.
12. E. Rescorla, *What's the Worst That Could Happen?* presentation at the DIMACS Workshop on Cryptography: Theory Meets Practice October 14–15, 2004, `http://dimacs.rutgers.edu/Workshops/Practice/slides/rescorla.pdf`.
13. A. Shamir, *RSA for paranoids*, RSA Laboratories' Cryptobytes, 1(3) (1995), 1–4.
14. X. Wang, D. Feng, X. Lai, H. Yu, *Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD*, eprint archive 2004/199, `http://eprint.iacr.org/2004/199`, presented at the Crypto 2004 rump session, August 17, 2004.
15. X. Wang and H. Yu , *How to Break MD5 and Other Hash Functions*, EuroCrypt 2005, Springer LNCS 3494 (2005), 19-35.
16. X. Wang, X. Lai, D. Feng, H. Chen and X. Yu, *Cryptanalysis of the Hash Functions MD4 and RIPEMD*, EuroCrypt 2005, Springer LNCS 3494 (2005), 1-18.
17. X. Wang, H. Chen, X. Yu, *How to Find Another Kind of Collision for MD4 Efficiently*, Preprint, 2004.
18. X. Wang, D. Feng, X. Yu, *An Attack on Hash Function HAVAL-128* (Chinese), Science in China Ser. F (Information Sciences) 35(4) (2005), 405-416.
19. M. Wiener, personal communication, November 17, 2004.

Tunable Balancing of RSA

Steven D. Galbraith, Chris Heneghan, and James F. McKee*

Department of Mathematics,
Royal Holloway, University of London,
Egham, Surrey TW20 0EX, UK.
[Steven.Galbraith,C.Heneghan,James.McKee]@rhul.ac.uk

Abstract. We propose a key generation method for RSA moduli which allows the cost of the public operations (encryption/verifying) and the private operations (decryption/signing) to be balanced according to the application requirements. Our method is a generalisation of using small public exponents and small Chinese remainder (CRT) private exponents. Our results are most relevant in the case where the cost of private operations must be optimised. We give methods for which the cost of private operations is the same as the previous fastest methods, but where the public operations are significantly faster. The paper gives an analysis of the security of keys generated by our method, and a new birthday attack on low Hamming-weight private exponents.

1 Introduction

In many implementations of the RSA cryptosystem the public exponent e is chosen to be very small. Encryption and signature verification then use very few operations modulo N. On the other hand, decryption and signature generation, even if they are performed using the Chinese remainder theorem (CRT), cost much more than encryption/verifying.

This imbalance can be inconvenient in some situations. One example is when a device with limited computational power (such as a smart card) is required to generate RSA signatures. A related issue for such devices is the space requirement for storing private keys, which we would like to minimise. Another example is when a server is required to handle a large number of decryptions of messages from numerous clients and so its computational burden should be minimised. A related issue in this case is to ensure that such a server is not overly vulnerable to denial-of-service attacks. Hence there has been much interest in speeding up the private operations in RSA. Many of the previous solutions have achieved this at the cost of a significant loss of performance of the public operations. Our goal is to have fast private operations without paying such a high price for the public operations.

* All three authors were supported in their research by a grant from the MathFIT programme, jointly sponsored by the EPSRC and the LMS.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 280–292, 2005.

In some applications it might be desired that the public and private operations can be performed with the same computational effort. For example, this may be the case in protocols where two parties are required to act synchronously without idle time, or in systems where, to ensure fairness (i.e., that no party is at an advantage to others), the computational burden of all parties should be equal.

One early proposal to speed up the private operations was to choose the private exponent d to be small. This was cryptanalysed by Wiener [24] who showed that the scheme is insecure if $d < N^{0.25}$. These results were extended by Boneh and Durfee [1] to $d < N^{0.292}$. Wiener proposed two countermeasures to the above attacks. The first is to increase the size of the public exponent e, which causes a severe penalty on the public operations. The second variant is to use a private exponent d which is not itself small, but which reduces to small values when performing decryption/signing using the CRT. We call these "small CRT private exponents". There is a birthday attack (see [16]), which means that the CRT private exponents must have at least 160 bits. Apart from this attack, the security of such variants has been analysed by May [14] and the only serious attack known is in the unbalanced case (i.e., where the modulus is a product of primes of differing size).

One drawback of using small CRT private exponents is that the public exponent e is large and then the cost of encryption/verification is very expensive (much worse than original cost of decryption/signing). In this paper we propose a combination of 'small' public exponents e and 'small' CRT private exponents. Of course, they cannot both be very small. Our approach allows the designer to choose the parameters in a suitable way to get a trade-off between the costs of encryption/verification and decryption/signing.

A proposal to balance public and private exponents in RSA was given by Sun, Yang and Laih [20] (improved in [19], to resist an attack in [8]). These results do not utilise CRT private exponents and so their solution is not competitive with ours. Lim and Lee [13] gave methods for generating RSA private exponents with relatively low Hamming weight. Some of their methods do not use the Chinese remainder theorem, and so are slow. Section 5 of [13] does use CRT private exponents, but there seems to be no suggestion that the exponents may be taken to be short. Hence the cost of private operations in [13] is much slower than we achieve.

Other methods which allow faster private operations are multiprime RSA (products of more than two primes, see [4], [2] and [15]) and the Takagi system [21] (which uses moduli of the form $p^k q$). The use of 3-prime moduli with small CRT private exponents is the fastest method (for 1024-bit moduli) for RSA decryption/signing, but the public operations are slow since the public exponent then has 1024 bits. The Takagi system is particularly appealing as it requires small public exponents (otherwise the Hensel lifting is slow) and so both public and private operations are fast. Note that it is impossible to 'equalise' encryption and decryption times using Takagi.

Our results are most relevant in the case where the cost of private operations must be optimised. We give methods where the cost of private operations is the same as (or almost as good as) the previous fastest methods, but where the public operations are significantly faster. For example, the fastest known (1024 bit) RSA decryption is using small CRT decryption exponents and moduli which are a product of three primes. In this case we match the fastest known decryption time and also make the encryption time around four times faster. Some applications may require the cost of public and private operations to be equal. Our method gives a solution to this problem which is roughly twice as fast as previously known solutions. For a comparison with previous systems see the timings given in section 10.

We give a thorough analysis of the security of keys generated by our method, including several new attacks. Some of our attacks use ideas of Coppersmith [5,6]. Another of our attacks exploits linearisation. We also develop a new birthday attack on RSA private exponents of low Hamming weight.

Independently, a related but less general scheme has been proposed in [25,18].

2 The Key Generation Method

Let $N = p_1 p_2$. Then CRT private exponents are integers d_1, d_2 such that

$$ed_i \equiv 1 \pmod{(p_i - 1)} \quad \text{for} \quad i = 1, 2.$$

Such an equation may be written as

$$ed_i = 1 + k_i(p_i - 1) = k_i p_i - (k_i - 1). \tag{1}$$

Clearly $ed_i > p_i - 1$ and so e and d_i cannot both be very small. But we would like to have e and d_i so that $\log_2(e) + \log_2(d_i) \approx \log_2(p_i)$ by imposing the condition that k_i be small. The case $k_i = 1$ is uninteresting, but $k_i = 2$ is possible, as are large values of k_i.

The method we propose in this section allows us to choose the sizes of e and the d_i and then construct primes p_i so that equation (1) will be satisfied. The modulus is therefore generated by a special algorithm which depends on parameters which are likely to be known to an adversary. We will analyse the security of moduli arising from this key generation algorithm in later sections.

Choose parameters n_N, n_p, n_e, n_d, n_k which will be the bit-lengths of N, p_i, e, d_i and the k_i respectively ($i = 1, 2$). For example, one typical case would be

$$n_N = 1024, n_p = 512, n_e = 176, n_d = 338, n_k = 2$$

while another would be

$$n_N = 1024, n_p = 512, n_e = 508, n_d = 200, n_k = 196$$

(we must have $n_e + n_d - n_k = n_p$).

Key generation algorithm:

- Input: n_e, n_d, n_k.
- Choose an odd n_e-bit integer e (one may wish to choose e to have low Hamming weight).
- For $i = 1, 2$: choose random n_k-bit integers k_i coprime to e and odd n_d-bit integers d_i satisfying the congruence

$$d_i \equiv e^{-1} \pmod{k_i}. \tag{2}$$

  until $p_i = 1 + (ed_i - 1)/k_i$ is prime.
- Output: (p_1, p_2, d_1, d_2).

Note that the primes p_i are roughly $n_e + n_d - n_k$ bits long. The public key $(N = p_1 p_2, e)$ has an n_e-bit value for e as well as n_d-bit CRT private exponents. This clearly gives a tunable balance between the costs of encryption/verification and decryption/signing.

Notes:

- Clearly, the key generation algorithm runs in random polynomial time.
- Due to equation (2) we usually assume $n_d \geq n_k$. It is possible to develop more general key generation methods but they have much worse performance.
- In some settings we may also want to choose the d_i to have low Hamming weight. This is easily done if the k_i are small. Security in this case is discussed in section 6 below.
- An analogous algorithm can be used for moduli of the form $p^k q$, pqr etc.

3 Security

The key generation method produces special moduli, and so the security of the resulting public keys must be analysed. We are concerned with attacks which enable an adversary to obtain either of the private keys d_1 and d_2 of the system (equivalently, the factorisation of the modulus). We consider later the case of moduli which are a product of more two primes.

As is already well known, there is a birthday attack on the individual CRT private exponents (see [16] for some details). Hence, we always require that $n_d \geq 160$.

The security analysis falls naturally into two cases. The first is where the values k_1 and k_2 are known to the attacker (for example, they may take especially small values, such as $k_1 = k_2 = 2$). The security in this case is addressed in section 4. The second case is where the values k_1 and k_2 are private (in which case the entire security may depend on whether or not it is possible to calculate one or both of them). This is studied in section 5.

4 Known k_i

Suppose that the k_i are known (e.g., because they are small and can be guessed, or because they have been obtained from some computation).

The first attack is to note that equation (1) implies that

$$p_i \equiv k_i^{-1}(k_i - 1) \pmod{e}\,.$$

Hence we are in a strong position to apply results of Coppersmith on factoring with partial knowledge of a factor. The original results of Coppersmith [5] were phrased in terms of most or least significant bits of p_1 whereas we have information modulo e. The presentation in section 5 of [6] is more general than [5], and from results stated there it is easy to deduce the following result.

Theorem 1. *Let $N = pq$ where $p, q \approx N^{1/2}$. Suppose $e > N^{1/4}$. Then the factorisation of N can be obtained in polynomial time if $p_0 \equiv p \pmod{e}$ is known.*

This technique will split N for us if $n_e > n_N/4$. This attack can be extended to larger ranges by combining it with exhaustive search, Namely, suppose $p_i \equiv a \pmod{e}$ and so $p_i = a + ex$ for some unknown x. We can write $x = x_0 + 2^m y$ and search over all $0 \leq x_0 < 2^m$, trying the attack for each guess (actually, since e and p_i are odd we only need try those $x_0 \equiv a + 1 \pmod 2$). This extends the range of the attack to cases where $n_e + m \geq n_N/4$, but multiplies the complexity by 2^m. For security we impose the condition

$$n_e \leq n_N/4 - m\,, \tag{3}$$

where m is a security parameter. We shall take $m = 80$ for 1024-bit moduli. For the sample parameters above in the case $k_i = 2$ we have $n_e = 176 = 256 - 80 = n_N/4 - m$ and so the parameters resist this attack.

There are several alternative attacks: see [10] for details. Briefly, the key ideas are as follows.

1. d_1 is a small solution to the equation $ex + (k_1 - 1) \equiv 0 \pmod{p_1}$.
2. $d_1 d_2 \approx k_1 k_2 N/e^2$ if n_e is sufficiently large.
3. If $k_1 = k_2$ then $p_1 + p_2$ is known modulo e^2, and one can use this to speed up the computation of the discrete logarithm of $a^{p_1+p_2} \equiv a^{N+1} \pmod{N}$.

Condition (3) guards against all of these attacks.

5 Unknown k_i

We now turn to the case where the k_i are private. We will give methods to find the k_i, from which it is usually immediate to recover the full private key using the methods of the previous section. Indeed for this section we imagine that the parameters are such that knowledge of the k_i would mean that the private key can be obtained using the methods of the previous section.

One can perform an exhaustive search on, say, k_1 and (if the parameters are suitable) apply the methods of section 4 to check each guess. Hence, we impose the condition $n_k \geq m$ for security.

5.1 Linearisation Attack

An obvious attack is to use information available by taking equation (1) modulo e. We have (for $i = 1, 2$)

$$k_i p_i \equiv (k_i - 1) \pmod{e}.$$

Multiplying these two equations together we see that $(x, y) = (k_1, k_2)$ is a solution to the multivariate equation

$$xy(N-1) + x + y - 1 \equiv 0 \pmod{e}. \tag{4}$$

For this subsection we suppose that $2n_k \leq n_e$ which implies that (k_1, k_2) is a relatively small solution to equation (4). We linearise by defining $u = xy$ and $v = x + y - 1$ and define $0 < A < e$ by $A = (1 - N) \pmod{e}$. We have

$$uA \equiv v \pmod{e} \tag{5}$$

where u and v have bounded size. Candidate (u, v) pairs can be found in polynomial time using the continued fraction method as long as $uv < e$ (the details are standard, and are given below in a more general setting).

Hence, to resist the attack we are required to impose the restriction $n_k \geq n_e/3$.

We now give details of how the algorithm can be extended to the case when $3n_k$ is only slightly larger than n_e by adding an exhaustive-search. A similar problem was investigated by Wiener [24] and Verheul and van Tilborg [23], who both give extensions to the continued fraction method.

First we recall a famous result on good rational approximations.

Theorem 2. *(see e.g., Theorem 184 of [9]). Suppose that* $\gcd(a, b) = \gcd(c, d) = 1$ *and that*

$$\left|\frac{a}{b} - \frac{c}{d}\right| \leq \frac{1}{2d^2}.$$

Then c/d is one of the convergents of the continued fraction expansion of a/b.

The congruence $Au \equiv v \pmod{e}$ means that there is some integer l such that

$$Au = v + le\,. \tag{6}$$

In our situation, we know that v has about $n_k + 1$ bits, and u has about $2n_k$ bits, but we may as well handle the most general situation. We suppose that there are parameters r and s bounding the sizes of u and v:

$$0 < v \leq r \leq e\,, \qquad 0 < u \leq s \leq e\,.$$

(The analysis where u and/or v is allowed to be negative is similar.)

We also suppose that $0 < A < e$, with $\gcd(A, e) = 1$. We then have $l \geq 0$. Dividing equation (6) by ue and adding $j/2s^2$ to each side gives

$$\frac{A}{e} + \frac{j}{2s^2} = \frac{v}{eu} + \frac{j}{2s^2} + \frac{l}{u}\,.$$

Now we apply theorem 2 to see that l/u is a continued fraction convergent to $A/e + j/2s^2$ if

$$\left| \frac{v}{eu} + \frac{j}{2s^2} \right| \leq \frac{1}{2u^2} ,$$

and this will certainly hold if j is the nearest integer to $-2vs^2/eu$, since $1/2s^2 \leq 1/2u^2$.

If u lies in the range $s/2 < u \leq s$, then we have $-4rs/e \leq j \leq 0$. To find smaller values of u, we repeat with the ranges

$$s/4 < u \leq s/2, \ s/8 < u \leq s/4, \ \ldots$$

(i.e., substitute $s/2^t$ for s everywhere).

We have established the following theorem.

Theorem 3. *Consider the congruence* $Au \equiv v \pmod{e}$*, where* $0 < A < e$ *and* $\gcd(A, e) = 1$*. All solutions* (u, v) *satisfying*

$$0 < v \leq r \leq e , \qquad 0 < u \leq s \leq e$$

have that u *is a multiple of a denominator of a continued fraction convergent for one of*

$$A/e + 4^t j/2s^2 , \qquad t \leq \log_2 s , \qquad j \leq 0 , \qquad |j| \leq 4rs/e2^t .$$

In particular, all solutions with $\gcd(u, v) = 1$ *can be found in time* $O(\lceil rs/e \rceil (\log e)^2)$*. Any solutions with* $\gcd(u, v) > 1$ *can be obtained from those with* $\gcd(u, v) = 1$ *by scaling.*

Naively, the expected number of solutions to $Au \equiv v \pmod{e}$ with $0 < v \leq r$ and $0 < u \leq s$ is rs/e (but this does depend on A and e: for extreme cases, consider $A = 1$, or let e be the product of all primes up to r). Hence Theorem 3 is essentially optimal. A qualitatively similar result was obtained by Dujella [7] by other means.

In our case, $r \approx 2^{n_k+1}$ and $s \approx 2^{2n_k}$. We can restrict to $t = 0$ in the above. Therefore, we obtain the following result.

Theorem 4. *Let* $N = p_1 p_2$ *be produced by the key generation method with parameters* (n_e, n_k)*. If* $n_e + m \geq 3n_k + 2$ *then with the computation of continued fraction approximations to* $O(2^m)$ *rational numbers near* A/e*, we will find the pair* $(u, v) = (k_1 k_2, k_1 + k_2 - 1)$*, up to scaling.*

Since it is improbable that $k_1 k_2$ and $k_1 + k_2 - 1$ will have a large common factor, scaling is not a significant issue. Hence, to keep the k_i secret, we impose the condition

$$3n_k \geq n_e + m . \tag{7}$$

For our suggested parameters in the case of k_i private we have $3n_k = n_e + m$, and so the attack is avoided.

Note that the algorithm yields $u = k_1 k_2$ which is already enough for attack variant 2 in Section 4. If the actual values k_1 and k_2 are required then they can be recovered as solutions to the quadratic $t^2 - (v+1)t + u = 0$. For parameters of interest, the value of e will be sufficiently large that the methods of Section 4 immediately recover the private key.

5.2 Further Attacks

As in section 4 we have presented the most successful attack on our scheme. We briefly discuss a number of other approaches to the problem, some of which are reformulations or special cases of the attack described in section 5.1. Full details will appear in [10]. In all cases, the parameter restrictions are either already implied by condition (7) or require n_e to be larger than is of interest. For any lattice-based attack, one needs to allow for a brute-force extension: one could try 2^m lattices if any one of them can be checked quickly.

1. One can perform a birthday attack on the variable v in equation (5).
2. One can attempt to find a small solution to the multivariate equation (4) using lattice-based methods, following the methods of Coppersmith [5,6] and Boneh and Durfee [1].
3. Another lattice attack was suggested to us by an anonymous referee. Multiplying together the key equations $ed_i - 1 = k_i(p_i - 1)$ we observe that the polynomial $f(x, y, z) = 1 - ex - (N+1)y + yz$ has the "small" solution $(x, y, z) = (d_1 + d_2, k_1 k_2, p_1 + p_2)$ modulo e^2. One can then attempt a trivariate lattice attack. This attack certainly does not apply if $n_e \leq 0.5 n_N$, and any larger value of e is not pertinent in our setting.
4. Another lattice-based attack arises from reducing equation (1) modulo p_1.
5. Another attack is to note that equation (1) implies that the polynomial $f(x, y) = xy - 1$ has the solution $(x, y) = (k_1, p_1 - 1)$ modulo e.

6 Low Hamming Weight Private CRT Exponents

Suppose that the CRT private exponents d_i are chosen to have low Hamming weight w. We have $ed_1 \equiv 1 \pmod{p_1 - 1}$ and so $g^{ed_1} \equiv g \pmod{p_1}$ for any g such that $\gcd(g, p_1) = 1$. In other words, our goal is to solve the discrete logarithm problem of g to base g^e modulo p_1 where the discrete logarithm is known to have low Hamming weight.

We mimic a randomised algorithm of Coppersmith (see Algorithm 3 of Stinson [17]) for discrete logarithms modulo a known prime p. If the bit-length of the exponent is m and the Hamming weight is w then this algorithm has complexity $\tilde{O}(\sqrt{w}\binom{m/2}{w/2})$.

Since we do not know p_1 (only the multiple N of p_1) we use the standard FFT technique as employed by Qiao and Lam [16] (also see Turk [22]). Hence we obtain an attack with complexity $\tilde{O}(\sqrt{w}\binom{n_d/2}{w/2})$. To guard against this attack, we require $\sqrt{w}\binom{n_d/2}{w/2} \geq 2^m$.

7 Summary of Parameter Restrictions in the Two Prime Case

We summarise the restrictions on parameters imposed by the above attacks. For a fixed modulus length n_N which is a product of two primes of the size $n_p = n_N/2$ we must specify parameters n_e, n_k and n_d. Let m be a security parameter (e.g., $m = 80$) so that we want security against an adversary whose total computational power is at most 2^m operations. The parameters must satisfy:

$$n_e + n_d - n_k = n_p \tag{8}$$

$$n_d \geq 2m \tag{9}$$

$$n_d \geq n_k. \tag{10}$$

The final condition is due to the key generation technique.

Now we must separate the two cases in our security analysis.

If the k_i are small (or are not supposed to contribute to security) then the restrictions on parameters also include

$$n_e \leq n_N/4 - m. \tag{11}$$

If the k_i are meant to stay private and add to the security of the system then we must have

$$n_k \geq (n_e + m)/3. \tag{12}$$

Also, if n_e is large, then the third attack in subsection 5.2 introduces an extra parameter restriction, but this not a concern if $n_e \leq 0.5n_N$.

The sample parameters given above satisfy all of these requirements.

The case of small k_i is good for the application of equalising public and private operations using RSA. The case of large k_i seems to be more suitable for the application of minimising the cost of private operations using RSA.

It is possible to give general families of parameters for large N. For example, in the case where the k_i are private, choosing $n_k = n_d$ leads to the general family of secure parameters

$$n_e = n_p \approx n_N/2,\ n_k = n_d = (n_p + m)/3 \approx n_p/3 \approx n_N/6.$$

For these parameters we have private operations about 3 times faster than standard RSA (using CRT) and public operations twice as fast as if using large public exponents (which would previously have been the case with such fast decryption).

Similarly, in the case when $n_k = 2$ we obtain the general family

$$n_e = n_p/2 - m \approx n_N/4, \quad n_d = n_p/2 + m \approx n_N/4.$$

This gives private operations about twice as fast as standard RSA, and public operations about 4 times faster than was previously possible in this setting.

8 Equalising Cost of Encryption and Decryption

As mentioned in the introduction, one application of our method is arranging that RSA encryption and decryption have the same running time. This can be achieved in standard 1024-bit RSA using a roughly 300-bit encryption exponent and using the CRT for decryption.

For our tunable scheme, the following parameters roughly equalise encryption and decryption times as long as the private exponents are chosen to have low Hamming weight (say, weight 38, as one can check that $\sqrt{38}\binom{338/2}{38/2} \geq 2^{80}$)

$$n_N = 1024, n_p = 512, n_e = 176, n_d = 338, n_k = 2.$$

An exact timing comparison depends on the platform. Our experiments give roughly 4.3ms for encryption and 5.2ms for decryption. This is nearly twice as fast as the balanced solution obtained using standard RSA.

For 2048-bit moduli we propose the parameters $(n_p, n_e, n_d, n_k) = (1024, 373, 653, 2)$. The timings for encryption and decryption in this case are around 32ms, which is significantly faster than the 58ms required for the classical solution (i.e., $n_e = 575$ and using the CRT for decryption).

9 Multiprime and Takagi

The Takagi system [21] requires e to be extremely small, and so our approach cannot be applied. We therefore turn our attention to the multiprime case. The key generation method is easily generalised to the case of products of three (or more) primes. We now consider the security of our keys in the multiprime case. See [3,11] for related analyses of private exponent attacks on multiprime RSA.

9.1 Known k_i

As in section 4, if k_1 is known then we obtain $p_1 \equiv k_1^{-1}(k_1 - 1) \pmod{e}$ and we are in the situation of trying to factor N when given partial information about one of the factors.

The results of Coppersmith [6] can be applied in this setting. The analogous result to Theorem 1 above is that if $N = pq$ where $p = N^{\beta}$ (where q is not necessarily prime) and if we know p modulo $e \geq N^{\beta(1-\beta)}$, then the value of p can be computed in polynomial time. For example, if $p \approx N^{1/3}$ then we would need p modulo $e \approx p^{2/3} \approx N^{2/9}$ to recover p. For security we impose $n_e \leq 2n_N/9 - m$.

Other variants of this attack can be considered (for example, considering information from several primes at once). Further analysis will appear in [10].

9.2 Unknown k_i

We generalise the linearisation attack of subsection 5.1 Suppose that $N = p_1 p_2 \dots p_r$. The equations $k_i p_i \equiv (k_i - 1) \pmod{e}$ multiply to give

$$\prod_{i=1}^{r} k_i p_i \equiv \prod_{i=1}^{r} (k_i - 1) \pmod{e}.$$

Performing the linearisation $u = \prod_{i=1}^{r} k_i$ and $v = u - \prod_{i=1}^{r}(k_i - 1)$ as before gives the equation

$$u(1 - N) \equiv v \pmod{e} \tag{13}$$

where $u \approx 2^{rn_k}$ and $v \approx r2^{(r-1)n_k}$. The continued fraction method finds (u, v) in polynomial time if $e > uv = r2^{(2r-1)n_k}$.

Once the values u and v have been computed then one can find the values k_i by factoring u and combining the factors in various combinations. One can then attempt to recover the private key as in section 4.

Hence, to resist this attack (and the extension by Theorem 3) requires $n_k \geq (n_e + m)/(2r - 1)$ (where, say, $m = 80$ for 1024-bit moduli). In practice, this condition is much more easily satisfied than the analogous condition in the two prime case.

Indeed, we propose the following parameters

$$n_N = 1024, n_p = 341, n_e = 261, n_d = 160, n_k = 80.$$

This choice of parameters recovers the fastest known decryption time for RSA (i.e., 160-bit CRT private exponents in the 3-prime case) but with public operation nearly 4 times faster than previously realisable using this method.

For the 2048-bit case we set $m = 128$ and suggest the parameters $(n_N, n_p, n_e, n_d, n_k) = (2048, 683, 582, 256, 156)$. Once again, we match the fastest speed for the private operations (256-bit 3-prime CRT exponents) while the public exponent is reduced from 2048 bits to 582 bits.

One can check that our sample parameters satisfy $n_k = 80 \geq (n_e + m)/5 = (261 + 80)/5 \approx 68$.

10 Performance Comparison

We have made a trial implementation of these variants of RSA. The timings for 1024-bit moduli are below (in milliseconds on a 1.80GHz Pentium 4 desktop using the GNP library). These timings are very approximate and should only be treated as a relative guideline. Note that all entries are using the Chinese remainder theorem for decryption. In the tunable case the triple indicates the values (n_e, n_d, n_k).

Variant	Encryption (ms)	Decryption (ms)
RSA-CRT ($e = 2^{16} + 1$)	0.5	8.2
3-prime ($e = 2^{16} + 1$)	0.5	4.4
Takagi ($e = 2^{16} + 1$)	0.5	3.2
RSA small CRT private exponents	41	2.8
3-prime small CRT private exponents	41	2.3
Tunable (176,338,2), wt(d)=38	4.4	5.2
Tunable (508,200,196)	12	3.4
3-prime tunable (261,160,80)	6.3	2.3

11 Note Added in Proof

We very recently became aware of another lattice attack on our system in the case of unknown k_i. This attack is similar to the trivariate attack in subsection 5.2. One observes that $(x, y, z) = (d_1(k_2 - 1) + d_2(k_1 - 1), k_1k_2, k_1 + k_2 + 1)$ is a solution to $ex + (1 - N)y - z \equiv 0 \pmod{e^2}$. To guard against this attack, one can take $n_d + 4n_k \geq 2n_e + 4m$.

Note that the parameters suggested above do not meet this requirement, but the benefit of a tunable key generation system is that we can easily fix the parameters so that they do. For example, in the two prime case we can take

$$n_e = n_p \approx n_N/2, \quad n_d = n_k = (2n_p + 4m)/5 \approx 2n_p/5 \approx n_N/5.$$

12 Acknowledgements

The authors thank J.-S. Coron and A. May for comments (the latter visited Royal Holloway with support from the ECRYPT Network of Excellence in Cryptology). In particular, J.-S. Coron suggested the linearisation attack in Section 5 and A. May pointed out a number of lattice-based attacks. The authors also acknowledge the useful comments provided by several anonymous referees.

References

1. D. Boneh and G. Durfee, Cryptanalysis of RSA with private key d less than $N^{0.292}$, in J. Stern (ed.), Eurocrypt '99, Springer LNCS 1592 (1999) 1–11.
2. D. Boneh and H, Shacham, Fast variants of RSA, *CryptoBytes*, **5**, No. 1 (2002) 1–9.
3. M. Ciet, F. Koeune, F. Laguillaumie and J.-J. Quisquater, Short private exponent attacks on fast variants of RSA, Louvain technical report CG-2003/4 (2003).
4. T. Collins, D. Hopkins, S. Langford and M. Sabin, Public key cryptographic apparatus and method. US Patent (1997).
5. D. Coppersmith, Small solutions to polynomial equations and low exponent RSA vulnerabilities, *J. Crypt.*, **10** (1997) 233–260.
6. D. Coppersmith, Finding small solutions to small degree polynomials, in J. H. Silverman (ed.), CaLC 2001, Springer LNCS 2146 (2001) 20–31.
7. A. Dujella, Continued fractions and RSA with small secret exponent, *Tatra Mt. Math. Publ.*, **29** (2004) 101-112.
8. G. Durfee and P. Nguyen, Cryptanalysis of the RSA scheme with short secret exponent from Asiacrypt '99, in T. Okamoto (ed.) Asiacrypt 2000, Springer LNCS 1976 (2000) 14–29.
9. G. H. Hardy and E. M. Wright, *An introduction to the theory of numbers*, 5th ed., Oxford (1979).
10. C. Heneghan, Ph.D. thesis, in preparation.
11. M. J. Hinek, M. K. Low and E. Teske, On some attacks on multi-prime RSA, in K. Nyberg and H. M. Heys (eds.), SAC 2002, Springer LNCS 2595 (2003) 385–404.
12. N. A. Howgrave-Graham, Finding small solutions of univariate modular equations revisited, in M. Darnell (ed.), Cryptography and Coding, Springer LNCS 1355 (1997) 131–142.

13. C. H. Lim and P. J. Lee, Sparse RSA secret keys and their generation, Proc. of 3rd Annual Workshop on Selected Areas in Cryptography (SAC'96), (1996) 117–131.
14. A. May, Cryptanalysis of unbalanced RSA with small CRT-exponent, in M. Yung (ed.) CRYPTO 2002, Springer LNCS 2442 (2002) 242–256.
15. C. A. M. Paixão, An efficient variant of the RSA cryptosystem, preprint (2003).
16. G. Qiao and K.-Y. Lam, RSA signature algorithm for microcontroller implementation, J.-J. Quisquater and B. Schneier (eds.), CARDIS '98, Springer LNCS 1820 (2000) 353–356.
17. D. Stinson, Some baby-step-giant-step algorithms for the low Hamming weight discrete logarithm problem, *Math. Comp.* **71**, No. 237 (2001) 379–391.
18. H.-M. Sun and M.-E Wu, An Approach Towards Rebalanced RSA-CRT with Short Public Exponent, Cryptology ePrint Archive, 2005/053.
19. H.-M. Sun and C.-T. Yang, RSA with balanced short exponents and its application to entity authentication, in S. Vaudenay (ed.), PKC 2005, Springer LNCS 3386 (2005) 199–215.
20. H.-M. Sun, W.-C. Yang and C.-S. Laih, On the design of RSA with short secret exponent, in K. Y. Lam et al (eds.), ASIACRYPT '99, Springer LNCS 1716 (2000) 150–164.
21. T. Takagi, Fast RSA-type cryptosystem modulo $p^k q$, in H. Krawczyk (ed.), CRYPTO '98, Springer LNCS 1462 (1998) 318–326.
22. J. W. M. Turk, Fast arithmetic operations on numbers and polynomials, in H. W. Lenstra Jr. and R. Tijdeman (eds.), Computational methods in number theory, Part 1, Mathematical Centre Tracts 154, Amsterdam (1984).
23. E. R. Verheul and H. C. A. van Tilborg, Cryptanalysis of 'less short' RSA secret exponents, Applicable Algebra in Engineering, Communication and Computing, Vol. 8 (1997) 425–435.
24. M. Wiener, Cryptanalysis of short RSA secret exponents, IEEE Trans. Inf. Th., 36 (1990) 553–558.
25. M.-E. Wu, A Study of RSA with Small CRT-Exponent, Thesis of Master Degree, Department of Applied Mathematics, National Chiao Tung University, Taiwan, June 2004.

Key Management for Role Hierarchy in Distributed Systems

Celia Li, Cungang Yang, and Richard Cheung

Department of Electrical and Computer Engineering
Ryerson University
Toronto, Ontario, M5B 2K3
cli@ee.ryerson.ca
cungang@ee.ryerson.ca
rcheung@ee.ryerson.ca

Abstract. As distributed computing system grow in size, complexity and variety of application, the problem of protecting sensitive data from unauthorized disclosure and tampering becomes increasingly important. In this paper, we present a cryptographic key management solution to the role-based access control (RBAC) model in distributed systems. The key management method used for distributed system is decentralized. Each local domain is managed by its local domain security manager and any key modifications of roles in a local domain will not affect the keys of roles in other local domains.

1 Introduction

In distributed systems, there are millions of privileges and thousands of users located in different domains and each privilege is assigned to thousands of users; thus, the conventional access control lists (ACL) are enormous in size and their maintenance are difficult and costly. To offer an acceptable solution, Role-Based Access Control (RBAC) as a key security technology was proposed [3,6,11]. With the framework of role-based access control, access decisions are based on the roles that individual users have as part of an organization. Users are granted membership into roles (such as doctor, nurse, teller, manager) according to their competencies and responsibilities. Under RBAC, roles can have overlapping responsibilities and privileges; that is, users belonging to different roles may need to perform common privileges. For instance, some general privileges may be performed by all employees. In this situation, it would be inefficient and administratively cumbersome to specify repeatedly these general privileges for each role that gets created. Role hierarchies can be established to provide for the natural structure of an enterprise. A role hierarchy defines roles that have unique attributes and that may contain other roles; that is, one role may implicitly include the privileges that are associated with another role. We define that role S is a *direct child roles* of role R and R is a *direct parent role* of S if R directly inherits all privileges of S. Furthermore, we consider that R is an indirect parent role of T, and T is an *indirect child role* of R if R is a direct parent role of S and S is a direct parent role of T.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 293-302, 2005.

A distributed system can be divided into multiple local domains and each local domain has a role hierarchy that is administrated by its local domain security manager. Figure 1 shows an example of role hierarchies in local domain 1 and local domain 2. In local domain 1, role A is a direct parent role of role B and role C, role D is a direct parent role of role C and role H, role B is a direct parent role of role E and role F, and role C is a direct parent role of role F and role G. Similarly, in local domain 2, role I is a direct parent role of role J and role K, role J is a direct parent role of role L and role M, and role K is a direct parent role of role M and role N. Moreover, a role in a local

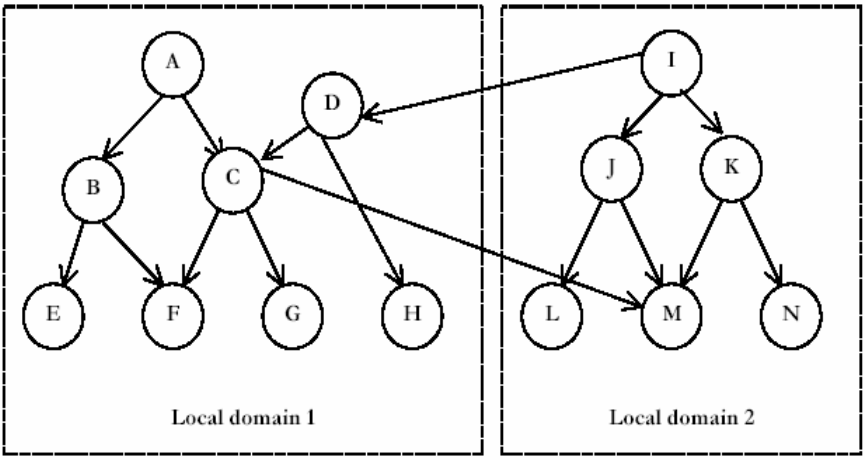


Figure 1: An Example of Role Hierarchies in Local Domain 1 and 2

domain may take tasks from other local domains and therefore may have privileges and work as roles from other local domains. For instance, we assume that role C in local domain 1 needs to have privileges of role M in local domain 2, in this case, we call role C an *extended parent role* of role M and role M an *extended child role* of role C. Similarly, Role I in local domain 2 is an extended parent role of role D in local domain 1 and role D is an extended child role of role I.

For the key management problem in distributed systems, we should consider the following two cases: (1) key management scheme for role hierarchy in a local domain and (2) key management scheme for multiple local domains. In a role hierarchy of a local domain, each role has a key assigned by its local domain security manager. A role inherits the privileges of its direct or indirect child roles. Therefore, the key of a role should have the keys of its direct or indirect child roles. The easiest and most efficient way to achieve this is that the user holds all those keys of its direct or indirect child roles. However, since the users have to hold a great deal of keys, it is memory consuming and difficult to manage keys and to keep system secure. The key management scheme for multiple local domains is related to the inter-relationships of the roles in different local domains. Each local domain has its own role hierarchy that is administrated by its local domain security manager and a role in a local domain may take roles from other local domains. It is not a good key management method if we

apply a single key derivation algorithm on multiple local domains' role hierarchies because any key value modified in a local domain could affect the keys of roles in other local domains. For instance, if the key value of role C in local domain 1 is changed, the keys of role F and role G in local domain 1 should be changed. Also, the key value of role M in local domain 2 will be affected. If we have multiple local domains that inter-connected with each other, the management work will be terrible for local domain security managers.

In the paper, we present an efficient cryptosystem solution to the access control problem of the RBAC model in distributed systems. The contributions of the solution are that (1) The key management method for multiple local domains is decentralised: keys in each local domain are generated by its local domain security manager. Any key modifications of a role in a local domain will not affect the keys of roles in other local domains. (2) It uses limited number of hash functions and deals with multiple hierarchy structures. (3) It has low storage requirement. Each key of a role is calculated through one-way hash functions. In this way, less extra public parameters are needed for the key derivation.

2 Related Work

Some works are related with the key management scheme for a hierarchical structure. Akl and Taylor [1,2] proposed an efficient solution to the key distribution problem in hierarchical structure using a cryptographic approach. A center authority is responsible for key generation and key distribution and each security class Sci is assigned a secret key ki and a public parameter ti. The information items held by Sci are enciphered by an available cryptosystem with the secret key Ki. For Sci <=SCj, Sci can use his secret key Ki and SCj's public parameter ti to derive SCj's secret key Kj, and then reads the information items held by SCj. The advantage of Akl and Taylor's scheme is that the key generation and key derivation algorithms are very simple; however, it has some drawbacks. (1) It uses one hash function and only deal with one hierarchy structure. (2) The public integer ti must be chosen carefully, otherwise, there exist the possibility of some users collaborating to compute a key to which they are not entitled. (3) It requires a large amount of storage for public parameters when the number of security classes in the hierarchy is large. (4) It is hard to be modified. To avoid this problem, Mackinnon et al. [9] proposed a near optimal algorithm to assign ti to each node Ui. However, the size of these integers ti is proportional to the number n of the nodes in the network. When n becomes very large this method becomes impractical. Therefore, the sizeable ti problem is still left unsolved. Moreover, it is impossible to add a new node into the system whenever all the security keys have been issued.

There are no previous work consider the key management issues for role hierarchies in distributed systems. Some recent related research work includes key management for encrypted data storage in distributed systems that dealing with the secure communication between storage devices (Server) and the user (Client). M. Blaze [3] proposed a method for encrypted storage in a local domain. However no specific files sharing mechanism by different local domains is provided. Thus, users must indi-

vidually provide the decryption keys if they want to share some of their files. Therefore CFS is not suited to be used in distributed environments with complex access control policies.

K. Fu[9] presents a method that provides encrypted storage and communication as well as distributed files sharing mechanisms. File sharing within a local domain is managed by symmetric keys that a restored encrypted with the same local domain members public keys on a key server. This may cause coordination problems in a distributed environment where a local domain contains members from different administrative domains. If such a local domain is managed by multiple authorities from different domains and membership changes dynamically, keeping the key server up to date could quickly become an administration nightmare.

The aspect of key management has not attracted much attention in neither the SESAME [4,7] nor the Kerberos [8,10] documents. Technical solutions are in place that distribute keys as part of the protocol, but not very much is said about how the keys should be administered properly. In Kerberos the issue of key control is quite simple. The KDS is always in control, except when negotiating sub session keys. Even the session keys used for communication between a client and a server are generated by the KDS. In SESAME the session keys are generated at the initiator side. Even the public key of an entity is generated by the entity itself. In Kerberos the only place in the system where key generation is done is at the KDC. In SESAME every client must be able to generate good keys.

The rest of the paper is organized as follows. Section 3 presents the key management method for distributed systems. It is mainly comprised of two parts: (1) The key management method for a role hierarchy in local domain and (2) The key management protocol for the role hierarchies in multiple local domains. Finally, section 4 concludes the paper.

3 Key Management Method for Distributed System

Based on the key generation rules described in Appendix A, we propose a novel technique that facilitates managing role hierarchy in a local domain.

3.1 Key Management Method for Role Hierarchy in a Local Domain

Our research employs XML technology for the definition and representation of an XML-based role hierarchy security policy. The XML specification [5] is the work of the World Wide Web Consortium (W3C) Standard Generalized Markup Language (SGML) Working Group. The advantages of using XML are that, as a meta-language, XML can effectively define role hierarchy and is able to be extended and modified easily. XML represents desired role hierarchy precisely and effectively and offers an additional degree of flexibility.

The role hierarchy security policy defines basic elements of the RBAC model such as roles. Also, it defines the relationships between different elements of the model.

(i.e. a role with its direct child roles, a role with its public parameters). The syntax of the security policy is described in Appendix B.

Based on the XML-based role hierarchy security policy, we introduce a key generation algorithm. The algorithm first parses the security policy and automatically creates a role graph. (An example role graph is shown in Figure 2), then the algorithm will try to find the least-expense path between each dead-end role and the role R in order to reduce the calculation time to generate a key value of a role R. According to the key generation rules, the time used to calculate a key value of a role depends on the number of hash functions and the time used by each hash functions. Assume that all hash functions cost the same time, the time used to calculate the key value of R will be related with the number of hash functions used in each path (from a dead-end role to role R). Also, according to the key generation rule, the number of public parameters in a path determines the number of hash functions that will be used. Thus, the number of public parameters will be mainly considered. For example, in the following diagram in Figure 2, to calculate the key value of a role K, from each dead-end role, we find three different paths: (1) D-H-K (2) D-C-G-K and (3) A-C-G-K. The number of public parameters in path (1) is 1, the number of public parameters in path (2) is 2, and the number of public parameters in path (3) is 2. Thus path (1) should be the least-expense path. If there are more than one path having the same least expense value, then the algorithm will pick up any of them and calculate the key value of the role.

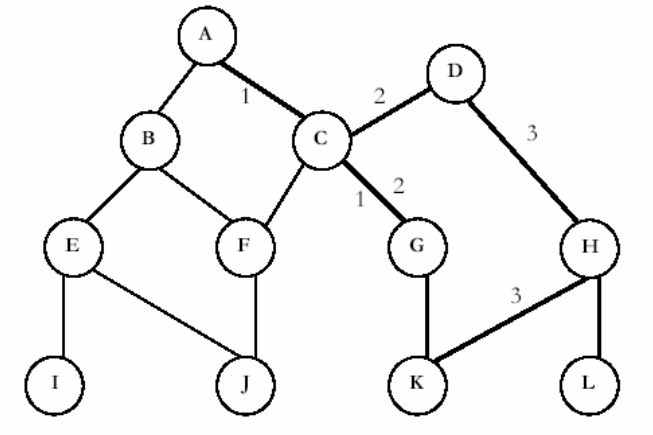


Figure 2: Key Generation for role K in the Role Graph

In summary, to get the key value of a role R, the algorithm will search for role R in the role graph and try to find all paths from dead-end roles to role R. The algorithm will then calculate the number of public parameters in each path and determines the least-expense path. Finally, the key value of role R will be calculated according to the key generation rules.

The key generation algorithm is shown as follows:

```
Key Generation Algorithm (Role R)
/* For a given role R, calculate its key value.
{
For each dead-end role in the role graph{
  Search role graph and calculate the least-expense
paths from role R to the dead-end role
  Calculate the key value of the role according to the
key generation rules.
  }
  Output the key value of role R.
}
```

After we designed the key generation algorithm, the general procedure of the key generation could be implemented in the following 3 steps and shown in Figure 3.

Step 1 The XML-based role hierarchy security policy is parsed and an object graph is created.

Step 2 Key generation algorithm generates the key of the role in the role hierarchy based on the object graph and the key generation rules

Step 3 The key value of the role is saved in a key database.

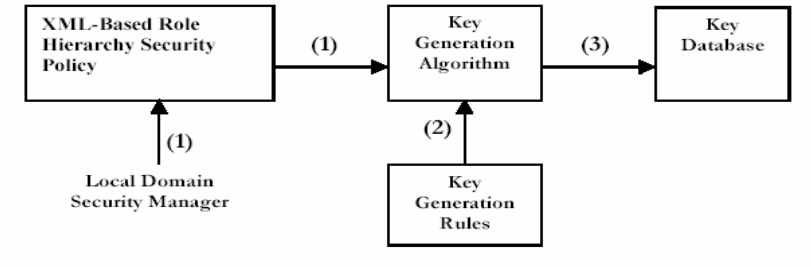


Figure 3: The General Procedural for Key Generation in a Local Domain

3.2 Key Management for Role Hierarchies in Multiple Local Domains

To deal with the key management for role hierarchies in multiple local domains, we propose a security management architecture. Under this architecture, any key value modification in a local domain will not affect the key values of other local domains.

The security management architecture is comprised of multiple local domains and each local domain has a local domain security manager, a number of servers, clients and a key database. An example of the architecture of a local enterprise domain is shown in Figure 4, where we assume that we only have two local domains in a distributed system: local domain 1 and local domain 2.

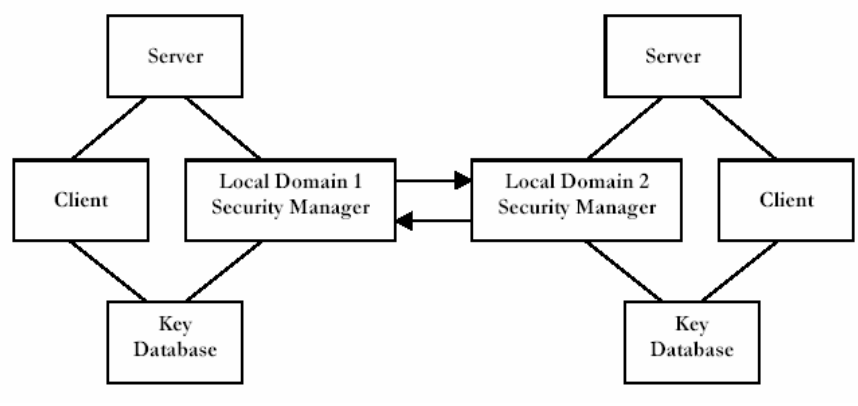


Figure 4: An Example of the PRBAC Security Management Architecture

The main functions of each element of the architecture are described as follows:

- The local domain security manager is in charge of authorizing roles to its local domain users and other local domain users.
- The client accepts the application of roles from its local domain users and returns the authorized roles back to the users.
- Key database stores the created key values for all the roles in the local enterprise domain.

Based on the management architecture described, an example key management procedure for multiple local domains is shown as follows. (See Figure 5)

- Local domain security manager generates and assigns a key to each role of its local domain.
- Negotiation between local domain security managers and assign a role such as role I in local domain 2 as a extended parent role of role D in local domain 1. Also assign role C in local domain 1 as a extended parent role of role M in local domain 2.
- When a user who is a member of a role in local domain 2, for instance role I, would like to apply for a role of local domain 1, for instance role H, he/she must apply it to the security manager of its local domain 2.
- The security manager of the local domain 2 authenticates the user and its membership of the role, then apply the user's key requirement to the security manager of local domain 1.
- Security manager of local domain 1 checks whether role H is a child role of role I, and verify if there exists a path between role H and role I.
- After the path between role H and role I is verified (see Figure 5.3), the security manager of local domain 1 send the key of role H to the security manager of local domain 2.
- The security manager of local domain 2 forwards the key of role H to user of role I in local domain 2.

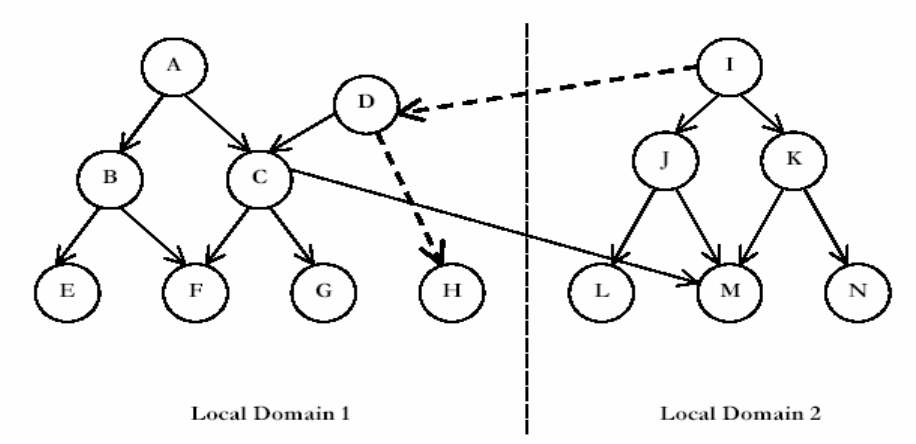


Figure 5: Inter-relationship between role hierarchies in local domain 1 and local domain 2

4 Conclusion

We present a cryptographic key management solution to solve the access control problem for distributed systems. The proposed key management method is suitable for role-based access control systems. We believe that it will also be suitable for other access control system such as access control lists (ACL) and mandatory access control (MAC). After the negotiation between the local domain security managers of a role-based and an ACL or MAC system, the proposed key management method also could be applied to other access control systems.

References

1. S.G. Akl and P.D. Taylor. Cryptographic Solution to a Multilevel Security Problem. In D. Chaum, R.L.Rivest, and A.T. Sherman, editor, *Advanced in Cryptology*.
2. S.G. Akland P.D. Taylor. Cryptographic Solution to a Problem of Access Control in a Hierarchy. *ACM Transaction on Computer Sysdtems*.1 (3) : 239-248, 1983.
3. Ezedin Barka and Ravi Sandhu, A Role-Based Delegation Model and Some Extensions, Proc. of 23rd National Information Systems Security Conference (NISSC 2000). December, 2000.
4. M. Blaze. A cryptographic file system for UNIX. In *ACM Conference on Computer and Communications Security*, pages 9–16, 1993.
5. Extensible Markup Language (XML) http://www.w3.org/XML/
6. D.Ferraiolo, R.Sandhu, E.Gavrila, D.R.Kuhn, and R.Chandramouli, Proposed NIST Standard for Role-Based Access Control, ACM Transactions on Information and System Security, Vol4, No3 (2001) 224-274.
7. K. Fu. Group sharing and random access in cryptographic storage file systems. Master's thesis, Massachusetts Institute of Technology, 1999.

8. Kaijser. A review of the SESAME development. In *ACISP: Information Security and Privacy: Australasian Conference*, volume 1438 of *Lecture Notes in Computer Science*. Springer-Verlag, 1998.
9. Mackinnon, S.T., Taylor, P.D.: An optimal algorithm for assigning cryptographic keys to control access in a hierarchy, *IEEE Transaction on Computer systems.,* 1985, C-34, (9), pp. 797-802.
10. Tom Parker and Denis Pinkas. *SESAME V4 – Overview*. SESAME systems documentation, https://www.cosic.esat.kuleuven.ac.be/sesame/.
11. Ravi Sandhu, Edward Coyne, Hal Feinstein and Charles Youman, Role-Based Access Control Models, IEEE Computer, Volume 29, Number 2, February, 1996.

Appendix A: Key Generation Rules

The key generation rules are based on the well-known one-way hash functions, which are easy to compute but computationally difficult to invert. For a given role hierarchy in a local domain, we choose a set of one-way hash functions H_i: $\{H_1, H_2 \ldots\ldots H_n\}$, where n is the maximum number of direct child roles that a role can directly access in the role hierarchy. These hash functions are public known.

The keys of roles in the role hierarchy are generated by local domain security manager and shown as follows:

In a role hierarchy, we call a role that has no direct parent roles a *dead-end role.* For each dead-end role, local domain security manager assigns it an arbitrary key.
If a role r_j only has one direct parent role whose key is K; and if r_j is the *i*th direct child role of its direct parent role (form left to right), then the key of role r_j will be H_i (K).

If role r_j has more than one direct parent roles (r_j^1, r_j^2 ..... r_j^m) (except the extended parent roles from other local domains), and suppose r_j is the *i*th direct child role of its leftmost direct parent role r_j^1. Also, if r_j is the *k*th direct child role of role r_j^2, r_i is the *n*th direct child role of role r_j^m, and the keys of the direct parent roles of r_i (r_j^1, r_j^2 ... r_j^m) are K_l, K_2 ……K_m, then the key of the role r_j will be H_i (H_i (K_1), H_k (K_2) ……H_n (K_m)), 1<= i, k, n <=m.

Appendix B: Specification of XML-based Role Hierarchy Security Policy

Our research employs XML for syntactic representation of the security policies. Each RBAC component is represented by an XML element shown as follows:

Role is represented by:

<ROLE ID= *role-id*> </ROLE>

The above syntax defines a new XML tag of type ROLE with a required ID attribute value *role-id*.

Role hierarchy is represented as a set of INHERITES elements, each of which associates a role with its direct child role. A role-role relationship is represented by:

<! --Role hierarchy definition-- >

<INHERITES FROM = *ri* To *rj*></INHERITES>

The above syntax defines a new XML tag of type INHERITES with a required FROM (role *ri*) and TO (role *rj*) attribute values which indicate role ri is a direct parent role of role rj.

A key assignment assigns key to a role and it is represented by:

<! – Key Generation definition-- >

<KEY-ROLE ROLE= *role k* FROM = *role1……role m*>

< CHILD-OF-LEFTMOST VALUE =*value* / CHILD-OF-LEFTMOST>

< CHILD-OF-PARENT ROLE= *role 1* VALUE= *value 1* /CHILD-OF-PARENT >

……

< CHILD-OF-PARENT ROLE=*role m* , VALUE=*value m* /CHILD-OF-PARENT >

</KEY-ROLE >

The above syntax defines a new XML tag of type Key-Role with a required ROLE attribute of value *role k* and a required attribute FROM attribute of value *role 1……role m*, role k is the direct child role of *role 1……role m*. The parameter values for the *role k* are defined by a CHILD-OF-LEFTMOST tag with a required VALUE attribute of value *value* and a CHILD-OF-PARENT tag with a required ROLE attribute of value *role 1……role m* and VALUE attribute of value *value1 …… value m.*

A Formalization of Distributed Authorization with Delegation

Shujing Wang and Yan Zhang

University of Western Sydney, Australia
{shwang, yan}@cit.uws.edu.au

Abstract. Trust management is a promising approach for the authorization in distributed environment. There are two key issues for a trust management system: how to design high-level policy language and how to solve the compliance-checking problem [3,4]. We adopt this approach to deal with distributed authorization with delegation. In this paper, we propose an *authorization language* $\mathcal{AL}$, a human-understandable high level language to specify various authorization policies. We define the semantics of $\mathcal{AL}$ through Answer Set Programming. Language $\mathcal{AL}$ has rich expressive power which can not only specify delegation, threshold structures addressed in previous approaches, but also represent structured resources and privileges, positive and negative authorizations, separation of duty, incomplete information reasoning and partial authorization and delegation. We also demonstrate the application of language $\mathcal{AL}$ through an authorization scenario.

Keywords: Access control, trust management, authorization, delegation, answer set programming, knowledge representation, nonmonotonic reasoning.

1 Introduction

Access control is an important topic in computer security research. It provides availability, integrity and confidentiality services for information systems. The access control process includes identification, authentification and authorization. With the development of Internet, there are increasing applications that require distributed authorization decisions. For example, in the application of electronic commerce, many organizations use the Internet (or large Intranets) to connect offices, branches, databases, and customers around the world. One essential problem among those distributed applications is how to make authorization decisions, which is significantly different from that in centralized systems or even in distributed systems which are closed or relatively small. In these traditional scenarios, the authorizer owns or controls the resources, and each entity in the system has a unique identity. Based on the identity and access control policies, the authorizer is able to make his/her authorization decision. In distributed authorization scenarios, however, there are more entities in the system, which can be both authorizers and requesters, and probably are unknown to each other. Quite often, there is no central authority that everyone trusts. Because the authorizer does not know the requester directly, he/she has to use the information from the third parties who know the requester better. He/She trusts these third parties only for certain things to certain

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 303–315, 2005.

degrees. The trust and delegation issues make distributed authorization different from traditional access control scenarios.

In recent years, the trust management approach, which was initially proposed by Blaze *et al.* in [3], has received a great attention by many researchers [3,4,5,8,9]. Under this approach public keys are viewed as entities to be authorized and the authorization can be delegated to third parties by credentials or certificates. This approach frames the authorization decision as follows: " Does the set C of $credentials$ prove that the $request\ r\ complies\ with$ the local security $policy\ P$? ", from which we can see that there are at least two key issues for a trust management system: (1) Designing a high-level policy language to specify the security policy, credentials, and request. Better it is if the language has richer expressive power and is more human-understandable; (2) Finding well theory foundation for checking proof of compliance.

In our research, we view the problem of a language for representing authorization policy and credentials as a knowledge representation problem. Logic programming approach has been proved very successful in knowledge representation. Some research using logic programming in centralized access control systems has been well developed [2,6], where underlying languages can support multiple access-control policies and achieve separation of policies from enforcement mechanisms. But their work focuses on centralized systems, and can not be used in distributed systems. Delegation Logic [9], developed by Li *et al.*, is an approach in distributed systems along this line. However the D1LP is based on Definite ordinary logic program, which is less expressive and flexible, and cannot deal with some important issues such as negative authorization, and nonmonotonic reasoning. D2LP extends D1LP to have the nonmonotonic features and bases its syntax and semantics on GCLP (Generalized Courteous Logic Programs). Since it was only briefly mentioned in [7], it was not clear yet how D2LP can handle nonmonotonic reasoning in distributed authorization. In our research, we design a language $\mathcal{AL}$, a nonmontonic language, which is based on Answer Set Programming. We adopt the delegation with depth control and static and dynamic threshold structure from DL approach. Compared to previous work in trust management systems, our language is able to specify positive and negative authorization, the request from conjunctive subjects, structured resources and privileges, incomplete information reasoning, and partial delegation and authorization. The reasons we choose Answer Set Programming as the foundation of language $\mathcal{AL}$ are as follows: (1) Through negation as failure, Answer Set Programming implements nonmonotonic reasoning which is reasoning about incomplete information. A language with nonmontonic feature is easy to specify security policies which is close to the natural language. For example, many systems permit a login request only if they do not find that the requester inputs the password wrong over consecutive three times; (2) The highly efficient solvers for Answer Set Programming have been implemented, such as `Smodels, dlv` etc. This is an important reason that Answer Set Programming has been widely applied in product configuration, planning, constraint programming, cryptanalysis, and so on. We need to indicate that `Smodels` supports some extended literals such as constraint literal and conditional literal which are particularly useful to express the static and dynamic threshold structures.

The rest of this paper is organized as follows. Section 2 presents the syntax and expressive features of language $\mathcal{AL}$. Section 3 develops an answer set language $\mathcal{L}_{Ans}$,

provides the translation from $\mathcal{AL}$ into $\mathcal{L}_{Ans}$, and defines the semantics of $\mathcal{AL}$ based on the translation. Section 4 provides a scenario to demonstrate our research. Finally, Section 5 concludes the paper.

2 An Authorization Language $\mathcal{AL}$

In this section, we define the syntax of the authorization language $\mathcal{AL}$ and illustrate its expressiveness via some examples.

The authorization language $\mathcal{AL}$ consists of *entities, atoms, thresholds, statements, rules* and *queries*. The formal BNF syntax of $\mathcal{AL}$ is given in Figure 1. We explain the syntax in detail as follows.

Entities
In distributed systems, the entities include *subjects* who are authorizers who own or control resources and requesters who make requests, *objects* which are resources and services provided by authorizers, and *privileges* which are actions executed on objects. We define constant entities starting with a lower-case character and variable entities starting with an upper-case character for subjects, objects and privileges. We provide a special subject, $local$. It is the local authorizer which makes the authorization decision based on local policy and credentials from trusted subjects.

Atoms
An atom is a function symbol with n arguments, generally one, two or three constant or variable entities to express a logical relationship between them. There are three types of atoms: (1) $\langle relation\text{-}atom \rangle$. An atom in this type is a 2-ary function symbol and expresses the relationship of two entities. For example, $below(ftp, pub\text{-}services)$ denotes that ftp is one of $pub\text{-}services$. (2) $\langle assert\text{-}atom \rangle$. This type of atoms, denoted by $exp(a_1, \ldots, a_n)$, is application dependant function symbol with n arguments, usually one, two or three constant or variable entities and states the property of the subjects, the relationship between entities. It is a kind of flexible atoms in language $\mathcal{AL}$. For example, $isaTutor(alice)$ denotes that $alice$ is a tutor. (3) $\langle auth\text{-}atom \rangle$. The $auth\text{-}atom$ is of the form, "$right(\langle sign \rangle, \langle priv \rangle, \langle obj \rangle)$". It states the $positive$ or $negative$ privilege executed on the $object$ based on its arguments, $\langle sign \rangle$, $\langle obj \rangle$, and $\langle priv \rangle$. When an auth atom is used in delegation statement, the $\langle sign \rangle$ is $\square$ to denote both positive and negative authorizations.

Statements
There are four types of statements, *relation statement, assert statement, delegation statement*, and *auth statement*. Only the local authorizer can issue the *relation statement* to denote the structured resources and privileges.

Threshold
There are two types of threshold structures, static threshold and dynamic threshold. The static threshold structure is of the form,"$sthd(k, [s_1, s_2, \ldots, s_n])$", where k is the threshold value, $[s_1, s_2, \ldots, s_n]$ is the static threshold pool, and we require $k \leq n$ and $s_i \neq s_j \; for \; 1 \leq i \neq j \leq n$. This structure states that we choose k subjects from the threshold pool.

$$
\begin{aligned}
\langle obj\rangle &::= \langle obj\text{-}con\rangle \mid \langle obj\text{-}var\rangle\\
\langle priv\rangle &::= \langle priv\text{-}con\rangle \mid \langle priv\text{-}var\rangle\\
\langle sub\rangle &::= \langle sub\text{-}con\rangle \mid \langle sub\text{-}var\rangle\\
\langle sub\text{-}set\rangle &::= \langle sub\text{-}con\rangle \mid \langle sub\text{-}con\rangle,\ \langle sub\text{-}set\rangle\\
\langle sub\text{-}struct\rangle &::= \langle sub\rangle \mid \text{“[”}\langle sub\text{-}set\rangle\text{“]”} \mid \langle threshold\rangle\\
\langle sub\text{-}ext\text{-}set\rangle &::= \langle dth\rangle \mid \langle dth\rangle,\ \langle sub\text{-}ext\text{-}set\rangle\\
\langle sub\text{-}ext\text{-}struct\rangle &::= \langle sub\rangle \mid \text{“[”}\langle sub\text{-}set\rangle\text{“]”} \mid \langle threshold\rangle \mid \text{“[”}\langle sub\text{-}ext\text{-}set\rangle\text{“]”}\\
\langle entity\rangle &::= \langle sub\rangle \mid \langle obj\rangle \mid \langle priv\rangle\\
\langle entity\text{-}set\rangle &::= \langle entity\rangle \mid \langle entity\rangle,\ \langle entity\text{-}set\rangle\\
\langle sign\rangle &::= + \mid - \mid \square\\
\langle relation\text{-}atom\rangle &::= below(\langle obj\rangle, \langle obj\rangle) \mid below(\langle priv\rangle, \langle priv\rangle \mid\\
&\quad neq(\langle entity\rangle, \langle entity\rangle) \mid eq(\langle entity\rangle, \langle entity\rangle)\\
\langle assert\text{-}atom\rangle &::= exp(\langle entity\text{-}set\rangle)\\
\langle auth\text{-}atom\rangle &::= right(\langle sign\rangle, \langle priv\rangle, \langle obj\rangle)\\
\langle k\rangle &::= \langle natural\text{-}number\rangle\\
\langle threshold\rangle &::= \langle sth\rangle \mid \langle dth\rangle\\
\langle sth\rangle &::= sthd(\langle k\rangle, \text{“[”}\langle sub\text{-}set\rangle\text{“]”})\\
\langle dth\rangle &::= dthd(\langle k\rangle, \langle sub\text{-}var\rangle, \langle assert\text{-}stmt\rangle)\\
\langle relation\text{-}stmt\rangle &::= \text{“}local\text{” } says\ \langle relation\text{-}atom\rangle\\
\langle assert\text{-}stmt\rangle &::= \langle sub\rangle\ asserts\ \langle assert\text{-}atom\rangle\\
\langle delegate\text{-}stmt\text{-}body\rangle &::= \langle sub\text{-}struct\rangle\ delegates\ \langle auth\text{-}atom\rangle\ with\ depth\ \langle k\rangle\ to\ \langle sub\rangle\\
\langle delegate\text{-}stmt\text{-}head\rangle &::= \langle sub\rangle\ delegates\ \langle auth\text{-}atom\rangle\ with\ depth\ \langle k\rangle\ to\ \langle sub\text{-}struct\rangle\\
\langle auth\text{-}stmt\text{-}body\rangle &::= \langle sub\text{-}struct\rangle\ grants\ \langle auth\text{-}atom\rangle\ to\ \langle sub\rangle\\
\langle auth\text{-}stmt\text{-}head\rangle &::= \langle sub\rangle\ grants\ \langle auth\text{-}atom\rangle\ to\ \langle sub\text{-}ext\text{-}struct\rangle\\
\langle head\text{-}stmt\rangle &::= \langle relation\text{-}stmt\rangle \mid \langle assert\text{-}stmt\rangle \mid\\
&\quad \langle auth\text{-}stmt\text{-}head\rangle \mid \langle delegate\text{-}stmt\text{-}head\rangle\\
\langle body\text{-}stmt\rangle &::= \langle relation\text{-}stmt\rangle \mid \langle assert\text{-}stmt\rangle \mid\\
&\quad \langle auth\text{-}stmt\text{-}body\rangle \mid \langle delegate\text{-}stmt\text{-}body\rangle\\
\langle list\text{-}of\text{-}body\text{-}stmt\rangle &::= \langle body\text{-}stmt\rangle \mid \langle body\text{-}stmt\rangle, \langle list\text{-}of\text{-}body\text{-}stmt\rangle\\
\langle rule\rangle &::= \langle head\text{-}stmt\rangle\ [\ if\ [\ \langle list\text{-}of\text{-}body\text{-}stmt\rangle\ ]\\
&\quad [\ with\ absence\ \langle list\text{-}of\text{-}body\text{-}stmt\rangle\ ]\ ]\\
\langle query\rangle &::= \langle sub\rangle\ requests\ (+, \langle priv\rangle, \langle obj\rangle) \mid\\
&\quad \text{“[”}\langle sub\text{-}set\rangle\text{“]”}\ requests\ (+, \langle priv\rangle, \langle obj\rangle)
\end{aligned}
$$

Figure 1. BNF for the Authorization Language $-\mathcal{AL}$

The dynamic threshold structure is of the form,“$dthd\ (k, S, \langle sub\rangle\ assert\ exp(\ldots, S, \ldots))$”, where S is a subject variable and we require that S is one argument of assert atom exp. This structure denotes we choose k subjects who satisfy the assert statement.

Rules

The rule is of the form,

$\langle head\text{-}stmt \rangle \ if \ \langle list\text{-}of\text{-}body\text{-}stmt \rangle \ with \ absence \ \langle list\text{-}of\text{-}body\text{-}stmt \rangle.$

The issuer of the rule is the issuer of the head statement. Then we limit the issuer in head statements is just a single subject while it can be a complex structure in body statements. We present the following examples to demonstrate expressive power of $\mathcal{AL}$.

Partial delegation and authorization: A firewall system protects the *allServices*, including *ssh, ftp*, and *http*. The administrator permits *ipA* to access all the services except *ssh* and delegates this right to *ipB* and allow it redelegated within 2 steps.

$local$ grants $right(+, \ access, \ X)$ to ipA if
$local$ says $below(X, \ allServices)$, $local$ says $neq(X, \ ssh)$.
$local$ delegates $right(\square, \ access, \ X)$ with depth 2 to ipB if
$local$ says $below(X, \ allServices)$, $local$ says $neq(X, \ ssh)$.

Separation of duty: A company chooses to have multiparty control for emergency key recovery. If a key needs to be recovered, three persons are required to present their individual PINs. They are from different departments, $managerA$, a member of management, $auditorB$, an individual from auditing department, and $techC$, one individual from IT department.

$local$ grants $right(+, \ recovery, \ k)$ to $[managerA, \ auditorB, \ techC]$.

Negative authorization: In a firewall system, the administrator *sa* does not permit *ipB* to access the *ftp* services.

sa grants $right(-, \ access, \ ftp)$ to ipB.

Incomplete information reasoning: In a firewall system, the administrator *sa* permit a *person* to access the *mysql* service if the human resource manager *hrM* asserts the person is a *staff* and not in holiday.

sa grants $right(+, \ access, \ mysql)$ to X if
$hrM \ asserts \ isStaff(X)$, with absence $hrM \ asserts \ inHoliday(X)$.

Query

Language $\mathcal{AL}$ supports single subject query and group subject query. They are of the forms,

sub requests $right(+, p, o)$, and $[s_1, s_2, \ldots, s_n]$ requests $right(+, p, o)$.

Through group subject query, we implement *separation of duty* which is an important security concept. It ensures that a critical task cannot be carried out by one subject. If we grant an authorization to a group subject, we permit it only when the subjects in the group request the authorization at the same time.

3 Semantics of $\mathcal{AL}$

In this section, we define the semantics for language $\mathcal{AL}$ through translating it to Answer Set Programming based language $\mathcal{L}_{Ans}$. We first present the definition for the *domain description* $\mathcal{D}_{\mathcal{AL}}$ and how to answer queries $\mathcal{Q}_{\mathcal{AL}}$ of language $\mathcal{AL}$. Queries are the requests in $\mathcal{AL}$. In subsection 3.1, we introduce the language $\mathcal{L}_{Ans}$ briefly. In the following subsection, we define function $TransRules(\mathcal{D}_{\mathcal{AL}})$ to translate $\mathcal{D}_{\mathcal{AL}}$ into

program $\mathcal{P}$ of $\mathcal{L}_{Ans}$, and function $TransQuery(\mathcal{Q}_{\mathcal{AL}})$ to translate query $\mathcal{Q}_{\mathcal{AL}}$ into program Π and ground literals $\varphi(+)$ and $\varphi(-)$. We use $\varphi(+)$ to denote positive right and $\varphi(-)$ to denote negative right. There is detailed description for them in section 3.2. We solve a query based on $\mathcal{P}$, Π and φ via *Smodels*.

An answer set program may have one, more than one, or no answer sets at all. For a given program Π and a ground atom φ, we say Π *entails* φ, denoted by $\Pi \models \varphi$, iff φ is in every answer set of Π.

Definition 1. *A* domain description $\mathcal{D}_{\mathcal{AL}}$ *of language* $\mathcal{AL}$ *is a finite set of rules.*

Definition 2. *Given a domain description* $\mathcal{D}_{\mathcal{AL}}$ *and a query* $\mathcal{Q}_{\mathcal{AL}}$ *of language* $\mathcal{AL}$*, there are* $TransRules(\mathcal{D}_{\mathcal{AL}}) = \mathcal{P}$ *and* $TransQuery(\mathcal{Q}_{\mathcal{AL}}) = \Pi \cup \varphi(+) \cup \varphi(-)$*. We say that query* $\mathcal{Q}_{\mathcal{AL}}$ *is* permitted, denied, *or* unknown *by the domain description* $\mathcal{D}_{\mathcal{AL}}$ *iff* $(\mathcal{P} \cup \Pi) \models \varphi(+)$*,* $(\mathcal{P} \cup \Pi) \models \varphi(-)$*, or* $(\mathcal{P} \cup \Pi) \not\models \varphi(+)$ *and* $(\mathcal{P} \cup \Pi) \not\models \varphi(-)$ *respectively.*

3.1 An Overview of Language $\mathcal{L}_{Ans}$

In this subsection, we first briefly introduce language $\mathcal{L}_{Ans}$, and then give the propagation rules, authorization rules, and conflict resolution and decision rules in $\mathcal{L}_{Ans}$.

Language $\mathcal{L}_{Ans}$ is based on Answer Set Programming [1] and we use *Smodels* as the solver of $\mathcal{L}_{Ans}$ which has some extended features such as constraint and conditional literals to express the threshold structures [10]. The alphabet of language $\mathcal{L}_{Ans}$ includes *entity sorts, function symbols* and *predicates symbols*. For an access control system, the authorization policy is the key component. We need to indicate that it is easy and flexible for $\mathcal{L}_{Ans}$ to specify different types of policies. We will only present some parts of rules for authorization policies to demonstrate the expressiveness of $\mathcal{L}_{Ans}$ because of a space limitation. Readers are referred to our full paper for the complete description of $\mathcal{L}_{Ans}$ [11].

Propagation rules: In most real world situations, the work to assign all the authorizations is burdensome and not necessary. The security officer prefers to assign them partly and propagate them based on propagation policy. The following rules are our policy.

$$auth(S_1,\ S_2,\ right(Sign, P, Obj_2), T) \leftarrow$$
$$auth(S_1, S_2, right(Sign, P, Obj_1), T),\ below(Obj_2, Obj_1).$$
$$below(A_1, A_3) \leftarrow\ below(A_1, A_2),\ below(A_2, A_3).$$

The first rule is for object propagation. We have a same rule for privilege propagation. The second rule is for structured data propagation.

Authorization rules: In this subsection, we present the authorization rules for the following authorization policy: if there is only positive authorization and no negative authorization, we conclude positive authorization; if there is no positive authorization, we grant negative authorization; if there are positive and negative authorizations at the same time, We leave the decision problem to conflict resolution and decision policy.

$$grant(X, right(+, P, O)) \leftarrow$$
$$auth(local, X, right(+, P, O), T),\ not\ exist\_neg(X, right(-, P, O)),$$
$$not\ exist\_subneg(X, right(-, P, O)).$$

$$ggrant(l, right(+, P, O)) \leftarrow auth(local, X, right(+, P, O), T),$$
$$match(X, right(+, P, O)),\ not\ exist\_neg(X, right(-, P, O)),$$
$$not\ exist\_subneg(X, right(-, P, O)).$$

In second rule, l is a special group subject entity to represent the set of subjects who make a request together. We have corresponding rules for negative authorizations [11].

Conflict resolution and decision rules When both positive and negative authorizations are permitted, the conflict occur. There are various different conflict solving policies in existing approaches [2,6]. In this paper, we consider *delegation* as an action and get the step for each authorization which is decided by the delegation step. All the authorizations arise from $local$ originally and then the step number denotes how far the authorization is away from $local$. We take the smallest step authorization preference. If the conflict occurs with the same step, we deny the request. Our approach belongs to the third category. The following are some of rules for our conflict resolution and decision policy.

$$grant(X, right(+, P, O)) \leftarrow\ auth(local, X, right(+, P, O), T1),$$
$$auth(local, X, right(-, P, O), T2),\ neg\_far(X, right(-, P, O), T2),$$
$$not\ pos\_far(X, right(+, P, O), T1),$$
$$not\ exist\_subneg(X, right(-, P, O)).$$
$$ggrant(l, right(+, P, O)) \leftarrow\ auth(local, X, right(-, P, O), T2),$$
$$neg\_far(X, right(-, P, O), T2),\ match(X, right(+, P, O)),$$
$$auth(local, X, right(+, P, O), T1),\ not\ pos\_far(X, right(+, P, O), T1),$$
$$not\ exist\_subneg(X, right(-, P, O)).$$

The both rules specify the policy we take positive authorization if positive and negative authorizations coexist and positive authorization has smaller step than negative authorization.

3.2 Transformation from $\mathcal{AL}$ to $\mathcal{L}_{Ans}$

In $\mathcal{D}_{\mathcal{AL}}$, the basic unit of a rule is a statement. Let *h* be a head statement and *b* a body statement, a rule is as follows,

$$h_0,\ if\ b_1,\ b_2, \ldots, b_m,\ with\ absence\ b_{m+1}, \ldots, b_n.$$

where h_0 is *head statement* denoted by $head(r_{\mathcal{D}})$ and b_is are *body statements* denoted by $body(r_{\mathcal{D}})$. We call the set of statements, $\{b_1,\ b_2, \ldots, b_m\}$, *positive statements*, denoted by $pos(r_{\mathcal{D}})$ and the set of statements, $\{b_{m+1},\ b_{m+2}, \ldots, b_n\}$, *negative statements*, denoted by $neg(r_{\mathcal{D}})$. If $m = 0$ and $n = 0$, the rule is h_0 called a *fact*.

In language $\mathcal{AL}$, there are function symbols, $assert\text{-}atom$ and $auth\text{-}atom$. Correspondingly there are functions $exp\ (a_1, \ldots, a_n)$ and $right\ (sign, priv, obj)$ in language $\mathcal{L}_{Ans}$. In our translation, if there is no confusion in the context, we use exp and $right$ to denote them in both languages.

Body translation: In language $\mathcal{AL}$, there are four types of body statements, *relation statement, assert statement, delegation statement*, and *auth statement*. As delegation statement and auth statement have similar structure, we give their transformation together. For each rule $r_{\mathcal{D}}$, its body statement b_i is one of the following cases.

(1) Relation statement:

$local$ says $below(arg_1,\ arg_2)$,

$local$ says $neq(arg_1,\ arg_2)$, and $local$ says $eq(arg_1,\ arg_2)$.

Replace them respectively in program $\mathcal{P}$ using:

$$below(arg_1,\ arg_2), \tag{1}$$

where arg_1 and arg_2 are of *object or privilege entity sort*,

$neq(arg_1,\ arg_2)$ and $eq(arg_1,\ arg_2)$,

where arg_1 and arg_2 are of same type entity sort to specify they are equal or not equal. In $Smodels$, neq and eq are internal functions and work as a constraint for the variables in the rules.

(2) Assert body statement:

$issuer$ asserts exp.

Replace it in program $\mathcal{P}$ using,

$$assert(issuer, exp), \tag{2}$$

where $issuer$ is a subject constant or variable, and exp is an assertion.

(3) Auth body statement or delegation statement:

$issuer$ grants $right$ to $grantee$, or

$issuer$ delegates $right$ with depth k to $delegatee$.

If $issuer$ is a subject constant or variable, we replace the statements in program $\mathcal{P}$ using,

$$auth(issuer, grantee, right, T),\ or \tag{3}$$

$$delegate(issuer, delegatee, right, k, Step), \tag{4}$$

where T is a step variable that means how many steps the right has gone through from $issuer$ to $grantee$, k delegation depth, and $Step$ length variable that the delegation has gone through.

For auth statements, if $issuer$ is a set of subjects, $[s_1, \ldots, s_n]$, we replace them in program $\mathcal{P}$ by conjunction forms of (3) as,

$auth(s_1, grantee, right, T_1), \ldots,\ auth(s_n, grantee, right, T_n)$.

If $issuer$ is a static threshold structure, $sthd(k, [s_1, s_2, \ldots, s_n])$, we use choice rule to replace them as follows,

$k\{auth(s_1, grantee, right, T_1), \ldots,\ auth(s_n, grantee, right, T_n)\}k$.

If $issuer$ is a dynamic threshold structure, $dthd(k, S, assert(sub, exp(S))$, we use choice rule including constraint literal to replace them using,

$k\{auth(S, grantee, right, T_i) :\ assert(sub, exp(S))\}k$.

The translation for delegation body statements is to replace (3) by (4) in previous forms.

We translate the positive statements as above steps, and for the negative body statements, we do the same translation and just add not before them.

Head translation: If the head statement h_0 is a *relation statement* or an *assert statement*, the translation is same as the body statements. We adopt the rules (1), (2) to translate them respectively. In relation head statements, there are no statements for atom neq and eq that just be used as a variable constraints in body statements. Here we present the translation for $assert\ head\ statement$, and $delegation\ head\ statement$.

1. Auth head statement:
   $issuer$ grants $right$ to $grantee$.
   If $grantee$ is a subject constant or variable, we replace it by,
   $auth(issuer, grantee, right, 1)$,
   where 1 means the $right$ is granted from $issuer$ to $grantee$ directly.
   If $grantee$ is a complex structure, *subject set, threshold*, or *subject extent set*, we introduce group subject entity l_{new} to denote the subjects in complex subject structures, and replace its head in program $\mathcal{P}$ as follows,
   $auth(issuer, l_{new}, right, 1)$.
   We add different rules for different structures.
   case 1: $[s_1, \ldots, s_n]$
   $match(l_{new}, right) \leftarrow auth(issuer, l_{new}, right, 1),$
   $n\{req(s_1, right), \ldots, req(s_n, right)\}n.$
   case 2: $sthd(k, [s_1, s_2, \ldots, s_n])$
   $match(l_{new}, right) \leftarrow auth(issuer, l_{new}, right, 1),$
   $k\{\ req(s_1, right), \ldots,\ req(s_n, right)\}k.$
   case 3: $dthd\ (k, S, sub\ assert\ exp(S)\ )$
   $match(l_{new}, right) \leftarrow auth(issuer, l_{new}, right, 1),$
   $k\{\ req(S, right) :\ assert(sub, exp(S))\ \}k.$
   case 4: $[dthd\ (k_1, S, s_1\ assert\ exp_1(S)\ ), \ldots,\ dthd\ (k_n, S, s_n\ assert\ exp_n(S)\ )]$
   $holds(l_{new}, exp_1(S)) \leftarrow auth(issuer, l_{new}, right, 1),$
   $assert(s_1, exp_1(S)),\ req(S, right).$
   $\ldots$
   $holds(l_{new}, exp_n(S)) \leftarrow auth(issuer, l_{new}, right, 1),$
   $assert(s_n, exp_n(S)),\ req(S, right).$
   $match(l_{new}, right) \leftarrow k_1\{holds(l_{new}, exp_1(S))\}k_1, \ldots,$
   $k_n\{holds(l_{new}, exp_n(S))\}k_n.$
2. Delegation head statement:
   $issuer$ delegates $right$ with depth k to $delegatee$.
   If $delegatee$ is a subject constant or variable, we replace the statement in program P using, "$delegate(issuer, delegatee, right, k, 1)$", where k is the delegation depth, and 1 means the $issuer$ delegates the $right$ to $delegatee$ directly
   Moreover, we need to add the following implied rules for it in program $\mathcal{P}$.
   Auth-delegation rules: When the issuer delegates a right to the delegatee, the issuer will agree with the delegatee to grant the right to other subjects within delegation depth. The authorization step increases 1. Since we consider structured resources and privileges, there are three auth-delegation rules.
   $auth(issuer, S, right(Sn, P, O), T+1) \leftarrow$
   $delegate(issuer, delegatee, right(\square, P, O), k, 1),$
   $auth(delegatee, S, right(Sn, P, O), T).$
   $auth(issuer, S, right(Sn, P, SO), T+1) \leftarrow$
   $delegate(issuer, delegatee, right(\square, P, O), k, 1),$
   $auth(delegatee, S, right(Sn, P, SO), T),\ below(SO, O).$
   $auth(issuer, S, right(Sn, SP, O), T+1) \leftarrow$
   $delegate(issuer, delegatee, right(\square, P, O), k, 1),$
   $auth(delegatee, S, right(Sn, SP, O), T),\ below(SP, P).$

Dele-chain rules: The delegation can be redelegated within delegation depth. We also have three dele-chain rules for structured resources and privileges. Here we just give one of them.

$$delegate(issuer, S, right(\square, P, O), min(k\text{-}Step, Dep), 1 + Step) \leftarrow$$
$$delegate(issuer, delegatee, right(\square, P, O), k, 1),$$
$$delegate(delegatee, S, right(\square, P, O), Dep, Step),\ Step < k.$$

Self-delegation rule: The delegatee can delegate the right to himself/herself within k depth.

$$delegate(delegatee, delegatee, right, Dep, 1) \leftarrow$$
$$delegate(issuer, delegatee, right, k, 1),\ Dep \leq k.$$

Weak-delegation rule: If there is a delegation with k steps, we can get the delegation with steps less than k.

$$delegate(issuer, delegatee, right, Dep, 1) \leftarrow$$
$$delegate(issuer, delegatee, right, k, 1),\ Dep < k.$$

If $delegatee$ is a complex structure, *subject set, static threshold*, or *dynamic threshold*, we introduce a new group subject l_{new} to denote the subjects in complex structures, and replace the statement in program $\mathcal{P}$ using,

$$delegate(issuer, l_{new}, right, k, 1).$$

We need to add *auth-delegation* and *dele-chain* rules for them. There are similar rules for them, and here we present the rules for *subject set* structure.

Auth-delegation rule:

$$auth(issuer, S, right, T + 1) \leftarrow\ delegate(issuer, l_{new}, right, k, 1),$$
$$auth(s_1, S, right, T_1),\ \ldots,\ auth(s_n, S, right, T_n),\ T = max(T_1, \ldots, T_n).$$

Dele-chain rule:

$$delegate(sub, S, right, T_1, T_2) \leftarrow\ delegate(sub, l_{new}, right, k, 1),$$
$$delegate(s_1, S, right, Dep_1, Step_1), \ldots, delegate(s_n, S, right, Dep_n, Step_n),$$
$$T_1 = min(k\text{-}Step_1, \ldots, k\text{-}Step_n, Dep),$$
$$T_2 = max(1 + Step_1, \ldots, 1 + Step_n),\ T_1 > 0.$$

Query Transformation: In language $\mathcal{AL}$, there are two kinds of queries, single subject query and group subject query. We present the function $TransQuery(\mathcal{Q}_{\mathcal{AL}})$ for both of them and this function returns program Π and ground literals $\varphi(+)$ and $\varphi(-)$.

If $\mathcal{Q}_{\mathcal{AL}}$ is a single subject query, "$s\ requests\ right(+, p, o)$", $TransQuery$ returns program Π and ground literals $\varphi(+)$ and $\varphi(-)$ as follows respectively, $\{req(s, right(+, p, o))\}$, $grant(s, right(+, p, o))$ and $grant(s, right(-, p, o))$.

If $\mathcal{Q}_{\mathcal{AL}}$ is a group subject query, "$[s_1, s_2, \ldots, s_n]\ requests\ right(+, p, o)$". Program Π and ground literals $\varphi(+)$ and $\varphi(-)$ are as follows respectively,

$\{req(\ s_i, right(+, p, o)) \mid i = 1, \ldots, n\ \}$,

$ggrant(l, right(+, p, o))$ and $ggrant(l, right(-, p, o))$,

where l is a group subject entity to denote the set of subjects, $[s_1, \ldots, s_n]$.

4 A Scenario

In this section we represent a specific authorization scenario to demonstrate the features of language $\mathcal{AL}$.

Scenario: A company chooses to have multiparty control for emergency key recovery. If a key needs to be recovered, three persons are required to present their individual PINs. They are from different departments, a member of management, an individual from auditing, and one individual from IT department. The system trusts the manager of Human Resource Department to identify the staff of the company. The domain description $\mathcal{D}_{\mathcal{AL}}$ for this scenario is the following rules represented using language $\mathcal{AL}$.

$local$ grants $right(+, recover, key)$ to
[$dthreshold(1, X, hrM$ asserts $isAManager(X))$,
$dthreshold(1, Y, hrM$ asserts $isAnAuditor(Y))$,
$dthreshold(1, Z, hrM$ asserts $isATech(Z))$].
hrM asserts $isAManager(alice)$.
hrM asserts $isAnAuditor(bob)$.
hrM asserts $isAnAuditor(carol)$.
hrM asserts $isATech(david)$.

We translate them into language $\mathcal{L}_{Ans}$,

$auth(local, l_{key}, right(+, recovery, key), 1)$.
$holds(l_{key}, isAManager(X)) \leftarrow auth(local, l_{key}, right(+, recovery, key), 1),$
$assert(hrM, isAManager(X)),\ req(X, right(+, recovery, key))$.
$holds(l_{key}, isAnAuditor(X)) \leftarrow auth(local, l_{key}, right(+, recovery, key), 1),$
$assert(hrM, isAnAuditor(X)),\ req(X, right(+, recovery, key))$.
$holds(l_{key}, isATech(X)) \leftarrow auth(local, l_{key}, right(+, recovery, key), 1),$
$assert(hrM, isATech(X)),\ req(X, right(+, recovery, key))$.
$match(l_{key}, right(+, recovery, key)) \leftarrow$
$1\{holds(l_{key}, isAManager(X))\}1,$
$1\{holds(l_{key}, isAnAuditor(Y))\}1,$
$1\{holds(l_{key}, isATech(Z))\}1$.

In this scenario, the program $\mathcal{P}$ consists of the above translated rules, and those authorization rules we specified in section 3.1. If Alice, Bob, and David make a request to recover a key together, that is,

$[alice, bob, david]$ requests $right(+, recovery, key)$.

After translation, we get program Π,

$req(alice, right(+, recovery, key))$,
$req(bob, right(+, recovery, key))$,
$req(david, right(+, recovery, key))$,

and the ground literal $\varphi(+)$ is,

$ggrant(l, right(+, recovery, key))$,

where l is a group subject entity to represent the set of subjects, $[alice, bob, david]$.

Then program $\mathcal{P} \cup \Pi$ has only one answer set, and $ggrant(l, right(+, recovery, key))$ is in the answer set. Therefore the request is permitted.

Now if we consider that Alice, Bob, and Carol make the same request, they cannot satisfy the rule for $match$, then $ggrant(l, right(+, recovery, key))$ is not in the answer set. Instead, we get $ggrant(l, right(-, recovery, key))$, then the request will be denied.

5 Conclusion and Future Work

In this paper, we developed an expressive authorization language $\mathcal{AL}$ to specify the distributed authorization with delegation. We used Answer Set Programming as a foundational basis for its semantics. As we have showed earlier, $\mathcal{AL}$ has a rich expressive power which can represent positive and negative authorization, structured resources and privileges, partial authorization and delegation, and separation of duty. It is worth mentioning that language $\mathcal{AL}$ can represent all the scenarios discussed by Delegation Logic [9]. Moreover, as we have illustrated in section 4, $\mathcal{AL}$ can also represent complex authorization scenarios which Delegation Logic cannot.

We should indicate that our formulation also has implementation advantages due to recent development of Answer Set Programming technology in AI community.[1] The scenario in section 4 has been fully implemented through Answer Set Programming.

Our paper leave space for future work. One issue we plan to investigate is using preference of policy rules for conflict resolution which is more reasonable and flexible in some real applications. We also plan to investigate how to find the authorization path (Trust path) based on answer sets.

References

1. C. Baral. Knowledge Representation, Reasoning and Declarative Problem Solving. Cambridge University Press, 2003.
2. E. Bertino, F. Buccafurri, E. Ferrari, and P. Rullo. A Logical Framework for Reasoning on Data Access Control Policies. In *Proceedings of the 12th IEEE Computer Security Foundations Workshop(CSFW-12)*, pages 175-189, IEEE Computer Society Press, Los Alamitos, CA, 1999.
3. M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized Trust Management. In *Proceedings of the Symposium on Security and Privacy*, IEEE Computer Society Press, Los Alamitos,1996, pages 164-173.
4. M. Blaze, J. Feigenbaum, and M. Strauss. Compliance-checking in the PolicyMaker trust management system. In *Proceedings of Second International Conference on Financial Cryptography (FC'98)*, volume 1465 of Lecture Notes in Computer Science, pages 254-274. Springer, 1998.
5. M. Blaze, J. Feigenbaum, J. Ioannidis, and A. D. Keromytis. The Role of Trust Management in Distributed Systems. *Secure Internet Programming*, Lecture Note of Computer Science, vol. 1603, pages 185-210, Springer, Berlin, 1999.
6. S. Jajodia, P. Samarati, and V. S. Subrahmanian. Flexible Support for Multiple Access Control Policies. In*ACM Transactions on Database Systems*, Vol.26, No.2, June 2001, Pages 214-260.
7. N. Li, J. Feigenbaum, and B.N. Grosof. A logic-based knowledge representation for authorization with delegation (extended abstract). In *Proceedings of the IEEE Computer Security Foundations Workshop (CSFW-12)*(June). IEEE Computer Society Press, Los Alamitos, Calif., pages 162-174.
8. N. Li, W. H. Winsborough, and J. C. Mitchell. Distributed credential chain discovery in trust management. In *Journal of Computer Security*, volume 11, number 1, pages 35-86, February 2003.

[1] Please refer to http://www.tcs.hut.fi/Software/smodels/index.html

9. N. Li, B. N. Grosof, and J. Feigenbaum. Delegation Logic: A logic-based approach to distributed authorization. In*ACM Transactions on Information and System Security (TISSEC)*, February 2003.
10. T. Syrjänen. Lparse 1.0 User's Mannual. http://www.tcs.hut.fi/Software/smodels.
11. S. Wang, and Y. Zhang. Handling Distributed Authorization with Delegation through Answer Set Programming (manuscript). 2005.

Two Improved Partially Blind Signature Schemes from Bilinear Pairings

Sherman S.M. Chow*, Lucas C.K. Hui, S.M. Yiu, and K.P. Chow

Department of Computer Science
The University of Hong Kong
Pokfulam, Hong Kong
{smchow, hui, smyiu, chow}@cs.hku.hk

Abstract. A blind signature scheme is a protocol for obtaining a digital signature from a signer, but the signer can neither learn the messages he/she sign nor the signatures the recipients obtain afterwards. Partially blind signature is a variant such that part of the message contains pre-agreed information (agreed by the signer and the signature requester) in unblinded form, while threshold blind signature distributes the signing power to a group of signers such that a signature can only be produced by interacting with a predetermined numbers of signers. In this paper, we propose a threshold partially blind signature scheme from bilinear pairings and an ID-based partially blind signature scheme, which are provably secure in the random oracle model. To the best of authors' knowledge, we give the first discussion on these two notions.

Key words: threshold partially blind signature, identity-based partially blind signature, bilinear pairings

1 Introduction

A blind signature scheme is a protocol for obtaining a signature from a signer, but the signer can neither learn the messages he/she sign nor the signatures the recipients obtain afterwards. Blind signatures scheme is one of the examples of cryptographic schemes that have been employed extensively in privacy oriented e-services such as untraceable electronic cash (e.g. [5]), anonymous multiple choice electronic voting (e.g. [9]), or even in steganographic protocol (e.g. [12]).

The basic idea of most existing blind signature schemes is as follows. The requester (of the signature) randomly chooses some random factors and embeds them to the message to be signed. The random factors are kept in secret so the signer cannot recover the message. Using the blinded signature returned by the signer, the requester can remove the random factors introduced and get a valid signature. However, the property that requesters can ask the signer to blindly sign any message is undesirable in some situations. Consider using blind signature to design a e-cash scheme, expiry date information should be embedded

* corresponding author

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 316–328, 2005.

in the e-cash issued, or there may be unlimited growth of the bank's database for double-spending checking. Besides, the possibility of including embedded information may provide a more convenient way for inscribing the face value of the e-cash to the blind signature. Hence it is more flexible if the message to be signed is not "completely blind" and is able to embed some agreed information, which motivated the introduction of partially blind signature [1].

Recently, some pairing-based blind signature schemes were proposed, such as threshold blind signature in [20] and partially blind signature in [24]. Compared with previous blind signature schemes based on other difficult problems, their work have some nice properties like short signature size. In this paper, we propose two improved partially blind signature schemes from bilinear pairings.

1.1 Related Work

Pointcheval and Stern presented the formal definition and the security notion for blind signature in [15]. Unfortunately, [16] showed an inherent weakness in their result and presented a novel parallel one-more signature forgery attack. Partially blind signature was introduced in [1], together with a RSA-based scheme. This notion was formalized in [2], a discrete-logarithm based scheme that is provably secure was also proposed. A pairing-based blind signature was proposed in [3].

Another line of research efforts were done in combining the properties of other classes of cryptographic schemes into blind signatures. In [6] (shown to be insecure by [13]) and [8], forward-secure blind signature was proposed. Blind threshold signature that enables any t out of n legitimate signers to give a blind signature, was considered in [11] and [20].

As an alternative to traditional public key infrastructure (PKI), Shamir introduced identity-based (ID-based) signature schemes [19]. The distinguishing property of ID-based cryptography is that a user's public key can be any string, such as an email address, that can identify the user. This removes the need for users to look up the signer's public key before the verification of signature. Utilizing bilinear pairings, ID-based blind signature [23] and blind signcryption [22] were proposed.

1.2 Our Contribution

We propose two new partially blind signature schemes. The first one is a PKI-based partially blind signature scheme from bilinear pairings, which is more efficient for the signature requesters' side than the existing scheme [24]. Moreover, we discuss how to extend it into a threshold partially blind signature scheme. The second proposed scheme is an ID-based partially blind signature scheme. To the best of authors' knowledge, our schemes are the first of their kind.

1.3 Organization

The next section contains some preliminaries about bilinear pairing and the formal definitions of (ID-based) partially blind signature schemes. Section 3

presents our proposed schemes, followed by the analysis in Section 4. Finally, Section 5 concludes our paper.

2 Preliminaries

2.1 Bilinear Pairing and Gap Diffie-Hellman Groups

Bilinear pairing is an important cryptographic primitive (see [3,6,20,22,23,24]). Let $(\mathbb{G}_1, +)$ and $(\mathbb{G}_2, \cdot)$ be two cyclic groups of prime order q. The bilinear pairing is given as $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, which satisfies the following properties:

1. *Bilinearity*: For all $P, Q, R \in \mathbb{G}_1$, $\hat{e}(P+Q, R) = \hat{e}(P, R)\hat{e}(Q, R)$, and $\hat{e}(P, Q+R) = \hat{e}(P, Q)\hat{e}(P, R)$.
2. *Non-degeneracy*: There exists $P, Q \in \mathbb{G}_1$ such that $\hat{e}(P, Q) \neq 1$.
3. *Computability*: There exists an efficient algorithm to compute $\hat{e}(P, Q)$ $\forall P, Q \in \mathbb{G}_1$.

Definition 1. *Given a generator P of a group $\mathbb{G}_1$ and a 3-tuple (aP, bP, cP), the Decisional Diffie-Hellman (DDH) problem is to decide if $c = ab$.*

Definition 2. *Given a generator P of a group $\mathbb{G}_1$ and a 2-tuple (aP, bP), the Computational Diffie-Hellman (CDH) problem is to compute abP.*

Definition 3. *If $\mathbb{G}_1$ is a group such that DDH problem can be solved in polynomial time but no probabilistic algorithm can solve CDH problem with non-negligible advantage within polynomial time, we call it a Gap Diffie-Hellman (GDH) group.*

We assume the existence of a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ that one can solve DDH problem in polynomial time.

2.2 Notations

The definitions of $\mathbb{G}_1$, $\mathbb{G}_2$ and $\hat{e}(\cdot, \cdot)$ will be used throughout the rest of the paper. Besides, we let $H(\cdot)$ and $H_0(\cdot)$ be two cryptographic hash functions where $H_0 : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$ and $H : \{0,1\}^* \rightarrow \mathbb{G}_1$.

2.3 Framework of (ID-based) Partially Blind Signature

A partially blind signature scheme consists of four algorithms: `Setup`, `KeyGen`, `Issue`, and `Verify`. `Issue` is an interactive protocol between the signer and the requester which consists of three sub-algorithms: `Blind`, `Sign` and `Unblind`.

- `Setup`: On an unary string input 1^k where k is a security parameter, it produces the public parameters *params*, which include a description of a finite signature space, a description of a finite message space together with a description of a finite agreed information space.
- `KeyGen`: On a random string input x, it outputs the signer's secret signing key sk and its corresponding public verification key pk.
- `Issue`: Suppose the requester wants a message m to be signed, after the execution of four sub-algorithms, a signature σ will be produced. The agreed information c is assumed to be negotiated beforehand.
  - `Blind`: On a random string r, a message m and agreed information c as the input, it outputs a string h to be signed by the signer, h is sent to the signer by this algorithm.
  - `Sign`: On a string h and the signer's private signing key sk as the input, it outputs a blind signature $\bar{\sigma}$ to be unblinded by the requester, $\bar{\sigma}$ is sent to the requester by this algorithm.
  - `Unblind`: On a signature $\bar{\sigma}$ and the previous used random string r, it outputs the unblinded signature σ.
- `Verify`: On an unblinded signature σ, a message m, a negotiated information c and the signer's public verification key pk as the input, it outputs $\top$ for "true" or $\perp$ for "false", depending on whether σ is a valid signature signed by the signer with the corresponding private key pk on a message m and agreed information c.

The framework of ID-based partially blind signature schemes is similar to that of its PKI counterpart. The differences are described below.

- `Setup`: This algorithm is usually executed by the private key generator (PKG). On an unary string input 1^k where k is a security parameter, it produces the public parameters *params*, which include a description of a finite signature space, a description of a finite message space together with a description of a finite agreed information space. The master secret s is part of the output too, which is kept secret.
- `KeyGen`: On an arbitrary string input ID, it computes the private signing key S_{ID} with the help of master secret s, and the corresponding public verification key Q_{ID}, with respect to *params*.

These algorithms must satisfy the standard consistency constraint of the partially blind signature, i.e. if $(\sigma, c) = \texttt{Issue}(m, r, sk)$, $\texttt{Verify}(pk, m, c, \sigma) = \top$ must hold (same for ID-based scheme, where (pk, sk) is replaced by (Q_{ID}, S_{ID})).

2.4 Unforgeability of ID-based Partially Blind Signature

The notion of existential unforgeability of partially blind signature (in PKI-based settings) was formally defined in [2]. We extend the notion in [2] to the ID-based settings in terms of the *existential unforgeability of ID-based partially blind signature under adaptive chosen-message-and-identity attack* (EUF-IDPB-CMIA2) game played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

EUF-IDPB-CMIA2 Game:

Setup: The challenger $\mathcal{C}$ takes a security parameter k and runs the `Setup` to generate public parameters *param* and also the master secret key s. $\mathcal{C}$ sends *param* to $\mathcal{A}$.

Attack: The adversary $\mathcal{A}$ can perform a polynomially bounded number of the following types of queries in an adaptive manner (i.e. each query may depend on the responses to the previous queries).

- Hash functions queries: $\mathcal{A}$ can ask for the value of the hash functions ($H(\cdot)$ and $H_0(\cdot)$ in our schemes) for the requested input.
- `KeyGen`: $\mathcal{A}$ chooses an identity ID. $\mathcal{C}$ computes $\texttt{Extract}(ID) = S_{ID}$ and sends the result to $\mathcal{A}$. The corresponding public verification key Q_{ID} can be calculated by using the hash function $H(\cdot)$.
- `Issue`: $\mathcal{A}$ chooses an identity ID, a plaintext m and the negotiated information c. $\mathcal{C}$ issues the signature by computing $\sigma = \texttt{Issue}\ (m, c, S_{ID})$ and sends σ to $\mathcal{A}$.

Forgery: The adversary $\mathcal{A}$ outputs (σ, ID, m, c) where (ID, m, c) and ID were not used in any of the `Issue` and `Extract` queries, respectively, in the Attack phase. The adversary wins the game if the response of the `Verify` on (ID, m, c, σ) is not equal to $\perp$.

The advantage of $\mathcal{A}$ is defined as the probability that it wins.

Definition 4. *An ID-based partially blind scheme is defined to be existential unforgeable against adaptive chosen-message-and-identity attacks if no adversary has a non-negligible advantage in the EUF-IDPB-CMIA2 game.*

2.5 Partial Blindness

In the normal sense of blindness, the signer can learn no information on the message to be signed. If the signer can link the signature to the instance of the signing protocol, then the blindness is lost.

In partially blind signature, a piece of information must be agreed by both the signer and the requester. If the signer embed an unique piece of the agreed information c in each message to be signed, it is easy to see that the signer can link the signature to the instance of the signing protocol by using the agreed information as an index, and hence the blindness property will be lost. For the scheme to be practical, the cardinality of the finite agreed information space should be small compared with the anticipated number of total `Issue` requests. This weakness is inherent to any partial blind signature schemes as it is the price for embedding agreed information to the message to be signed.

So the normal sense of blindness is not applicable in our situation. The extended notion of partial blindness is defined in terms of the *Unlinkability Game* (UL) played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. Again, we adopt a similar notion as [2].

Unlinkability Game:

Setup: The challenger $\mathcal{C}$ takes a security parameter k and runs the `Setup` to generate public parameters *param* (and also the master secret key s in ID-based case). $\mathcal{C}$ sends *param* to $\mathcal{A}$.

Preparation: The adversary $\mathcal{A}$ chooses two distinct messages m_0 and m_1, together with the agreed information c. For the ID-based case, the adversary $\mathcal{A}$ also chooses an identity *ID* and sends them to $\mathcal{C}$.

Challenge: The challenger $\mathcal{C}$ chooses a random bit b secretly, and then ask the adversary $\mathcal{A}$ to partially sign on the message m_b with agreed information c and m_{1-b} with the same piece of agreed information c. After $\mathcal{C}$ unblinds both signatures, it presents the signature of m_b to $\mathcal{A}$.

Response: The adversary $\mathcal{A}$ returns the guess b' and wins if $b' = b$.

The advantage of $\mathcal{A}$ is defined as $Adv(\mathcal{A}) = |2P[b' = b] - 1|$ where $P[b' = b]$ denotes the probability that $b' = b$.

Definition 5. *An (ID-based) partially blind scheme is said to have the perfect partial blindness property if any adversary has zero advantage in the UL game.*

3 Our Proposed Schemes

3.1 PKI-based Partially Blind Signature

`Setup`: The system parameters are $params = \{\mathbb{G}_1, \mathbb{G}_2, \hat{e}(\cdot,\cdot), q, P, H(\cdot), H_0(\cdot)\}$.

`KeyGen`: The signer randomly selects $s \in_R \mathbb{Z}_q^*$ and computes $P_{pub} = sP$ as his/her public verification key. The signing key is s and is kept in secret.

`Issue`: Suppose the requester now wants to get the signature of message m and the requester has already negotiated with the signer with public key P_{pub} on the agreed information c to be attached to the message. The interaction between the requester and the signer is as follows:

- `Sign` (Part 1): The signer randomly chooses $r \in_R \mathbb{Z}_q^*$, computes $Z = H(c)$, $Y = rZ$ and sends Y to the requester. Notice that the `Sign` algorithm has not finished yet.
- `Blind`: The requester randomly picks $\alpha \in_R \mathbb{Z}_q^*$ and $\beta \in_R \mathbb{Z}_q^*$, sends $h = \alpha^{-1}H_0(m, Y') + \beta$ to the signer and computes $Y' = \alpha Y + \alpha\beta H(c)$.
- `Sign` (Part 2): The signer computes $S = (r + h)sZ$ and sends it to the requester. Now the `Sign` algorithm has been finished.
- `Unblind`: The requester unblinds the received S by $S' = \alpha S$.

Finally (Y', S', m, c) is the partially blind signature of message m and agreed information c.

`Verify`: Any verifier (including the signature requester) can verify the validity of the partially blind signature by checking whether the equation $\hat{e}(S', P) = \hat{e}(Y' + H_0(m, Y')H(c), P_{pub})$ holds. If so, the partially blind signature is accepted.

3.2 Threshold Partially Blind Signature

To extend our proposed partially blind signature scheme into the threshold version, we need the help of the following techniques in threshold cryptography.

Polynomial Interpolation Secret Sharing [18]: Many threshold schemes are based on Shamir's secret sharing, which is derived from the concept of Lagrange polynomial interpolation.

For a (t, n) instantiation (i.e. any t out of n pieces of share can be used to reconstruct the secret, but no one can get the secret with the knowledge of only $t-1$ of them), a trusted dealer first selects t random coefficients $a_0, a_1, \cdots, a_{t-1}$ from $\mathbb{Z}_q$ where a_0 is the master secret to be shared. Then n different public points $x_{i_j} \in \mathbb{Z}_q^*$ are chosen (where $1 \leq j \leq n$), one for each participant. Let f be a polynomial of degree $t-1$ and $f(x) = a_0 + a_1 x + \cdots + a_{t-1}x^{t-1}$, the share to be distributed to the participant with public point x_{i_j} assigned is $f(x_{i_j})$.

When t participants decided to reconstruct the secret, they can do so by recovering the polynomial. With the knowledge of t points $(x_{i_j}, f(x_{i_j}) = s_{i_j})$ on the curve, the coefficients $(a_0, \cdots, a_t)$ of f are uniquely determined by the Lagrange interpolation as below.

$$f(x) = \sum_{j=1}^{t} s_{i_j} \prod_{1 \leq l \leq t, l \neq j} \frac{x - x_{i_l}}{x_{i_j} - x_{i_l}}.$$

So $a_0 = f(0)$ can be obtained by $\sum_{j=1}^{t} b_j s_{i_j}$, where $b_j = \prod_{1 \leq l \leq t, l \neq j} \frac{x_{i_l}}{x_{i_l} - x_{i_j}}$.

Joint Random Secret Sharing (JRSS) [14]: In this protocol, each player can collectively generate a random secret and each of them can receive a (t, n)-secret sharing of this random value. Basically, this can be achieved by asking each participant to share his/her own random secret with the remaining participants by a (t, n)-secret sharing, and the final random secret shared by all these players is the sum of the random value selected by each participant.

Multiplication of Two Shared Secrets [10]: Two values shared by the (t, n)-secret sharing can be multiplied without revealing any information about the shares (except the wanted result of their products). The principle behind is as follows. Suppose there are two polynomials of degree $t-1$ for the (t, n)-secret sharing of value r and s respectively, their multiplications gives another polynomial of degree $2t-2$, which can be used for a $(2t-1, n)$-secret sharing of the products of r and s. However, this "newly generated" polynomial is not randomly generated anymore. To avoid leaking any information about r and s, we need to "re-randomize" it by using joint random secret sharing of a zero-value (such that the polynomial is randomized but the value to be shared remains unchanged).

Now we describe the $(2t-1, n)$ threshold extension of our scheme. Firstly, the shares s_i of the secret key s is generated by a (t, n)-JRSS. For signing, any $2t-1$ of the n signers jointly execute a $(t, 2t-1)$-JRSS to generate the random value r, and compute the value of $Y = rZ$ where $Z = H(c)$. The shares r_i of r are distributed to the participating $2t-1$ signers. Each of them executes a

$(2t-1, 2t-1)$-JRSS of a zero-value to get the shares c_i. After received the value of h from the requester, each signer increments his/her share r_i by $r'_i = r_i + h$, the value of $(r+h)s$ can be recovered by these $2t-1$ signers, by interpolating the value of $r'_i s_i + c_i$ from each of them. Hence these signers can compute the blinded signature $S = (r+h)sZ$ to be sent to the requester, by the point scalar multiplication of their shares with Z.

3.3 ID-based Partially Blind Signature

`Setup`: The PKG randomly chooses $s \in_R \mathbb{Z}_q^*$. The master secret is s. The system parameters are $params = \{\mathbb{G}_1, \mathbb{G}_2, \hat{e}(\cdot,\cdot), q, P, P_{pub}, H(\cdot), H_0(\cdot)\}$.

`KeyGen`: The signer with identity $ID \in \{0,1\}^*$ submits ID to PKG. PKG sets the signer's public key Q_{ID} to be $H(ID) \in \mathbb{G}_1$, computes the signer's private signing key S_{ID} by $S_{ID} = sQ_{ID}$ Then PKG sends the private signing key to the signer.

`Issue`: Suppose the requester now wants to get the signature of message m and the requester has already negotiated with the signer of identity ID on the negotiated information c to be attached to the message. The interaction between the requester and the signer is as follows:

- `Sign` (Part 1): The signer randomly chooses $r \in_R \mathbb{Z}_q^*$, computes $C = rP$, $Y = rQ_{ID}$ and sends (Y, C) to the requester. Notice that the `Sign` algorithm has not finished yet.
- `Blind`: The requester randomly picks α, β and $\gamma \in_R \mathbb{Z}_q^*$, computes $Y' = \alpha Y + \alpha\beta Q_{ID} - \gamma H(c)$, $C' = \alpha C + \gamma P_{pub}$, $h = \alpha^{-1} H_0(m, Y') + \beta$ and sends h to the signer.
- `Sign` (Part 2): The signer computes $S = (r+h)S_{ID} + rH(c)$ and sends it to the requester. Now the `Sign` algorithm has been finished.
- `Unblind`: The requester unblinds the received S by $S' = \alpha S$.

Finally (Y', C', S', m, c) is the partially blind signature of the message m and the agreed information c.

`Verify`: Any verifier (including the signature requester) can verify the validity of the ID-based partially blind signature by checking whether the equation $\hat{e}(S', P) = \hat{e}(Y' + H_0(m, Y')Q_{ID}, P_{pub})\hat{e}(H(c), C')$ holds. If so, the partially blind signature is accepted as valid.

4 Analysis of the Proposed Schemes

4.1 Efficiency Analysis

We consider the costly operations which include point addition on $\mathbb{G}_1$ ($\mathbb{G}_1$ Add), point scalar multiplication on $\mathbb{G}_1$ ($\mathbb{G}_1$ Mul), multiplication in $\mathbb{Z}_q$ ($\mathbb{Z}_q$ Mul), division in $\mathbb{Z}_q$ ($\mathbb{Z}_q$ Div), hashing into the group (MapToPoint, the hash operation

Algorithms	Efficiency					
	G_1 Add	G_1 Mul	Z_q Mul	Z_q Div	MapToPoint	Pairing
Existing Partially Blind Signature [24]						
`Issue(Signer)`	0	1	0	1	0	0
`Issue(Requester)`	3	3	0	0	1	0
`Verify`	1	1	0	0	1	2
Proposed PKI-based Partially Blind Signature						
`Issue(Signer)`	0	2	1	0	1	0
`Issue(Requester)`	1	3	0	1	1	0
`Verify`	1	1	0	0	1	2
Proposed ID-based Partially Blind Signature						
`Issue(Signer)`	1	4	0	0	1	0
`Issue(Requester)`	3	6	0	1	1	0
`Verify`	1	1	0	0	1	3

Table 1. Efficiency of our Proposed Schemes

in BLS short signature scheme [4]) and pairing operation (Pairing). Table 1 shows a summary of the efficiency of our schemes and also the revised scheme in [24].

The signature requesters usually posses less computational power than the signature issuer. Comparing our proposed schemes with the scheme in [24] (PKI-based but not ID-based), our PKI-based scheme is more efficient on the requesters' side, while our ID-based scheme only requires one more inversion in $\mathbb{Z}_q$ and three more point scalar multiplications.

4.2 Security Analysis

Theorem 1 *In the random oracle model, if there is an algorithm $\mathcal{A}$ for an adaptively chosen message attack to our scheme. Then, there exists an algorithm $\mathcal{C}$ that can solve the CDH problem.*

Theorem 2 *Our partially PKI-based blind signature scheme satisfies the partial blindness property in information theoretic sense.*

Theorem 3 *In the random oracle model, if there is an algorithm $\mathcal{A}$ for an adaptively chosen message and ID attack to our scheme. Then, there exists an algorithm $\mathcal{C}$ that can solve the CDH problem.*

Theorem 4 *Our ID-based partially blind signature scheme satisfies the partial blindness property in information theoretic sense.*

Proof. Refer to the full version [7] for the proof of the above theorems. □

4.3 Changing Agreed Information Attack

Changing agreed information attack is the attack in which the requester, after obtained the signature issued by the signer, can subsequently change the agreed information c to another one c' on his/her wish, yet the signature remains valid. In both of our schemes, since r (in ID-based scheme) and s (in PKI-based scheme) are unknown to the requester, changing $H(c)$ to $H(c')$ involves solving the CDH problem, which is computationally infeasible.

5 Conclusion

In this paper, we propose two improved partially blind signature schemes. One is a PKI-based threshold partially blind signature scheme while another one is an ID-based partially blind signature scheme. To the best of authors' knowledge, our schemes are the first of their kind. The proposed schemes are provably secure in the random oracle model. Future research directions include finding a formal proof of security against the parallel one-more signature forgery attack.

Acknowledgement

The authors would like to thank Dr. Fangguo Zhang for pointing out the mistake of the preliminary version of this paper and all the anonymous reviewers for their helpful comments.

References

1. Masayuki Abe and Eiichiro Fujisaki. How to Date Blind Signatures. *Advances in Cryptology - ASIACRYPT 1996*, *LNCS* 1163, pp. 244–251. Springer.
2. Masayuki Abe and Tatsuaki Okamoto. Provably Secure Partially Blind Signatures. *Advances in Cryptology - CRYPTO 2000*, *LNCS* 1880, pp. 271–286. Springer.
3. Alexandra Boldyreva. Threshold Signature, Multisignature and Blind Signature Schemes Based on the Gap-Diffie-Hellman-Group Signature Scheme. *Public Key Cryptography - PKC 2003*, *LNCS* 2567, pp. 31–46. Springer.
4. Dan Boneh, Ben Lynn, and Hovav Shacham. Short Signatures from the Weil Pairing. *Advances in Cryptology - ASIACRYPT 2001*, *LNCS* 2248, pp. 514–532.
5. David Chaum, Amos Fiat, and Moni Naor. Untraceable Electronic Cash. *Advances in Cryptology - CRYPTO 1988*, *LNCS* 403, pp. 319–327. Springer.
6. Sherman S.M. Chow, Lucas C.K. Hui, S.M. Yiu, and K.P. Chow. Forward-Secure Multisignature and Blind Signature Schemes. *Applied Mathematics and Computation*, Accepted, 2004.
7. Sherman S.M. Chow, Lucas C.K. Hui, S.M. Yiu, and K.P. Chow. Two Improved Partially Blind Signature Schemes from Bilinear Pairings. Full version. Available at Cryptology ePrint Archive, 2004/108.
8. Dang Nguyen Duc, Jung Hee Cheon, and Kwangjo Kim. A Forward-Secure Blind Signature Scheme based on the Strong RSA Assumption. *International Conference on Information and Communications Security - ICICS 03*, *LNCS* 2836, pp. 11–21.

9. Wen-Shenq Juang and Chin-Laung Lei. A Secure and Practical Electronic Voting Scheme for Real World Environments. *TIEICE: IEICE Transactions on Communications/Electronics/Information and Systems*, 1997.
10. Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Robust Threshold DSS Signatures. *Advances in Cryptology - EUROCRYPT '96*, *LNCS* 1070, pages 354–371. Springer.
11. Jinho Kim, Kwangjo Kim, and Chulsoo Lee. An Efficient and Provably Secure Threshold Blind Signature. *International Conference on Information Security and Cryptology - ICISC 2001*, *LNCS* 2288, pp. 318–327. Springer.
12. Józef Lenti, István Loványi, and Ákos Nagy. Blind Signature Based Steganographic Protocol. In *IEEE International Workshop on Intelligent Signal Processing*, 2001.
13. Lihua Liu and Zhengjun Cao. Universal Forgeability of a Forward-Secure Blind Signature Scheme Proposed by Duc et al. Cryptology ePrint Archive, 2004/262.
14. Torben P. Pedersen Distributed Provers with Applications to Undeniable Signatures. *Advances in Cryptology - EUROCRYPT '91, LNCS* 547, pages 221–242. Springer.
15. David Pointcheval and Jacques Stern. Provably Secure Blind Signature Schemes. *Advances in Cryptology - ASIACRYPT 1996, LNCS* 1163, pp. 252–265. Springer.
16. Claus-Peter Schnorr. Security of Blind Discrete Log Signatures against Interactive Attacks. *International Conference Information and Communications Security - ICICS 2001, LNCS* 2229, pp. 1–12. Springer.
17. Claus Peter Schnorr. Enhancing the Security of Perfect Blind DL-Signatures. Manuscript, December 2003.
18. Adi Shamir. How to Share A Secret. *Communications of the ACM*, 22(11):612–613, 1979.
19. Adi Shamir. Identity-Based Cryptosystems and Signature Schemes. *Advances in Cryptology - CRYPTO 1984 LNCS* 196, pp. 47–53. Springer-Verlag.
20. Duc Liem Vo, Fangguo Zhang, and Kwangjo Kim. A New Threshold Blind Signature Scheme from Pairings. In *Symposium on Cryptography and Information Security, SCIS2003*, volume 1/2, pp. 233–238.
21. David Wagner. A Generalized Birthday Problem. *Advances in Cryptology - CRYPTO 2002*, *LNCS* 2442, pages 288–303..
22. Tsz Hon Yuen and Victor K. Wei. Fast and Proven Secure Blind Identity-Based Signcryption from Pairings. *Topics in Cryptology - CT-RSA 2005, LNCS* 3376, pp. 305–322, Springer.
23. Fangguo Zhang and Kwangjo Kim. Efficient ID-Based Blind Signature and Proxy Signature from Bilinear Pairings. *Australasian Conference on Information Security and Privacy - ACISP 2003, LNCS*, 2727 pp. 312–323. Springer.
24. Fangguo Zhang, Reihaneh Safavi-Naini, and Willy Susilo. Efficient Verifiably Encrypted Signature and Partially Blind Signature from Bilinear Pairings. *Progress in Cryptology - INDOCRYPT 2003, LNCS* 2904, pp. 191–204, Springer. Revised version available at http://www.uow.edu.au/∼wsusilo.

Appendix

We remark that the security of our schemes also depends on the intractability of the ROS (find an Overdetermined, Solvable system of linear equations modulo q with Random inhomogeneities) problem.

Definition 6. *Given an oracle random function* $F : {\mathbb{Z}_q}^l \rightarrow \mathbb{Z}_q$*, the ROS problem is to find coefficients* $a_{k,i} \in \mathbb{Z}_q$ *and a solvable system of* $l+1$ *distinct equations (1) in the unknown* $c_1, c_2, \cdots, c_l$ *over* $\mathbb{Z}_q$*:*

$$a_{k,1}c_1 + \cdots + a_{k,l}c_l = F(a_{k,1}, \cdots, a_{k,1}) \text{ for } k = 1, 2, \cdots, t. \tag{1}$$

Now we describe how an adversary $\mathcal{A}$ that is able to solve ROS problem efficiently can get $l+1$ valid ID-based partially blind signature associated with the *same* agreed information c by requesting only l signatures from the *same* signature issuer $\mathcal{S}$ of identity ID.

1. $\mathcal{S}$ sends commitments $C_1 = r_1P$, $C_2 = r_2P$, $\cdots$, $C_l = r_lP$ and $Y_1 = r_1Q_{ID}$, $Y_2 = r_2Q_{ID}$, $\cdots$, $Y_l = r_lQ_{ID}$ to $\mathcal{A}$.
2. $\mathcal{A}$ chooses randomly $a_{k,1}, a_{k,2}, \cdots a_{k,l}$ from $\mathbb{Z}_q$ and messages $m_1, m_2, \cdots, m_t$ and computes $f_k = \sum_{i=1}^{l} (a_{k,i}Y_i)$ and $H_0(m_k, f_k)$ for $k = 1, 2, \cdots, t$ where $l+1 \leq t < q_{H_0}$, the maximum number of queries of H_0 issued by $\mathcal{A}$.
3. $\mathcal{A}$ solves the ROS-problem: $l+1$ of equations (2) in the unknowns $c_1, c_2, \cdots, c_l$ over $\mathbb{Z}_q$:

$$\sum_{j=1}^{l} (a_{k,j}c_j) = H_0(m_k, f_k) \text{ for } k = 1, 2, \cdots, t. \tag{2}$$

4. $\mathcal{A}$ sends the solutions $c_1, c_2, \cdots, c_l$ as the challenge (value to be signed) to $\mathcal{S}$.
5. $\mathcal{S}$ sends back $S_i = (r_i + c_i)S_{ID} + r_iH(c)$ for $i = 1, 2, \cdots, l$.
6. For each solved equation (2), $\mathcal{A}$ gets a valid signature $({Y_k}', {C_k}', {S_k}')$ on message m_k by setting ${Y_k}' = f_k$, ${C_k}' = \sum_{j=1}^{l} a_{k,j}C_j$ and ${S_k}' = \sum_{j=1}^{l} a_{k,j}S_j$.

Now we show these $l+1$ signatures are valid.

$$\begin{aligned}
\hat{e}({S_k}', P) &= \hat{e}(\sum_{j=1}^{l} a_{k,j}S_j, P) \\
&= \hat{e}(\sum_{j=1}^{l} a_{k,j}[(r_j + c_j)S_{ID} + r_jH(c)], P) \\
&= \hat{e}(S_{ID}, P)^{\sum_{j=1}^{l} a_{k,j}r_j} \hat{e}(S_{ID}, P)^{\sum_{j=1}^{l} a_{k,j}c_j} \hat{e}(H(c), \sum_{j=1}^{l} a_{k,j}r_jP) \\
&= \hat{e}(\sum_{j=1}^{l} a_{k,j}r_jQ_{ID}, P_{pub})\hat{e}(Q_{ID}, P_{pub})^{H_0(m_k, f_k)} \hat{e}(H(c), \sum_{j=1}^{l} a_{k,j}r_jP) \\
&= \hat{e}(\sum_{j=1}^{l} a_{k,j}Y_j, P_{pub})\hat{e}(H_0(m_k, f_k)Q_{ID}, P_{pub})\hat{e}(H(c), \sum_{j=1}^{l} a_{k,j}C_j) \\
&= \hat{e}({Y_k}' + H_0(m_k, {Y_k}'), P_{pub})\hat{e}(H(c), {C_k}')
\end{aligned}$$

A similar attack can be applied on our PKI-based partially blind signature if an adversary can solve ROS problem efficiently. However, ROS problem is "a plausible but novel complexity assumption" [16]. We refer interested reader to [17] and [21] for more discussions on the relationship between ROS problem and blind signature schemes.

On the Security of Nominative Signatures

Willy Susilo and Yi Mu

Centre for Information Security Research
School of Information Technology and Computer Science
University of Wollongong
Wollongong 2522, AUSTRALIA
{wsusilo, ymu}@uow.edu.au

Abstract. Nominative signatures are the dual scheme of undeniable signatures, where *only* the nominee can verify the nominator (signer)'s signature and if necessary, *only* the nominee can prove to the third party that the signature issued to him (her) is valid. The first construction was proposed by Kim, Park and Won [7] and it was shown in the recent work of Huang and Wang in ACISP 2004 [5] that Kim-Park-Won's scheme does not satisfy the goal of nominative signatures. Moreover, Huang and Wang suggested a new nominative signature scheme in the same paper. They claimed that the new scheme satisfies all requirements of nominative signatures. In this paper, we show that Huang and Wang's scheme *does not* satisfy one important property of nominative signatures, namely the nominator (signer) can also verify the validity of the published signature. Moreover, we will show that the nominator can always show to anyone that the signature is indeed a valid signature *without* any cooperation from the nominee. Hence, the scheme is *not nominative*, since it does not satisfy the requirements of nominative signatures. Finally, we also discuss the security assumption that needs to be satisfied to obtain secure and efficient nominative signatures.

Keywords: Cryptography, Digital Signatures, Nominative Signatures, Convertible, Untransferable

1 Introduction

Digital signature schemes are the most important cryptographic primitive for providing authentication in an electronic world. Digital signatures, introduced in the pioneering paper of Diffie and Hellman [4], allow a signer with a secret key to sign messages such that *anyone* with access to the corresponding public key be able to verify authenticity of the message. The "public-verifiable" property is quite suitable for some uses, but it is unsuitable for many other applications, for example where a signed message is personally or commercially sensitive, such as in a bill of tax, a bill of health, etc. Hence, it is preferable to place some restrictions on this property to prevent potential misuse of signatures.

There are several works to provide some restrictions on the publicly-verifiability of digital signatures, including [3,6,7]. In this paper, we consider the

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 329–335, 2005.

notion of nominative signatures introduced by Kim, Park and Won [7], which is the dual signature scheme of the undeniable signature [3], in which *not* the signer (or nominator) *but only* the recipient (or nominee) can control the use of the signatures. The first construction of nominative signature scheme is the scheme proposed in [7]. Recently, in ACISP 2004, Huang and Wang showed that this scheme violates the definition of nominative signatures, in the sense the the nominator (or signer) can always verify the signature by himself. Moreover, Huang and Wang proposed a nominative signature scheme in [5] and claimed that the scheme satisfies all the requirements of nominative signatures. They also extended the notion of nominative signatures to convertible nominative signatures, where the nominee (and not the nominator) can convert the signature to a public-verifiably signature.

Our Contributions
In this paper, we show that the scheme in [5] does not satisfy the important property of nominative signatures, namely the nominator can also verify the authenticity of the signature. Moreover, the nominator can always publish the signature and make it publicly verifiable.

1.1 Cryptographic Tools: Signature of Knowledge

The first signature of knowledge (SPK) was proposed in [2,1]. We will use the following definition of signature of knowledge from [2].

Definition 1. *[2] A pair* $(c, s) \in \{0,1\}^{\ell} \times \mathbb{Z}_q$ *satisfying*

$$c = H(S||V||m) \text{ with } S = g||y \text{ and } V = g^s y^c \pmod{p}$$

is a signature knowledge of the discrete logarithm of a group element y *to the base* g *of the message* $m \in \{0,1\}^*$ *and is denoted*

$$SPKLOG\{\alpha : y = g^{\alpha}\}(m).$$

An $SPKLOG\{\alpha : y = g^{\alpha} \pmod{p}\}(m)$ can be computed if the value (secret key) $\alpha = \log_g(y)$ is known, by selecting a random integer $r \in \mathbb{Z}_q$ and computing $t = g^r \pmod{p}$ and then c and s according to

$$c = H(g||y||t||m)$$

and

$$s = r - c\alpha \pmod{q}.$$

This is also known as a non-interactive proof of the knowledge α.

1.2 Organization of the Paper

The rest of this paper is organized as follows. In section 2, we review the notion of nominative signatures, together with the scheme proposed in [5]. In section 3, we present a flaw to the proposed scheme in [5] and show that it violates the basic requirement of nominative signatures. In section 4, we discuss the security assumptions required to construct secure nominative signatures. Section 5 concludes the paper.

2 Nominative Signatures

Unlike ordinary digital signature schemes, nominative signatures are aimed to achieve the following objectives [7]:

- Only nominee (recipient) can verify the nominator (signer)'s signature.
- If necessary, only nominee (recipient) can prove to the third party that the signature is issued to him (her) and is valid.

It is noted in [7] that to construct a nominative signature scheme, the following two conditions must be satisfied:

- Only nominee can verify the nominator's signature S. Even the nominator himself *cannot* verify the signature S.
- If necessary, only nominee can prove to the third party that the signature S was indeed valid. Even the nominator cannot provide a proof that the signature S is valid.

The first nominative signature scheme is the scheme proposed in [7]. Unfortunately, recently, Huang and Wang [5] showed that the proposed scheme in [7] does not satisfy one of the requirements of nominative signatures, namely the nominator (signer) can verify the signature and provide a proof of the validity of the signature to any third party. In the same paper, Huang and Wang presented a scheme that is claimed to satisfy all the requirements of nominative signatures [5]. They also extended the notion of nominative signatures to convertible nominative signatures, where the nominee can convert the nominative signature to a universally verifiable signature. In the next section, we will review Huang and Wang's scheme from [5].

2.1 Huang and Wang's Nominative Signature Scheme [5]

Huang and Wang's nominative signature scheme was proposed in [5]. The aim of this scheme is to achieve a secure nominative signature, without the flaw that has been identified in the previous scheme [7]. Additionally, Huang and Wang also proposed a conversion algorithm to allow their nominative signature scheme to be converted into universally verifiable signatures. We omit the description of conversion algorithm and refer the reader to [5] for a more complex account.

The cryptographic setting for Huang and Wang's algorithm is as follows. Let p, q be large prime numbers, where $q|p-1$, and g be a generator in $\mathbb{Z}_p^*$. The nominator S's secret key is $x_S \in \mathbb{Z}_q^*$, the corresponding public key is $y_S = g^{x_S} \pmod p$, and the nominee V's secret key is $x_V \in \mathbb{Z}_q^*$, where x_V is odd, and the corresponding public key is $y_V = g^{x_V} \pmod p$. Let H denote a secure one-way hash function and $||$ denote concatenation between strings. The scheme consists of three algorithms as follows.

- *Signing.* The signing algorithm is an interactive algorithm between the nominator and the nominee as follows.

1. The nominee chooses random numbers $R_1, R_2 \in_R \mathbb{Z}_q^*$ and computes

$$a = g^{R_1} \pmod{p},$$
$$c = y_v^{R_2} \pmod{p}$$

and sends (a, c) to the nominator.
2. The nominator selects a random number $r \in_R \mathbb{Z}_q^*$ and computes

$$b = ag^{-r} \pmod{p},$$
$$e = H(y_V||b||c||m),$$
$$s' = r - x_S e \pmod{q}$$

then sends (e, b, s') to the nominee.
3. The nominee accepts iff

$$e \stackrel{?}{=} H(y_V||b||c||m) \text{ and } g^{s'} y_S^e b \stackrel{?}{=} a \pmod{p}$$

hold with equality. If the nominee accepts, she computes

$$s = s' + R_2 - R_1 \pmod{q}$$

and publishes the signature $\sigma = (b, c, s)$. Otherwise, she outputs $\perp$.

– *Verification.* Given a signature $\sigma = (b, c, s)$ on a message m, the nominee computes $e = H(y_V||b||c||m)$ and checks whether

$$(g^s y_S^e b)^{x_V} \stackrel{?}{=} c \pmod{p}$$

holds with equality. If the equality holds, then the signature is accepted. Otherwise, reject.

– *Confirmation and Disavowal.* The nominee V can confirm or disavow a signature $\sigma = (b, c, s)$ via proving the equality/non-equality of discrete logarithm $\log_d c = \log_g(y_V)$ or $\log_d c \neq \log_g(y_V)$. We refer the reader to [5] for the detail on how this proof can be conducted interactively. The proof is done using the technique proposed in [8].

3 A Flaw of Huang and Wang's Nominative Signatures

In [5], the authors claim that their scheme satisfies:

– Only the nominee can verify the authenticity of the nominator's signature. Even the nominator *cannot* verify the signature.
– If required, *only* the nominee can prove to the third party that the signature was issued to him and is valid. Even the nominator *cannot* prove that the signature is valid.

However, as we will show in this section, the above arguments are false. In the scheme proposed in [5], the nominator (the signer) can verify the authenticity of the signature. Moreover, the nominator can *convert* this signature to a publicly verifiable signature, which can be verified by any third party. This argument is illustrated as follows.

Verification by the Nominator.
We note that the nominator knows the published signature $\sigma = (b, c, s)$, where $(g^s y_S^e b)^{x_V} \stackrel{?}{=} c \pmod p$, for $e = H(y_V||b||c||m)$. Moreover, the nominator also knows the following relations.

- $a = g^{R_1} \pmod p$.
- $c = y_V^{R_2} \pmod p$.
- $s = s' + R_2 - R_1 \pmod q$.

From the above knowledge, the nominator can compute

$$\begin{aligned} g^{R_2} &= g^s \cdot g^{R_1} \cdot g^{-s'} \pmod p \\ &= g^s \cdot a \cdot g^{-s'} \pmod p \end{aligned}$$

Obtaining g^{R_2}, he can verify whether the signature $\sigma = (b, c, s)$ is correct by testing whether

$$g^s y_S^e b \stackrel{?}{=} g^{R_2} \pmod p$$

holds with equality.

Correctness.
The correctness of the above verification test is due to the following equations.

$$\begin{aligned} (g^s y_S^e b)^{x_V} &\stackrel{?}{=} c \pmod p \\ &\stackrel{?}{=} y_V^{R_2} \pmod p \\ &\stackrel{?}{=} g^{x_V R_2} \pmod p \end{aligned}$$

which is equivalent to

$$g^s y_S^e b \stackrel{?}{=} g^{R_2} \pmod p$$

□

Theorem 1. *Huang and Wang's scheme allows the nominator to verify the published signature* without *collaborating with the nominee. This problem violates the definition of nominative signature schemes.*

Public Verification.
The nominator can *convert* the signature $\sigma = (b, c, s)$ to a publicly verifiable signature for any third party, by publishing the value of $\gamma = g^{R_2} \pmod p$, together with a signature of knowledge proof $SPKLOG\{\alpha : \gamma = g^\alpha\}(\sigma)$. We

note that by knowing γ and verifying the signature of knowledge proof, any third party can be convinced with the authenticity of the signature σ, γ, where $\sigma = (b, c, s)$ for a message m by testing whether

$$g^s y_S^e b \stackrel{?}{=} g^{R_2} \pmod{p}$$

holds with equality. □

Theorem 2. *Huang and Wang's scheme allows the nominator to convert the* nominative signature *to a publicly verifiable signature. This violates the requirement of nominative signature schemes.*

Corollary 1. *The nominative signature scheme proposed in [5] is* not *nominative.*

The proof can be derived from Theorems 1 and 2. The nominator (signer) can verify a signature by himself, which contradicts the requirements of nominative signature schemes. Moreover, public verification is possible to be achieved by publishing the value of g^{R_2}. □

4 Discussion

Nominative signature scheme requires that the nominator (signer) cannot *even* verify the signature that he created. The attack that was proposed in [5] against the proposed nominative signature scheme in [7] suggested that *all* information that was provided/generated by the nominator during the signature generation *will be available to him at the time of verification* (including the random numbers used). This information equips the nominator with all information that he requires together with the published signature, and the aim of the signature scheme is supposed to *stop* the nominator to verify the authenticity of the signature. As we have shown in Section 3, when the signer is equipped with this information, then the proposed scheme in [5] was also flawed against the same problem as the one that they raised in [5].

Therefore, we argue that this assumption is too strong. If the system parameters that were generated during signature generation are always available to the signer *after* the signature is published, then the signer will always be able to verify the published signature by himself. We note that in practice, the signer may have signed n messages (for a large number of n) during the system lifetime. Therefore, keeping all the parameters during the lifetime of the system requires an enormously large space. Hence, it is reasonable to assume that the system parameters generated during the signature generation will *not* be available for the signer. This way, both schemes in [7] and [5] will not suffer from the ability of the signer to verify the published signature.

5 Conclusion

In this paper, we showed that the nominative signature scheme proposed in [5] is *not* nominative since the nominator can also verify the authenticity of the signature. Moreover, we also showed that the nominator *can convert* the signature to a universally verifiable signature. We also discuss an assumption that should be placed to enable a secure nominative signature scheme to be constructed. When this assumption does not hold, then the search for a secure and efficient nominative signature scheme remains an open research problem.

Acknowledgement
The authors would like the anonymous referees of ACISP 2005 for their comments and suggestions to improve this paper.

References

1. J. Camenisch. Efficient and generalized group signatures. *Adv in Cryptology - Eurocrypt '97, LNCS 1233*, pages 465–479, 1997.
2. J. Camenisch. Group signature schemes and payment systems based on the discrete logarithm problem. *PhD thesis, ETH Zürich*, 1998.
3. D. Chaum. Zero-knowledge undeniable signatures. *Advances in Cryptology - Eurocrypt '90*, Springer-Verlang, Berlin, pages 458–464, 1990.
4. W. Diffie and M. Hellman. New directions in cryptography. *IEEE IT*, 22:644–654, 1976.
5. Z. Huang and Y. Wang. Convertible Nominative Signatures. *Proceedings of Information Security and Privacy, ACISP 2004, Lecture Notes in Computer Science 3108*, Springer-Verlang, Berlin, pages 348 – 357, 2004.
6. M. Jakobsson, K. Sako, and R. Impagliazzo. Designated Verifier Proofs and Their Applications. *Advances in Cryptology - Eurocrypt '96, Lecture Notes in Computer Science 1070*, Springer-Verlang, Berlin, pages 143 – 154, 1996.
7. S. J. Kim, S. J. Park, and D. H. Won. Zero-Knowledge Nominative Signatures. *Proceedings of PragoCrypt 96, International Conference on the Theory and Applications of Cryptology*, pages 380 – 392, 1996.
8. M. Michels and M. Stadler. Efficient Convertible Undeniable Signature Schemes. *Proc. of 4th Annual Workshop on Selected Areas in Cryptography (SAC 97)*, Springer-Verlang, Berlin, pages 231 – 244, 1997.

Who Goes There?
Internet Banking: A Matter of Risk and Reward

Adrian McCullagh and William Caelli

Information Security Institute
Queensland University of Technology
a.mccullagh@qut.edu.au & w.caelli@qut.edu.au

Abstract. There are now more than 7 million internet banking users in Australia. Despite this substantial uptake in Australia, Australian banks continue to concentrate their respective security efforts upon internal mechanisms. Education of bank customers has not for the most part solved the fundamental flaws existent in internet banking. It is widely accepted that the weakest link in internet banking facilities is not with the banks' internal mechanisms but with customer PC. This paper analyses the research opportunities available to improve internet banking in Australia, which research could be exported to other jurisdictions where internet banking is available.

1 Introduction

The purpose of this paper is to promote further discussion on the topic of "internet banking". This has become a major issue in relation to the acknowledged need for the development and deployment of practical cost effective solutions, including technical, legal, industry and policy aspects, to better securing "internet banking". This activity is now clearly significant to the national economy[1]. While internet based banking facilities may exist at many levels in the banking and finance industry, the main focus of this paper is the domain of "retail banking", that is generally regarded as that set of transactions entered into by individuals and small to medium sized enterprises (SMEs) concerning the relationship they have with their own banking services partner. Moreover, to enable depth of analysis, this paper limits its discussion to holders of personal bank accounts or the like rather than business accounts. The paper makes reference to a number of relevant court cases in this area in the USA as there has been little case law in Australia concerning the issues raised in this paper. These cases are referred to as the authors believe that this case law could be very persuasive to the development of Australian jurisprudence in this area.

Of late, there has been a substantial amount of unflattering journalism regarding the security and assurance of internet based banking, particularly in a home and small business environment[2]. This publicity has primarily concerned the insecure framework, which bank customers are required to endure in order to take advantage of this relatively new form of convenient, online banking activity. At the forefront of this journalism has been the fact that even though banks have expended substantial funds on securing their own internal financial systems, the financial payments

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 336-357, 2005.

systems utilised by their merchant customers including so-called "EFTPOS" or "Electronic Funds Transfer at Point of Sale" systems, inter-bank financial exchange and correspondent banking functions, and other dedicated banking/finance industry systems, serious security vulnerabilities may be seen at present as being concentrated in the associated "client" systems, services and users rather than with the "back-end" banking systems and the banks themselves. However, banks are now themselves the subject of notable internal security breaches and this is exemplified in the recent Sumitomo Mitsui Bank incident in London as report in the press is a prime example[3]. This type of sophisticated attack was previously only directed at internet banking customers[4].

These attacks involve the use of "trojan horse" programs and so-called "spyware", which may implement and use such illicit processes as key capturing, mouse movement tracking, and screen logging, and interception of "smart-card" data flow connection to fraudulent internet sites, manipulation of main memory and stored data/program files, etc. In response to these attacks the finance sector has in some jurisdictions announced that they will move to "new" technology, such as "two factor authentication", i.e. requiring the client user to present two forms of identity claims involving entry of account code and password, known as a "what you know" claim, and use of a physical token in their possession for some purpose, known as a "what you possess" identity authentication process. However, as one noted and respected cryptographer and social technology scientist has announced, such two factor authentication may have been a solution "10 years ago" but it does not address the current style of attacks on client systems that the finance sector currently has to address[5].

Indeed, both of these identity claim and verification processes assume that a complete "trusted path" exists between the "claimant" and the "verifier", a situation that manifestly does not exist with use of commodity level personal computer systems.

In this paper, the following topics will be discussed:

- a short history of the development of internet banking and the factors that influenced the move by financial institutions to this environment;
- some of the more prevalent attacks that exist, the ICT vulnerabilities that these attacks exploit and how banks have attempted to address these attacks;
- what the banking and finance sector can expect if the current technology framework is not substantially re-engineered so as to better manage the possible systemic risk arising from the next generation of vulnerabilities, threats and resulting attacks on these critical systems;
- the cultural reticence by many in the banking sector to address the present catastrophic position regarding safe internet banking;
- the failure of self–regulatory regimes on the safety of internet banking and its possible effects upon society and national security;
- whether legislative and regulatory frameworks are a possible answer to reach an acceptable level of security for the banking environment, and
- some possible solutions for investigation

2 Historical Perspective to Internet Banking

Henry Ford is alleged to have stated that "It is well that the people of the nation (the USA in this case) do not understand our banking and monetary system, for if they did, I believe there would be a revolution before tomorrow"[6]. This statement was made some 75 years ago and it is clear that it probably holds truer today than when it was made. With the widespread adoption of both the internet and its associated "world-wide-web (WWW)" structures, the global economy has advanced to an unprecedented level by creating an environment whereby the consumer can transact business on national and international scales with relative ease. Convenience has taken almost total precedence over security and this is especially so for the banking sector[7]. The reward factors are being pushed by merchants and the banking sector to unprecedented levels without the balance of the risk being adequately elucidated to the consumer. The radical assumption made by the banking and finance industry, and others, is that the commodity level personal computer, both hardware and software, and its associated internet connection structures are secure and suitable for this purpose without major and costly adaptation. The mistaken assumption is that the PC was designed, developed, sold, supported and deployed for use for such security sensitive transactions

At the centre of this consumer driven global economy is the banking industry with its payment system mechanisms. The banking industry is highly competitive and thus effective cost management is not a luxury but a necessity.[8] De Young has noted that in the US, branch banking costs the banking enterprise about $1.07 per transaction, telephone banking costs about $0.55 per transaction, ATM banking costs about $0.27 per transaction and Internet banking costs about $0.01 per transaction[9]. Similar figures have been noted in Australia and therefore there is, in this highly competitive environment, a substantial push for an increased uptake in internet banking by consumers. It has been argued that the "health" of the global economy is now highly dependent upon this uptake not stalling or slowing down and even that any loss of confidence by consumers in this new commercial environment could have a catastrophic affect upon the global economy. The banking industry is particularly concerned that, if there was a substantial loss of confidence in electronic business and in particular in electronic banking, then this could result in systemic failure in electronic banking as a whole. The banking industry, in practice, has re-engineered their business model substantially around the use of internet banking so as to drive transaction costs down and to, thus, maximise profits and return to shareholders.

The movement of the banking sector to an emphasis in retail level activity to transacting business via the internet is relatively new and currently remains relatively immature. In many respects the internet as a delivery channel for business transactions is still evolving. The first bank to move to an online delivery of its products and services was the Presidential Bank of Maryland in October 1995[10]. Shortly thereafter in 1996 the Wells Fargo bank followed suit[11]. Since then there has been a global proliferation of banks providing their products via the internet. In Australia, the National Australia Bank, in 1999, was the first bank to offer and deliver its product set via the internet[12]. Presently every bank in Australia provides its product set to its retail clients via the internet. As noted by De Young the rewards in

providing this delivery mechanism have been substantial. It has permitted the banks to substantially drive down internal costs and permitted them to obtain a new form of revenue through the various service charges imposed upon their respective clients. However, it should be noted that electronic delivery of other banking and financial services is not new as such elements as "telegraphic transfer" of funds; interbank financial transactions and the like have occurred since the inception of telecommunications services.

This new delivery mechanism has not been derived without a cost. Where there is money then so to follows criminal activity. The development of the ARPANET/DARPNET of the 1970s into the global internet of today has permitted criminals to use it as a new mechanism for illegal gains. Most recently, criminal activity has concentrated upon the weakest link in the internet banking environment namely; the computer systems of a bank's retail customers.

3 Industry Statements – Fitness for Purpose

Indeed, the manufacturers and suppliers of commodity level, home PCs have warned about the use of their products and systems for secure transaction purposes for some years, purposes for which they were never designed or developed. For example, the following statement has been made in a text published by Microsoft Corporation in 2003 in relation to one set of its systems software products commonly used by households:

> *".. if the data being secured is high-risk (such as medical data), use Windows 95, Windows 98, Windows ME, or Windows CE only if you get a key from a user or an external source to encrypt and decrypt the data. .. These platforms cannot be used in secure environments."*[13]

This book, moreover, has the following statement on its cover:

> "*Required reading at Microsoft" – Bill Gates*

At the same time, Microsoft products such as Windows'2000, XP, etc. as well as other operating system software used on home and small business PCs suffer from the problem that such elements as "device drivers" may be installed in "kernel mode" which enables them to bypass all security mechanisms in the system. This has been acknowledged with the research and development project at Microsoft Corporation, entitled "Next Generation Secure Computing Base (NGSCB)" which may essentially be seen as the incorporation of a secure "PINpad" onto the motherboard of a modern PC with associated systems software support.[14] At the industry level the formation of the "trusted Computing Platform Alliance (TCPA)" in 1999, now the "Trusted Computing Group (TCG)" , again confirmed the need to rethink the fundamental design of the PC to create a "trusted computing platform" through the addition of a low-cost "trusted platform module (TPM)". Pearson, S et al[15],in their book detailing the specifications clearly summarise the problem with securing the home Pc for use in secure transaction applications as follows:

> *".. the ability to protect a PC or other computing platform through software alone has developed as far as it can and has inherent weaknesses…..experts in information security conclude that some security problems cannot be solved by software alone, for example, trusted hardware is needed as the basis for software security mechanisms…."*

Essentially, manufacturers and suppliers of the necessary PC systems have constantly and consistently claimed that the products are not suitable for secure transaction usage without notable additions and enhancements to the products, mainly at the hardware level. The claim is repeatedly made that the PC was created for exactly that purpose, a "personal" computer, operating in an isolated and individually controlled environment. The demand and thus the market for security mechanisms and sub-systems in the PC simply did not exist when the fundamentals of these systems were created. Connection to the global Internet changed all that. In simple terms, manufacturers and suppliers of PC products and systems claim that it is not suitable, unaided with additional hardware and supporting systems software, for use in applications requiring sophisticated security and protection, such as home banking, medical record maintenance and distribution, and so on. The very existence of the Microsoft NGSCB project and the TCPA/TCG specification set clearly attest to this fact.

This is an unusual position as the Information Technology manufacturers of this type of technology have consistently and clearly indicated that the technology is not fit for the purpose of transacting securely over the internet. On the other hand the banking industry has consistently encouraged consumers to move to internet bank based services. By analogy, this is akin to a taxi company using a vehicle for passenger services even though the manufacturer of the relevant vehicle has stated that the vehicle was not designed for such purposes. In this regard, the issue of "fitness for purpose" under the Trade Practices Act is a substantial research issue that needs further investigation.

4 Current Threat/Vulnerability Profile

As the consumer market has developed in the use of the internet so has the sophistication of the criminal attacks. Banking industry members have consistently advocated that internet banking is safe and in Australia the National Australia Bank has stated that the "National Internet Banking is secure, convenient and easy to use[16]". It is interesting in this case that the term "National Internet Banking" is not defined even though it is pivotal to the contractual framework that is offered by the National Australia Bank. National Australia Bank could argue that this reference to "National Internet Banking" only refers to its side of the transaction processing system and does not include the customer's computer system as it does not have any control over or offer any banking industry specific enhancement to the security mechanisms employed by its customers. At best it can only make recommendations and where contractually possible exclude certain liabilities that may arise from inadequate security measures taken by its customers or the total lack of any related security

mechanisms and services in the computer and data communications products and systems used by the customer. On the other hand by not defining the term it allows the bank to argue a position that best suits it at the time of a dispute. Sometimes, from a legal liability position, there can arise a certain amount of legal risk protection through use of obscurity in drafting contracts; especially from the perspective of the party who is most likely the party intending to ultimately rely upon such obscurity in any proceeding. This lack of definition and precision could, in actual fact, not work in the bank's favour. This issue is further explored later in this paper.

Principal criminal activity related to internet banking activity has for the most part concentrated upon the customer side of the process. The banks have for the most part expended substantial funds in "hardening" their own internal security measures as well as bank-to-bank and bank-to-merchant schemes.. However, customer side system weakness has resulted in criminals concentrating their activity upon that side of internet banking. At the same time, manufacturers and suppliers of the personal computer systems for home/retail banking purposes have clearly stated that security mechanisms urgently need to be enhanced to make these systems suitable for this purpose[17]. For the most part the criminal activity has taken the form of "phishing" scams and more recently trojan horse, "root kit" and spyware software that is deposited on internet banking customer's computer often accompanied with these "phishing" or like scams.

4.1 Phishing

Phishing scams have in the past involved a mass email using a emailing list to multiple persons who may or may not be associated with the relevant bank to which the scam relates. For example even though neither of the writers are customers of the Wells Fargo Bank, they have each received messages purporting to come from the security section of that bank, which indicates that the bank has undergone some form of security software upgrade or related activity. The message then states that the bank's customer should update respective access details. The scam message directs the recipient of the scam message to click on a preset link which looks substantially close to or is indeed the real URL of a real bank but unfortunately it is an internet site operated by the scam merchants. Of course, the false URL is designed as a repository for the collection of customer bank details including access details, which can later be used by the scam merchants to transfer funds from the victim's bank account to another bank account which is operated by the scam merchants. Alternatively, the fraudulent site may simply act as a "man-in-the-middle" site, transferring data to and from the real bank site while recording all activity or even inserting additional data traffic into the connection during a real bank session. This "piggy-backing" activity is a significant threat where the bank customer is connected by broadband data services since it enables criminal activity to actually progress while a legitimate transaction session is underway without the knowledge of the customer. Indeed, in this case the use of "dual factor" authentication by the customer will have no effect as the criminal activity occurs during a specific data session without the need for the criminal to record passwords or other data for later use, particularly where such passwords entered by the PC keyboard may change.

The banks in reply to this known criminal activity have expended substantial funds in publicising the activity but there are always some gullible persons who will activate such emails even though they may know better. However, the form of this type of activity has become very sophisticated. For example, on 11 April, the US online payment system known as "Paypal" was attacked through a phishing exercise[18]. Paypal customers were informed through an elaborate message that Paypal had upgraded its computer authentication mechanism to overcome the activity of phishing. That is, the criminal in this case were using the uncertainty of the phishing scam to entice Paypal customers to update their access details. The success of this social engineering approach implemented by the scam cannot be under-estimated.

4.2 Key Logger Software

Another criminal approach that has proliferated in recent times is the use of trojan horse and root-kit software designed to capture key strokes, screen displays, mouse movements, smart card data entered into the system by an appropriate reader unit, and the like. These systems are highly relevant in relation to home banking and the use of a simple password, personal identification number/code, or the like. In electronic funds transfer activities the security of such schemes is set out in an Australian standard, AS 2805.3-2000[19], entitled *"Electronic Funds Transfer - Requirements for Interfaces, Part 3: PIN Management and Security"*. Appendix A to this standard is unequivocal as to the responsibilities of the so-called "acquirer" of banking related transactions. This appendix clearly states as follows:

> *"In addition to ensuring that the equipment or services are not misused or exploited, the acquirer has to also receive, transmit, and/or verify the customer PIN, as well as protect the secrecy and accuracy of the PIN during this process."*

Moreover, the standard clearly states that a "PIN entry device shall be a secure cryptographic device." and that "during PIN entry at a terminal, protection becomes the responsibility of the card acceptor."

A home banking user is essentially entering a PIN to authenticate themselves and then to allow for ensuing transactions to occur. The principles of 2805.3 are highly relevant to the home PC situation and cannot be ignored by the banking industry, which substantially participated in the development these standards themselves. Supporting this view is the fact that many of the relevant consumer level internet banking contracts are entitled "electronic transactions" or similar wording, and often offer multiple entry types and cover not only transactions effected through an ATM or EFTPOS machine but also internet banking transactions.

There appears to have been one court case concerning the use of such key stroke loggers. The case, though, does not involve a bank client but does give some insight into the effect of the use of this type of software system. In the case of "United States v. Scarfo"[20], the court took particular interest in the key logging technology that was used by the FBI to capture the pass-phrase that Mr. Scarfo was using to activate his "PGP-private-key-ring" for secure email purposes.

The facts of this case are interesting as it does give some insight into the relative ease by which skilled artisans can deposit this type of software onto the computer systems of unsuspecting victims.

> *Acting pursuant to federal search warrants, the F.B.I. on January 15, 1999, entered Mr. Scarfo's business office, to search for evidence of an illegal gambling and loansharking operation. During their search of these offices, the F.B.I. came across a personal computer and attempted to access its various files. They were unable to gain entry to an encrypted file named "Factors." Suspecting the "Factors" file contained evidence of an illegal gambling and loansharking operation, the F.B.I. returned to the location and, pursuant to two search warrants, installed what is known as a "Key Logger System" ("KLS") on the computer and/or computer keyboard in order to decipher the pass-phrase to the encrypted file, thereby gaining entry to the file. The KLS records the keystrokes an individual enters on a personal computer's keyboard. The government utilized the KLS in order to "catch" Scarfo's pass-phrases to the encrypted file while he was entering them onto his keyboard. Scarfo's personal computer features a modem for communication over telephone lines and he possesses an America Online account. The F.B.I. obtained the pass-phrase to the "Factors" file and retrieved what is alleged to be incriminating evidence"*

According to the affidavit of Randall Murcha, a Supervisory Special Agent with the FBI, who has a Ph.D. with extensive engineering experience and was at the time Deputy Assistant Director of the FBI's Laboratory Division[21]:

> *"The challenge for the FBI in this situation was to devise a technical search capability, which could search for and record key or key related information entered through as least one of these mechanisms [the key board in this case] without detection. ...in this case a "keystroke capture" component that was designed to record, under certain conditions.., each keystroke typed on the key board. This component was imbedded into Scarfo's computer in such a way as to conceal its very existence amidst other pre-existing elements of the computer".*

It is interesting that this case was first reported in December 2001, and since that time there has been a noted increase in this type of attack; especially upon internet banking customers. Currently, before the courts in the USA (Florida State Court[22]) there is the first legal case involving the allegation that internet banking is not safe because trojan horse key logger software captured certain bank details.

AHLO Inc is a customer of Bank of America (BOA) and uses the BOA internet banking facility to transact business. On 6 April, 2004, the CEO of AHLO had been expecting an electronic deposit into the AHLO account and noticed, whilst connected to the BOA internet banking site that there had been a US$90,000 transfer from the company's account to an account with the Parex Bank in Riga, Latvia. Once aware of this event, it is alleged in the filed court documents, BOA was immediately advised of this fact by the CEO of AHLO but the bank did not react for some 19 hours. After the elapse of this time it is alleged by AHLO that BOA was able to stop the further

dispensing of the funds from the Parex Bank but not after some US$20,000 had been withdrawn from the Parex Bank. The remaining US$70,000 according to all reports remains with the Parex Bank, and AHLO has been unable to retrieve it. Hence, AHLO has commenced proceedings against BOA for the following causes of action:

1. Breach of Contract and breach of the implied covenant of good faith and fair dealing; and
2. Breach of Fiduciary Duty;
3. Negligence; and
4. Fraud and deceit – intentional misrepresentation.

These causes of action, except for that of "breach of fair dealing", are generally recognised in Australian Jurisprudence, though they may not be accepted by the courts as being available in the bank/customer relationship.

The basis for the breach of contract cause of action is that there is an implied term of good faith in BOA's contract with its clients. By failing to recall promptly the fraudulently transferred funds, BOA had not acted in good faith and therefore should be liable for the lost funds. It is highly likely that a court could accept this position because of the alleged delay by the bank in responding to the notice from the CEO of AHLO. In Australia, in addition to the general contractual provisions that each bank contracts with their respective customers, the EFT Code of Conduct also sets out additional rules and procedures to govern the relationship between users and deposit-taking institutions (this includes all banks as well as all building societies and credit unions) that provide electronic funds transfers including electronic access to customer accounts[23]. That is, it governs internet banking and the relationship between banks and their customers who utilise internet banking sites operated by Australian banks. Section 5 of the EFT code of Conduct covers the "liability for unauthorised transactions". An unauthorised transaction is any transaction not authorised by the user. Clause 5.1 provides that:

> *"No account holder liability in respect of any fraudulent or negligent conduct of account institutions' employees, or agents,* ***forged****, faulty, expired or cancelled* ***access method;*** *losses occurring prior to receipt of access method; or incorrect double debit transactions".* (emphasis added)

The top six banks operating in Australia[24] clearly state in their respective product disclosure statements that they will each respectively comply with the EFT Code of Conduct. The issue is: what does the term "forged" mean in the context of a key logger that captures the access codes of an unsuspecting bank customer? The term forged or forging is defined similarly in each of the Australian criminal Acts and Codes as follows:

> *"Forging" means the counterfeiting, or altering in any particular, by whatsoever means effected, with intent to defraud, of an instrument, or document, or of some signature, or other matter, or thing, or of any attestation, or signature of a witness, whether by law required or not to any instrument, document, or matter, the forging of which is punishable under this Act."*

It is clear that the term "forged" would be interpreted by a court in such a case as meaning that when a criminal has used a key logger to obtain access codes without authority and then uses them for his/her own fraudulent benefit then the use of such access codes is a forgery. It is highly likely that the courts would take a policy position in making such an interpretation as was the case in Kennison v. Daire[25]. The question regarding the potential for insertion of fraudulent transactions by a third party during an authenticated session, as previously discussed, also needs to be examined.

The next cause of action involves a claim of breach of fiduciary duty. It has been accepted in Australia for a substantial period that the relationship between the bank and its customers is primarily that of debtor/creditor and not a fiduciary relationship; though in some cases a bank does have such a relationship and therefore the duty that follows therefrom. In the case of Foley v. Hill[26] the plaintiff had deposited a sum of money with its bank many years before bringing its action. The bank had agreed to pay 3% interest but failed to pay such interest for 6 years, which period exceeded the statute of limitations at the time. The bank argued that the bank/customer relationship was simply a creditor/debtor relationship and therefore the statute of limitation would obviate the action brought by the customer. Lord Cottenham at p1005 stated"

> *Money paid into a bank is money known to the principal to be placed there for the purpose of being under the control of the banker; it is then the banker's money; he is known to deal with it as his own; he makes what he can, which profit he retains to himself... He has contracted, having received that money, to repay to the principal when demanded a sum equivalent to the paid into his hands.*

This does not mean that the above facts would not result in the courts in Australia holding that in this case a bank is acting in breach of its fiduciary duty though it is doubtful that such would be the case.

The next cause of action centres about whether the bank owned a duty of care, and whether the bank has breached such duty of care in not reacting more quickly than it did? The issue of causation regarding the damage that resulted in this USA case from a delay on the bank's part is more problematical as even if the bank had reacted more diligently this may not have altered the resulting damage. This issue will revolve around timing and whether, even if the bank had reacted more quickly, this would have altered the position the AHLO currently finds itself in. The delay, if proved by AHLO, would certainly amount to an excessive delay and negligence could readily be inferred by the courts but it may not be enough for AHLO.

The final cause of action raised by AHLO is probably the more difficult for AHLO to prove but, if it does so, it is also the most difficult for the banks to rebut. In Australia, a customer would need to establish that the banks have intentionally misrepresented that internet banking is secure and safe to use. The Banks in Australia, though encouraging customers to use internet banking, and through branch closure activities, establishment of specific "internet" accounts and the like have made such activity more or less imperative for some communities. However at the same time it could not be said that the banks have comprehensively disclosed the risks involved. Each of the top six banks operating in Australia have extensive material concerning "safe" internet banking. This extensive material is not easy to

comprehend by inexpert home users or the general public largely ignorant in matters of information and communications technologies. In some cases the customer is given far too much information which makes the material incomprehensible. Further, in some cases the material is not easy to find even though the product disclosure statements refer to the material as if the customer is fully conversant with the security obligations placed upon them. It is arguable that a customer in Australia may be able to succeed in this cause of action because, even though the material is available, it is incomprehensible or obscure or difficult to find.

5 Cultural Reticence to Establish Secure Internet Banking

The banks have made substantial savings in moving their consumer and business operations to the internet environment but this has also resulted in a new risk profile being created. The banks have promoted their respective internal banking systems as being highly secure and, on the basis of extensive experience over many years, they probably are. But as has been stated in this discussion, criminals are principally concentrating on the customer side of internet banking.

With the increased use of phishing, trojan horse key logging, root kit sub-systems and like technology, what can the banks do and at what cost? The Australian Federal Government has designated the banking and finance sector as being critical to the well being of the nation and thus a major "critical infrastructure"[27]. Most western nations have also designated their respective banking industries as being part of such critical infrastructure. Being critical infrastructure which vitally supports the Australian economy, it is imperative that action be taken. If there was a substantial loss of confidence in internet banking in Australia then it has been argued that the economy could even falter through lack of confidence in the nation's payments system and the productivity gains achieved in recent time could be drastically set back.

Since the weakest point in internet banking is on the customer side then the banks need to devote more resources to this aspect in conjunction with relevant manufactures and suppliers of the relevant equipment and systems. To date the banks have concentrated their efforts on promoting safety and security through the use of "soft" options. These soft options include attempts at educating their respective customer base and the promotion of certain software based security technology which usually includes some intrusion detection technology, a virus scanner, spyware detector and firewall technology. It is clear, however, that the banks cannot dictate any particular vendor's technology to their customers as this could amount to a contravention of section 47 of the Trade Practices Act concerning third line forcing. However, overall the banks are claiming that the home PC is a suitable vehicle, unmodified and without enhancement, for the purposes they claim, i.e. trustworthy and safe home banking operations. It is apparent that the manufacturers of these systems make no such claims and, indeed, over recent years have clearly distanced themselves from any such claim [28]

There is a major logistical difficulty with educating the general populace as all of the relevant information first has to broken down to its simplest, understandable form, if that is even possible at present, and not everyone will necessarily be reached. Multiple channels need to be used and this increases the cost substantially. It is also a scatter gun approach as it is difficulty for the target audience to be specifically identified. Further, the implementation, configuration, maintenance and support of many of the security technologies proposed for use by the home user is relatively expansive and requires some sophisticated knowledge of ICT. For example, even though a firewall sub-system may be installed on the home PC it may not be properly configured or optimised so as to protect the customer from nefarious activity such as the introduction of a trojan horse key logger, memory mapper, etc. Also most if not all virus systems and for that matter many firewalls do not adequately deal with software that has been voluntarily downloaded by the customer and which surreptitiously includes key logger code embedded in it. In some cases, operating system software in use may not even possess rudimentary access controls with the result that such elements as spyware can obtain complete control of the home user's PC. Even if later software systems are used that incorporate basic access control mechanisms, the principal of "least privilege" is not followed and any software obtained by a user, knowingly or unknowingly, will "inherit" all the privileges of that user.

Also, as has been recently publicised in the Australian press, the National Australia Bank has reimbursed one of its customers who was adversely affected by a trojan key logger when he accessed his bank account when using a computer located at an internet café. It appears that the computers situated at the café were infected with key logger programs and as such collected the customer's bank access details for on forwarding. This activity has also been publicised overseas and in Japan two people were convicted for embedding key logger technology on a number of computers also located at a local internet café.

There are a number of approaches that the banks could take to better manage the rapidly changing risks involved in internet banking. It is not out of the question that if the incidence of failure in internet banking continues to increase then the regulatory authorities could drastically change the consumer banking landscape by setting down more stringent rules of operation or legislation could be enacted that would govern internet banking in Australia. Legislation would apparently be an action of last resort in Australia as the current Federal Government has clearly embarked on the recommendations set out in the Self Regulation Task Force Report.[29]

6 Managing the Risk – Protecting Bank Customers

According to the annual returns of the top 6 banks they each reported substantial profits for the 2004 bank year (see Table 1).

Table 1

Name of Bank	Net Profit after tax
Commonwealth Bank Limited	$2,572 million[30]
Westpac Banking Corporation	$2,539 million[31]
The National Australia Group Limited	$2,807 million[32]
The ANZ Banking Group Limited	$2,815 million[33]
St George Bank Limited	$717 million[34]
SuncorpMetway Bank Limited	$318 million[35]

Therefore the total cumulative Net Profit after tax for the top 6 banks in Australia amounted to approximately $11 billion.

According to a recent press article the up-take of online banking in Australia grew at it fastest quarterly pace in two years in the three months to December 31 2004. The article which is the result of some research sponsors by a number of Australian banks claimed that there are now more than 7 million online bank users[36], which is a clear indication that the online banking is not only a substantial aspect of banking per se but it is entrenched in the critical infrastructure of the Australian economy. If there was systemic failure in internet banking such that resulted in a total lack of confidence in this facility then this could have a substantial adverse affect upon the Australian economy. The banking sector obviously understands this situation and as a whole is taking this possibility seriously but more needs to be done. Research projects need to be established that can assist the banking sector in better securing this vital aspect of their business operations. It is clear that the activities to date are either too little or as one noted commented has stated too late[37]. Some of the solutions that have been publicised recently by the banking sector such as two factor authentication[38] should have been implemented some ten years ago and the new attacks that online banking customers are being confronted with are far more sophisticated than the solutions being proposed.

The above numbers are not insignificant and yet the banks appear reticent to raise the bar in providing the safe and secure facility for their customers to take advantage of internet banking. The same may be alleged for PC system manufacturers and suppliers themselves. It is always open for the Banking industry to work more closely with PC manufacturers to ensure that their systems are capable of being used to securely undertake internet banking transactions. This may require a smartcard mechanism to be included in the standard offering of consumer PCs. The proliferation of untrusted PCs and their use in undertaking internet banking transactions has allowed the criminal element of society to take advantage of this; much to the disadvantage of both the bank sector and their customers.

It is suggested that the banks could undertake a number of actions that could substantially risk manage the exposure that currently exists such as:

1. Implementing Mutual Authentication Procedures.

Currently the banks attempt to authenticate the customer to ensure that the person claimed is indeed that person using the computer at the relevant time. But the customer is not in a position to authenticate the bank. If the customer could securely authenticate the bank site then this could reduce the successful incidence of the phishing scam and "middle" site attack. In the phishing scam the bank customer is directed to a web page that is identical in form to that of the proper bank's site or, as explained above, to an intermediary site. The customer then inputs their bank details and in some cases the response from the surrogate or false site is that "the bank is experiencing difficulties. please try again later". Once the information has been collected the criminal can then uses the information to effect fraudulent activity on the customers account. Mutual authentication if properly implemented could vastly reduce the incidence of successful phishing activity but it must cater for the acknowledged "man-in-the-middle" attack which is more relevant in broadband connections. The criminals should not be in a position whereby a customer could authenticate the false site as an authentic site. The phishing scam success relies upon the look and feel of the criminal site to emulate the proper bank site or the ability for the illicit action to insert the "middle" site in between the customer and the web site of the customer's bank. More needs to be undertaken to authenticate the bank sites and thereby prevent the success of this activity. However, any form of authentication scheme critically depends upon a complete "trusted path" between the claimant, in this case the home banking user, and the verifier, which would be the bank's server site.

2. Client Side Security:

The principal problem with internet banking is that customers use acknowledged untrusted systems in gaining access to the bank internet facilities. Trojan horse key loggers can, as has been identified in the Scarfo case, be lurking on a customer's own computer collecting relevant information which can later be used for nefarious activity.

This matter has been again acknowledged by Microsoft Corporation in relation to its operating systems offerings and its NGSCB program[39]. The home website for the project clearly identifies this problem and states that the NGSCB project will provide *"the ability to protect data with a secure pathway from the keyboard through the computer"*. If the Banks could efficiently and cost effectively deploy across their respective customer base a smart device mechanism that does not rely upon the key board and which has been designed to prevent the surreptitious lodgement of trojan horse key logger technology then safe internet banking could be achieved, as outlined in Microsoft Corporation's "Next Generation Secure Computing Base (NGSCB)" proposals. The logistics of such an exercise is substantial and is worthy of research as it would benefit the economy as a whole by better securing a vital portion of Australia's critical infrastructure.

If the Banks could efficiently and cost effectively deploy across their respective customer base a smart device mechanism that does not rely upon the key board and which has been designed to prevent the surreptitious lodgement of trojan horse key logger technology then safe internet banking could achieved. The cost has been noted by the banks in Australia as being a major impediment to the deployment of a smart device for use by bank customer. The task would obviously not be without some difficulties but if properly research a logistical solution could be developed.

For example, if it can be assumed that the cost of a smart device can be obtained for no-more than AUS$35 each and the cost of deployment could be contained to no-more than AUS$165 per deployment then the total cost of deployment could be contained to AUS$200 per customer. The deployment rate could be over a 12 month period and therefore spread over two financial years. Hence, in any financial year the cost to the banks in aggregate would be approximately AUS$700 million, which amounts to less than 10% of the after tax profit for the top 6 banks operating in Australia. Of course, there would be some ongoing costs in maintaining such a facility but this surely could be absorbed in the current costs and expenses that the banks currently encounter. This position assumes that the banks do not recover the costs from their customers over a certain timeframe. For example, if these costs were amortised over a three year period and directly met by the customer then the charge per annum would be commensurate with the current credit card charge to customers.

The task of logistically rolling our more than 7 million smart devices would not as stated above be an easy task and so the bank's reticence could be attributed to two major factors:

a. The banks do not want there to be an undermining of the currently perceived trust existing in the presently deployed system by those customers who have to wait for their new smart device to arrive and be adequately installed. This could also have a follow on adverse affect upon the branding value of internet banking, which needs to be protected;

b. The logistical issues of planning an efficient rollout is not, as can be expected, an easy task. The banks, due to the possible issue in (a), want the time frame for the completion of the roll-out to be as short as possible. When ever there is a contraction of time for the roll-out of any new process, there is an increase in the risk of some failure arising.

The figures stated in this solution are not exact and as such this solution should be investigated especially regard the type of technology that would be deployed and the logistics involved.

3. Client Activity Profiling:

The banks could deploy client transaction profile technology. This type of technology is currently immature but warrants further investigation as it

could create an early warning mechanism to identify abnormal transactions affecting a client's bank account. However, it does assume that again a trusted path exists between the user and server sites so that an intermediary system cannot intercept and control all channel activity.

4. **Suspect Recipient Bank Monitoring**:

   This would involve the banks having a watching brief on account activity that involves suspect jurisdictions; usually out of the way jurisdictions like Latvia or Lithuania or some tax haven jurisdiction. If a transaction does arise that is suspect then the banks could under take an out of bounds verification with the customer to ensure that the transaction is authentic and proper. Once again, the communications channel for such notification would itself have to be trusted.

5. **Education**:

   Even though this aspect has been criticised earlier in this discussion it is still a very important part of any bank's aim in better protecting its customers. Education is one part of a multi-pronged approach to better servicing customers and to better secure internet banking. However, it must be relevant to the background and experience of those customers. The education factor must take into account all strata within society and one solution does not necessarily fit all strata within society.

6. **Contracts:**

   Bank contracts are never easy to understand and online bank contracts are no exception. Many of the provisions are not contained in one document not is the language used that simple to comprehend. Further some of the contracts are simply impossible to satisfy. A standard form online contract should be researched and developed much like the ACIF contracts (though they too have their own peculiar difficulties), which could be used by bank customers when they become online bank users. From the customer's perspective this would have to be better than what is currently on offer. The next section delves more fully into this aspect.

7 Contractual Position: A Wolf in Sheep's Clothing

As has been discussed above, the principal relationship between a bank and its customers is that of "contract". However, this relationship is not established in a vacuum, as it is also regulated by the EFT code of Conduct and the Banking Code of Conduct. Each of these codes has their genesis in the self-regulatory framework that has been promoted by the Australian Federal Government since the year 2000.

The EFT Code of Conduct (EFTCC) is governed by the Australian Securities and Investment Commission (ASIC). Part A of the EFTCC applies to all EFT transactions, which are defined as "*funds transfers initiated by giving an instruction through electronic equipment and using an access method, to an account institution (directly or indirectly) to debit or credit an EFT account maintained by an account*

institution". Further, Part A of EFTCC does not apply to business accounts and therefore only applies to consumer accounts. The EFTCC applies to all account institutions that have adopted it and currently the top 6 major banks in Australia have adopted the position to be bound by the EFTCC[40]. Clause 22 of the EFTCC concerns the use of electronic communications between the account institution and its customers. One can understand the use of electronic communications based on its efficiency advantages except for the fact that it is also electronic communications that are used by criminals to carry out the phishing and allied scams discussed above. Clause 22 provides that unless prohibited by law, a user may agree that any information that the EFTCC requires to be given by a "subscriber" (the account institution that has adopted the EFTCC) can be give via electronic means to a user's device or electronic address nominated by the user.

Table 2 provides a breakdown of the provisions which the top 6 banks may rely upon concerning their respective customer's agreement to this type of communication. Curiously, the Westpac Bank under the clause headed *"Your email address*" in Part 1 of their terms and conditions states that: "*We (Westpac Bank) may use your email address to advise you of any enhancement or changes to Internet Banking which may alter our delivery of, or your ability to use, Internet Banking*". One has to question what a customer has to do when this type of provision forms part of their contract when there are other provisions at the Westpac Banking site that try to inform their customers about phishing scams in the following terms:

> *"Westpac is warning its customers to be aware of a recent Internet scam, which involves sending out hoax emails claiming to be from Westpac and other major Australian banks."*

This information is on top of another aspect of the Westpac's documentation which needs elucidating:

> *When you use Internet Banking, you can be confident that we employ the highest level of security to protect your accounts and personal information.*
>
> *Subject to investigation, we guarantee that you will not be personally liable for any unauthorised transactions on your Westpac accounts, provided that you:*
>
> - ***Were in no way responsible for the unauthorised transaction***
> - ***Did not contribute to the loss***
> - ***Complied with the Westpac Internet Banking terms and conditions***
>
> *If you have any security concerns contact the Westpac Internet Banking help desk on 1300 655 505. (author's emphasis)*

Since, Westpac, for example, has through their website publicised information about the phishing scam, if a customer did fall foul of the scam by responding to a fraudulent email then the question is one of has the customer also contributed to the

loss. From a customer's perspective this is very confusing. They have agreed to accept email communications with their bank and yet there are phishing scam emails being sent which on the surface appears to originate from, for example, Westpac but in actual fact originate from criminals who are trying to capture the account details of Westpac customers. Further, Westpac has also dealt with spyware in its terms and conditions in the following manner:

> *Not only can spyware compromise your online security but it can also contravene Westpac's Internet banking conditions of use.*

This is a curious claim by Westpac, as the criminals, who write spyware and allied software components, as noted in the Scarfo case, go out of their way to ensure that the spyware is not detected by the victim. This appears to the authors to amount to some form of unconscionable conduct on the part of Westpac as it attempts to sheet the liability squarely upon the customer, even where the customer may not be in a position to know that their PC has been contaminated. It comes back to whether a PC is a safe and secure instrument to undertake internet banking or any other commercial transaction, including electronic government, electronic healthcare and related sensitive transactions.

This position is not isolated to just Westpac, though it is the easiest to identify, as other banks have also included similar provisions dealing with internet transactions and security[41]. Each Bank has its own set of provisions, which make reference to the EFT code of Conduct and are each very difficult to comprehend.

8 Conclusion

This paper has identified serious flaws in the current technology being encouraged by the Australian Banking community to be used to perform retail level banking transactions via the global internet and consumer level, commodity PCs. These flaws are not insurmountable and can be overcome. However, this would require the banks to take seriously the current and developing security vulnerabilities, particularly if a move to "web service applications" is contemplated. The current contractual frameworks are for the most part incomprehensible, particularly to the normal banking customer, and urgent research needs to be undertaken.

The banks obviously receive substantial rewards in providing internet banking facilities through decrease in transaction costs and allied savings. In doing so, they have for the most part looked to their own internal systems security which obviously is important and will remain so, but they must now extend an equivalent security umbrella to their on-line customer base.

The current approach from the banks has been by way of attempts at consumer education but this does not really address the vulnerability that is inherent in the consumer PC. It was never designed to be used to undertake such sensitive commercial transactions. Its very name indicates that the PC was designed for "personal", non-networked usage. At the time of its advent it was not really envisaged that it would be connected to a global, unprotected network. Nonetheless, it

Table 2: Top 6 Banks Electronic Communications and security mechanism

Banks/Issue	Westpac	Commonwealth	National	ANZ	St George	Suncorp-Metway
Electronic Communications permitted	YES **Part 1 : "Communication with you" and "Your Email Address**	YES **CLAUSE 16**	YES **CLAUSE 48**	**Amendment to ANZ Electronic Banking Conditions - "Securemail" reference though this is appears to be different to other bank's approach**	YES **CLAUSE 3.8**	YES **CLAUSE 23.12**
Security Bank's End	YES **Security page at Westpac internet banking site**	YES **Security page at NetBank site**	YES **Security page at National Bank site**	YES **Security Page at ANZ web site**	YES **CLAUSE 33**	YES **Security Page at SuncorpMetway bank site**
Security Customer's End	YES **CLAUSES 7 and 8**	YES **CLAUSES 1, 2, 3, 4, and 5**	YES **CLAUSE 19**	YES **SECTION 8** "Unauthorised Transactions"	YES **CLAUSE 5** (General) **CLAUSE 25.** (Access Methods - Passwords) **CLAUSE 41.** (PINs and Cards)	YES **CLAUSES 23.6 and 23.7 and 23.10**
Liability	$150 **CLAUSES 10,11,12,and 13**	$150 **CLAUSE 11 in particular clause 11.4(c)**	$150 **CLAUSE 27**	$150 **SECTION** 8 **"Unauthorised Transactions"**	**Subject to specific conditions : $150 CLAUSE 42**	$150 **CLAUSE 23.8 and 23.9**
Variations	YES **CLAUSE 3.1**	YES **CLAUSE 10**	YES **CLAUSES 34, 35 and 36**	YES **SECTION 8 "Changes to Electronic Banking Conditions of Use"**	YES **CLAUSE 3.3 (table of page 8)**	YES **CLAUSE 1.10**

has evolved into the principal vehicle to gain access to the internet which is the world's largest unregulated network. Nor was it envisaged that the PC would be used to effect vital commercial activities in a real-time, on-line manner.

It is no use lamenting over the inadequacies that presently exist, as action needs to be undertaken now. Such indications have already been clearly given by the ICT industry itself, e.g. through the TCG specifications, etc. discussed in this paper. The banking and finance sector, being a principal beneficiary of the growing electronic commerce environment, needs to take some responsibility for better protecting their investment in internet banking by better protecting their customers.

As Thomas Jefferson stated *"I believe that banking institutions are more dangerous to out liberties than standing armies"*. However, the banking sector has a real opportunity to downplay this view by providing cost effective security solutions that are noted above. Research will need to continue so that a systemic failure in the national economy does not occur.

---

1 The Federal Government at is Critical Infrastructure Protection Web site identifies the finance sector as being within the scope of critical infrastructure protection. http://www.tisn.gov.au/

2 Financial Review, article dated

3 http://www.theregister.co.uk/2005/04/13/sumitomu_bank/

4 Sumitomo Mitsui Bank attack involved criminals who were able to penetrate the banks internal systems through the use of trojan horse spyware software which captured relevant access control passwords. The criminal according to all report were on the verge of transferring £220 million, which approximates to $527 million

5 Schneier, B., in his Cryptogram Newsletter dated 15 March and 15 April. The banks activity is too little and too late. In Schneiere's opinion two factor authentication will not obviate the new attack mechanisms being employed by internet savvy criminals.

6 Banking and Federal Reserve Quotes, http://www.freedomdomain.com/bankquot.html

7 Claessens,J., Dem., V., De Cock, D., Preneel, B., and Vandewalle, J., " On Security of Today's On-line Electronic Banking Systems" located at http://joris.claessens.ws/

8 Molchanov, P., "Virtual Money, Real Profits – The Development and Future of the Online Banking Industry"

9 De Young, R., Senior economist and economic advisor to the Federal Reserve Bank Chicago "Chicago Fed Latter – The Internet's Place in the Banking Industry", March 2001, No. 163

10 Presidential Bank Web Page" Online Banking Marketspace and Trends", located at http://www.presidential.com/pk_marketspace.htm

11 http://www.wellsfargo.com

12 personal communication with Mr. Tony Burke of the Australian Banker's Association.

13 Howard, M and LeBlanc, D, "Writing Secure Code, 2nd Edition", Microsoft Corporation 2003, ISBN 0-7356-1722-8

14 See URL http://www.microsoft.com/ngscb (Accessed at 23 April 2005).

15 Pearson, S et al "Trusted Computing Platforms: TCPA Technology in Context", Prentice-Hall, 2003, ISBN 0-13-009220-7

16 National Australia Bank Product Disclosure Statement: "National Internet Banking: Product Disclosure Statement 'including Terms and Conditions'" effective 7 February 2005. located at http://www.national.com.au

17 Trusted Computing Group located at https://www.trustedcomputinggroup.org/home
Next Generation Secure Computing Base located at:
http://www.microsoft.com/resources/ngscb/default.mspx

18 This event was brought to the attention of the authors by Steve Teppler who was kind enough to give us permission to include the email scam that was sent to him concerning Paypal

19 AS 2805.3-2000, Australian Standard "electronic funds transfer – Requirements for interfaces, Part 3: PIN management and security", Standards Australia, 2000.

20 United States of America v. Nicodermo S. Scarfo and Frank Paolercio, United States District Court of New Jersey, Criminal No. 00-404
http://www.epic.org/crypto/scarfo.html

21 Affidavit of Randall S. Murch, located at
http://www.epic.org/crypto/scarfo/murch_aff.pdf

22 AHLO Inc. v. Bank of America in the Circuit Court of the Eleventh Judicial Circuit in and for the Miami-Dade County, Florida, General Jurisdiction Division, CASE NO. 05-2538-CA27

23 The EFT code of conduct binds all authorised deposit taking institutions that provide electronic banking facilities and this includes all Australian Banks.

24 The Commonwealth Bank of Australia Limited
Westpac Banking Corporation
Australian and New Zealand Banking Group Limited
National Australia Bank Limited
St. George Bank Limited
Suncorpmetway Limited

25 Kennison v. Daire (1986) 160 CLR 129

26 (1884) 2 HL Cas 28; 9 ER 1002

27 *http://www.tisn.gov.au*

28 Next Generation Secure Computing Base located at:
http://www.microsoft.com/resources/ngscb/default.mspx

29 The 1999 Australian Federal Government Taskforce on "Industry Self Regulation", chaired by Professor Berna Collier and delivered to the Government in May 2000.

30 Annual Return for the Commonwealth Bank Limited for year ending September 2004

31 Annual Return for Westpac Corporation

32 Annual Return (Financial Details Statement) for National Australia Bank Limited

33 Annual Return for ANZ Banking Group Limited

34 Annual Return for St George Bank Limited

35 Annual Return for Suncorp Metway Limited but only limited to banking section.

36 Moullakis, J., "Online Banking users surge past 7m"
http://www.afr.com/premium/articles/2005/04/20/1113854258268.html

37 Schneier, B., "The Failure of Two Factor Authentication", Crypto-Gram News Letter, March 15 2005.

38 ABN Ambro Bank in Europe commenced using two factor authentication in the late 1980s but this was a most advance situation at the time which other members of the international banking community did not follow suit.

39 Op cit note 28

40 http://www.asic.gov.au

[41] Commonwealth Bank site
http://www.commbank.com.au/NetBank/Security/default.asp?b=Overview
and their PDS located at
http://www.commbank.com.au/NetBank/Intro/default.asp?b=Conditions

SuncorpMateway Bank site
http://www.suncorp.com.au/suncorp/security.html
and their PDS located at
http://www.suncorp.com.au/suncorp/legal/pds_download/banking.html

St George Bank site
http://www.stgeorge.com.au/int_bank/security/what_you.asp?orc=personal
and their PDS is located at:
http://www.stgeorge.com.au/resources/sgb/downloads/pds/terms/sgbterms_290305.pdf

Role Activation Management in Role Based Access Control

Richard W.C. Lui, Sherman S.M. Chow, Lucas C.K. Hui, and S.M. Yiu

Department of Computer Science,
The University of Hong Kong, Pokfulam, Hong Kong
{wclui, smchow, hui, smyiu}@cs.hku.hk

Abstract. Role Based Access Control (RBAC) [6] is a popular approach to specify and enforce security policies in organizations. In RBAC, users are not directly assigned permission but with the use of roles as the intermediary. Role activation is one important component in RBAC. A user may activate a subset of his/her assigned roles to exercise the associated permission. This paper proposes a number of ways in which the role activation constraints can be specified and enforced in the enterprise environment. Also, an access control model and an authorization process are proposed to support the specification and enforcement of dynamic separation of duty constraints in a decentralized manner.

Key words: Role Based Access Control, Security Management, Role Activation, Dynamic Separation of Duty

1 Introduction

1.1 Role Based Access Control (RBAC)

Role Based Access Control (RBAC) [6] is a popular approach to specify and enforce security policies in organizations. A role is a collection of permissions, which acts as an abstraction of the job duties in the organization. Users are assigned to roles based on their responsibilities and qualifications (U-R assignment). In addition, permissions are assigned to the corresponding roles (P-R assignment). A role hierarchy is mathematically a partial order, in which the senior roles inherit the permissions from the junior roles. In RBAC, a user may not get all the permissions of the roles assigned to him/her at the same time. In order to exercise the permission associated with a role, the user should activate the corresponding role in a certain "session". During role activation, dynamic constraints (such as separation of duty and time constraints) are checked to ensure they are not violated at the time of role activation. For instance, a user may not be allowed to activate the "cashier" and "cashier supervisor" role at the same time in a certain session due to the conflict of interest.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 358–369, 2005.

1.2 Scalability in RBAC

Scalability, which refers to the ease with which a system can adapt to and accommodate increasingly complex requirements without the need for substantial changes to system structure or application algorithms [3], is an important issue in access control. In enterprise RBAC systems, the number of users, roles and permissions can be in hundreds or thousands. Managing these users, roles and permission can be a tedious task. It has been proposed that RBAC itself [15] and organization structure can be used for performing role based administration [8]. Also, logical objects can be used as the target of permission to remove the technical aspects of permission management from the security administrator [10].

Despite the needs for managing users, roles, permissions, and their relationship, the importance of managing the process of role activation has been neglected in existing studies. In many existing RBAC systems, the role activation process is often performed in an ad-hoc and uncoordinated manner. For instance, Park et al. proposes the use of cookies [9] to support RBAC in the web-based environment. However, the role activation process is missing in this system. A user is immediately granted the permission of all the assigned roles without the need for role activation. A RBAC system to control access to the resources stored in web servers is proposed in [5]. In this system, the role activation should be performed explicitly by the user. However, the role activation is handled independently by each web server. As a result, it is difficult to enforce dynamic constraints (in particular dynamic separation of duty constraints [16]) in a consistent manner. OASIS [7] is a distributed architecture which supports the specification and enforcement of RBAC. The role entry function is similar to the role activation process. However, the architecture does not provide a way for the decentralized and coordinated management of the role activation process in the enterprise environment.

1.3 Contributions and Organization

In this paper, three different approaches are described to manage the role activation process (which includes the specification and enforcement of dynamic constraints) in an autonomous, centralized and decentralized manner respectively. Among the three approaches, the decentralized approach provides the most flexible way to specify dynamic constraints. Despite its flexibility, it cannot be supported by most existing RBAC systems. Therefore, an access control model and an authorization process will be proposed to support decentralized management of the role activation process.

The rest of the paper is organized as follows: The three approaches in which the role activation process can be performed will be discussed in Section 2. In Section 3, an access control model which supports the decentralized specification and enforcement of dynamic constraints will be described. In particular, the model will focus on handling the dynamic separation of duty requirements which are specified in the form of conflicting role pairs. Also, the corresponding RBAC process will be described. Finally, in Section 4, the summary and some future research directions will be discussed.

2 Role Activation Process

2.1 Basic Concepts

In this section, three approaches to manage role activation are described. In the discussion througtout the rest of the paper, a *resource* is an entity which either stores data, performs specific services, or represents a network equipment (e.g. a computer). A resource can be active such as human users, or inactive such as web servers. An active resource can be granted permission (a set of privileges) to access other resources.

A *domain* is an enterprise-level classification of the entities within an organization. It provides a convenient means to enumerate objects under common authority [3]. A domain can also be a group of resources (e.g. applications) with similar security requirements. Examples include departments, faculties and teams. Domains can be organized in a hierarchy. A domain (the sub-domain) may reside in another domain (the ancestor domain), where the sub-domain will be subjected to the policies of the ancestor domains. The top-level domain is a domain not included in any other domain.

A constraint imposes restrictions on the acceptable configuration of the different RBAC components. Examples of constraints include static constraints (e.g. cardinality of a role) and dynamic constraints (e.g. time constraint and dynamic separation of duty) [6]. In this paper, the dynamic constraints are specified in the form of *role activation policy* (RAP), which will be enforced in the process of role activation.

A *role activation service* (RAS) is a resource responsible for handling role activation requests from the users and determining if a role can be activated according to the RAP. One important task of the RAS is to enforce dynamic separation of duty. This can be done by maintaining a repository which keeps track of the activated role set (ARS) for each user. To authorize a user to activate a role, the RAS should consult the ARS to determine that conflicting roles will not be activated at the time of role activation.

2.2 Autonomous Management

Many existing applications (e.g. [5] and [11]) adopts the autonomous role activation approach, where the role activation process is handled independently by different resources. For each resource in the organization, an RAS will be deployed and the corresponding RAP will be defined (Figure 1). In order to access the resource, the user should authenticate to the RAS associated with the resource and perform role activation. The RAS determines if the user is assigned to a role and verifies that the activation of the role conforms to the RAP by consulting the ARS maintained by the RAS. The user may access the resource only if the required roles are activated.

In this approach, in order to access a number of resources, the user should repeat the role activation process. Since the ARS is maintained by each individual

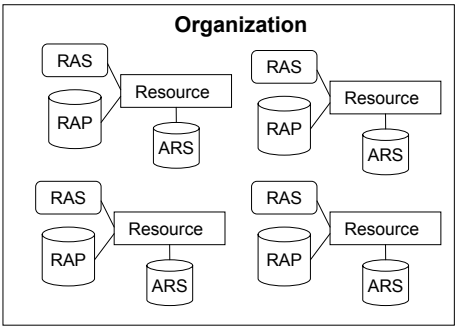


Fig. 1. Autonomous Approach to Role Activation Management

resource in an independent manner and the RAS in one resource has no knowledge of the ARS maintained by the other resources, consistent enforcement of the separation of duty constraints across the enterprise is difficult. For instance, a user may be able to activate conflicting roles in different resources at the same time.

2.3 Centralized Management

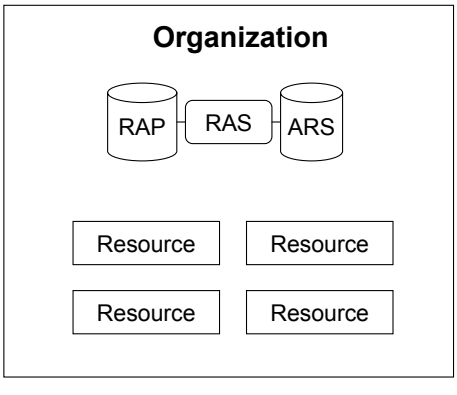


Fig. 2. Centralized Approach to Role Activation Management

Alternatively, the role activation and administration can be centralized (Figure 2). The security administrator of the organization specifies an RAP and it will be enforced across the organization. In order to access a RBAC secured resource, the user should first access the centralized RAP to activate the required roles. The user can only be granted access to the resource if the user has activated the required roles.

The advantage of this approach is that a user may activate a role and access the various resources across the enterprise. Also, as the ARS of the users in the organization is maintained in a centralized manner, the RAP can be enforced consistently across the different applications. However, in the enterprise environment, there may be many users and each user may be assigned multiple roles. Whenever a user intends to exercise the associated permission, the role should be activated (possibly a number of times in order to complete a given task). As there may be a large number of role activation requests in the organization, the RAS may become the bottleneck.

Also, it is difficult to specify the security policy for the whole organization in a centralized way. For complex organizations with many divisions (such as a bank where the divisions may be situated in different states or countries) , the various divisions may be subject to different laws and may have evolved their own specific practices. Although the same role may exist across an organization, the responsibility of that role and the ways in which the associated permission can be discharged may be quite different. For instance, the requirements for the RAP may be different in different branches of a bank and may be conflicting with each others. A role can be activated during the office hour in a branch, but the same role may only be allowed to activate during the non-office hour in another branch. A branch may also require that the role activation to be performed at a more secure server than the other branches. Therefore, it is not realistic to assume that a single RAS and RAP can be used to handle the diversity of security requirements in the enterprise environment.

2.4 Decentralized Management

The decentralized management approach is a generalization of the centralized management approach. In this approach, the RAP may be defined at multiple locations by the security administrators (domain administrators) of the different domains. For instance, the security administrators of the different regional branches of a bank may specify their own RAP. The resources (data and applications) will be stored in multiple servers, which reside in a certain domain/sub-domain. The P-R assignment will be handled by the security administrator (resource administrator) of the corresponding resource. For instance, the resource administrator of the human-resource department server and the marketing department server can independently determine what permissions are available to a role [1].

To access a resource in a certain domain, the user should activate the required roles (as defined in the P-R assignment) in one of the role activation servers (RAS) in the organization. The RAS of a given domain should enforce the RAP of that domain, as well as the RAP of the various ancestor domains. For instance, in order to access a resource in a regional branch, the user may activate the role at the RAS in the regional branch as well as the RAS in the headquarter. In this way, the enforcement of the role activation process can be decentralized to the various sub-domains, with the enterprise-level RAP being enforced at the same time.

If the role activation is authorized, the ARS for the user can be updated to include the newly activated role. Note that we restrict the RAS to handle only the RAP of the domain and the ancestor domains. The RAP of the sub-domains will not be enforced. It is because we do not assume that the RAS is able to enforce the security requirements of the sub-domains in a precise manner (e.g. the RAP of the sub-domain should only be enforced in a more secure RAS, which reside only in the sub-domain).

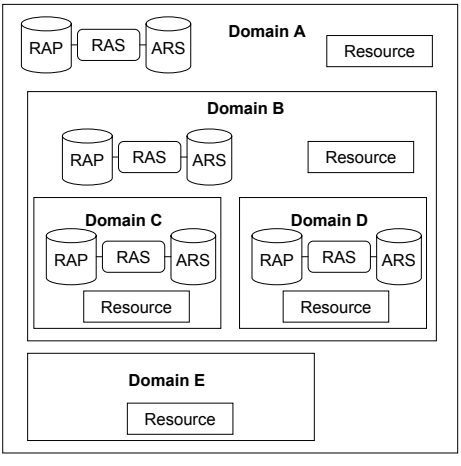


Fig. 3. Decentralized Approach to Role Activation Management

As an example, consider the scenario in Figure 3. In domain A, B, C, and D, an RAS (together with the associated RAP and ARS) is deployed. To access the resources in domain A, the user should activate the required roles at the RAS in that domain. However, in domain E, the RAS is not deployed. In order to access the resources in this domain, the user should activate the required roles at the RAS of an ancestor domain (domain A). Consider also the resources in domain D. In order to access the resources, the user may activate the required roles at the RAS in domain D. In this case, the RAP of both the domain D, B and A will be enforced. Alternatively, the user may also activate the required roles at the RAS in the ancestor domains (domain A and B). As an RAS does not enforce the RAP of the sub-domains, only the RAP of the domain B and A will be enforced if the role activation is performed at domain B. Similarly, if the role activation is performed at domain A, only the RAP of the domain A will be enforced.

In this approach, since multiple RAP may be defined in the organization, an RAP for a domain may conflict with the RAP defined in an ancestor domain. Therefore, policies should be defined to handle the potential conflict. Another important task to be handled by the RAS is to support the enforcement dynamic separation of duty constraints across multiple domains. For instance, if a certain

pair of roles (r_1, r_2) is defined to be in conflict in domain d, a user who has activated the role r_1 in d' (which is a sub-domain of d) should not be able to activate the role r_2 in d'' (which is another sub-domain of d). This requires the synchronization of the role activation information among the various RAS.

3 Authorization Process for Decentralized Management

Among the three approaches, the decentralized approach provides the most flexible way to specify dynamic constraints. It allows the specification of the role activation policy to be decentralized to the sub-domains. Being more familiar with the nature of permissions assigned to each role, the security administrators of the sub-domains may be in a better position to define the RAP in their own domains. Also, this approach provides a way for the RAP to be enforced in a flexible but coordinated manner. For instance, the RAS in an ancestor domain may provide a coarse grained enforcement of the RAP while the RAS in a sub-domain may enforce the RAP in a more precise manner. Since the RAS in a sub-domain will enforce the RAP of the ancestor domains, central control on broad policy can be performed. The model also provides fault tolerance. Even when some of the RAS are unavailable (e.g. under a denial of service attack), the user can still be able to activate the roles in another RAS in the organization.

In this section, an access control model which supports decentralized specification and enforcement of the dynamic constraints will be described. In particular, the model will focus on handling dynamic separation of duty requirements which are specified in the form of conflicting role pairs. After that, the corresponding RBAC process will be described.

3.1 An Access Control Model

The various entities in the system are defined as follows.

- A set of users U
- A set of objects OBS
- A set of operations OPS
- A set of permission $P = 2^{OBS \times OPS}$
- A set of roles R
- A set of policy domains D
- $RH \in R \times R$, a partial order on R called the role hierarchy
- $DH \in D \times D$, a partial order on D called the domain hierarchy. In this paper, we only focus on the case where the domains are arranged in a hierarchy.
- $UR \in U \times R$, a many to many user to role assignment
- $PR \in P \times R$, a many to many permission to role assignment
- $OD : O \rightarrow D$ a function mapping each object $o \in O$ to a domain $d \in D$

To simplify the specification, we will only consider dynamic separation of duty constraints in the form of conflicting role pairs [1]. We will not model

[1] Note that it is possible to generalize the constraints to a form where a user cannot activate n or more roles from a certain set at the same time, which is the form adopted by [12], to support more flexible policies.

the static constraints and the other types of dynamic constraints (like the time constraint) as they are not relevant for the discussion. In this section, we will denote r_1 and r_2 to be in conflict in a domain if a user cannot activate r_1 and r_2 at the same time in that domain. The set of conflicting role pairs in a domain $d \in D$ can be represented by

- $CR_d \subseteq 2^{R \times R}$, a set of conflicting role pair defined for domain d. If $(r_1, r_2) \in CR_d$, then the roles r_1 and r_2 cannot be activated at the same time in domain d.

A conflicting role pair is symmetric and so the associated roles will conflict with each others. Formally, $[\forall d \in D]((r_1, r_2) \in CR_d \Rightarrow (r_2, r_1) \in CR_d)$. Also, if a role r_1 conflicts with another role r_2 in a domain, the senior roles of r_1 will also conflict with all the senior roles of r_2 in that domain. Formally, $[\forall d \in D]((r_1, r_2) \in CR_d \Rightarrow [\forall r_1' > r_1, \forall r_2' > r_2]((r_1', r_2') \in CR_d)$.

In our model, we require the sub-domain to implement all the dynamic constraints of the ancestor domains. In particular, we require that if a role pair is conflict in a domain, the role pair should also be in conflict in the sub-domains. Formally, $[\forall d \in D]((r_1, r_2) \in CR_d \Rightarrow [\forall d' \in D : d' < d]((r_1, r_2) \in CR_{d'}))$

In addition, for each domain $d \in D$, we define the notions of activated role set and conflicting role set

- $ARS_d : U \rightarrow 2^R$ a function which maps a user to a set of roles activated in a domain d
- $CRS_d : U \rightarrow 2^R$ a function which maps a user to a set of roles which should not be activated in domain d.

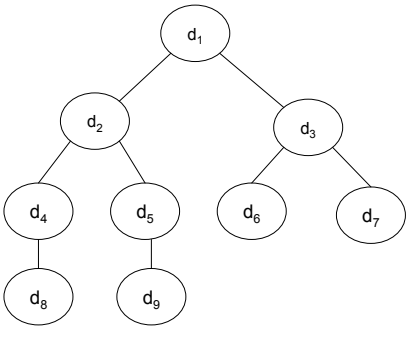


Fig. 4. Domains forming a hierarchy

The activated role set is used to keep track of the roles which are activated in a domain. The conflicting role set is introduced to enforce separation of duty constraints across the various domains in the domain hierarchy. As an example, consider the domain hierarchy in Figure 4. Suppose $r_1, r_2, r_3, r_4 \in R$ with $r_1 >$

r_2, $r_3 > r_4$ and (r_2, r_4) is a conflicting role pair defined in domain d_2. Consider the case where a user $u \in U$ activates the role r_1 in domain d_4. The activated role set $ARS_{d_4}(u) = \{r_1\}$. As r_2 is conflict with r_4 in domain d_2, r_1 (which is a senior role of r_2) is also conflict with r_4 and r_3 (which is a senior role of r_4). Since r_1 is activated in d_4 (which is a sub-domain of d_2), the roles r_3 and r_4 should not be activated in d_2 as well as the sub-domains of d_2 (which include d_4, d_5, d_8, and d_9). Therefore, $CRS_{d_i}(u) = \{r_3, r_4\}$ for $i = \{2, 4, 5, 8, 9\}$.

In order to activate a role, a user should be assigned the role or a senior role. Formally, a user $u \in U$ may activate a role $r \in R$ in a domain $d \in D$ only if $\exists r' > r : (u, r') \in UA$. In addition, r should not be included in the conflicting role set of domain d. Formally, $r \notin CRS_d(u)$. Also, the activation of the role should not violate the other types of dynamic constraints (e.g. time constraints).

A user may exercise the permissions which are associated with an object, if the user has activated the role (or a senior role) as defined in the P-R assignment in the immediate or any of the ancestor domains. Formally, a user may exercise a permission $p = (ops, o)$ where $op \in OPS$ and $o \in OBS$ if and only if $[\exists (p, r) \in PR, \exists r' \in R : r' \geq r]([\exists d' \in D : d' \geq OD(o)]r' \in ARS_{d'}(u))$.

3.2 The RBAC Process

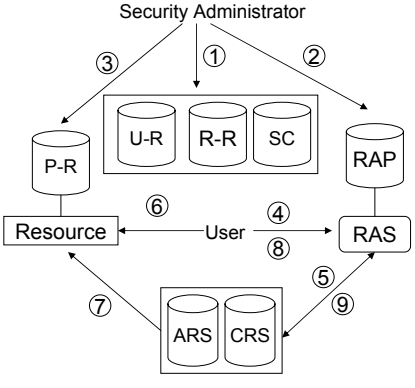


Fig. 5. The RBAC process

In this section, the access control process which adopts the decentralized approach to role activation management will be described (see Figure 5). The security policy for the organization should first be specified. The domain administrator of the top-level domain defines the roles, users and assigns the user to the roles (U-R assignment)(step 1). The domain administrator may cooperate with the other parties (e.g. the human resource department) to determine who may be assigned to a given role. To simplify the specification of policy, a role

hierarchy (R-R assignment) may also be defined. In the discussion, we assume that only a single role hierarchy is defined in the organization. The role hierarchy will be used for both permission usage and role activation [2]. In addition, static constraints (SC) may be defined (e.g. to constrain the number of users that may be assigned to a role).

The domain administrators of the top-level and sub-domains may define the role activation policy (RAP) for the corresponding domains (step 2), and configure the various role activation servers (RAS) to enforce the policy. To support more efficient role activation process, the conflict between the RAP of a domain and its sub-domains should be resolved in a way that the dynamic separation of duty requirements which are defined in the RAP of a sub-domain always implements the requirements as defined in the RAP of all the ancestor domains.

The resource administrator of a resource should also define the access policy. This is performed by defining the P-R assignment (step 3), which binds a permission to a role to be activated in order to perform the access. Each resource may maintain a P-R relationship. Alternatively, the P-R assignment may be shared by a number of resources and the access to them will be mediated by a common interface (e.g. a Java Servlet). In this way, there is no need for the P-R assignment to be defined separately in each resource in the organization and this simplifies the process of security management.

In order to access a resource (which requires a certain role r to be activated) in a certain domain d, the user should activate a role r' (which may be r or a senior role) at an RAS (step 4). The user may perform role activation in domain d' (which may be domain d or an ancestor domain). To perform role activation. the RAS should first check that the user is assigned to a role r'' (which may be r' or a senior role). The RAS should also check that the activation of the role will not violate the defined RAP. For instance, the RAS should check that the role to be activated is not included in the conflicting role set of the domain.

If the activation of role r' is authorized, the ARS of domain d' will be updated to keep track of the fact that the role is activated (step 5). The conflicting role set of the domain may be updated if required. At the same time, the role activation event will be sent to the other relevant domains for updating the corresponding conflicting role sets (CRS) if needed.

When the user requests access to the resource (step 6), the resource should verify whether the role r (or a senior role) is currently activated (step 7). Also, the resource should check that the role is activated in a trusted RAS. If so, the user can be granted authorization to access the resource.

To handle the deactivation of roles, the domain administrator can set a certain time-out such that the role is automatically deactivated after a certain time. Alternatively, the user may explicitly request for the deactivation of the role (step 8 and 9).

[2] An alternative is to make use of two separate hierarchies to support some application specific requirements [14]

4 Summary and Future Research Directions

In many of the existing RBAC applications, the role activation is handled independently in each application. As a result, it is difficult to handle separation of duty constraints across a number of applications. An alternative is to centralize the role activation management process. However, this approach suffers from the scalability problem. Also, it is difficult for a centrally defined security policy to meet the security requirements for all the applications in the whole organization. To provide flexibility in the specification and enforcement of the dynamic constraints, a model to decentralize the role activation process to the various sub-domains is described.

In this paper, we assume the separation of duty constraints are specified in the form of a pair of roles which conflict with each others. However, there may be other forms of dynamic separation of duty constraints [16,4,13,2]. Therefore, a possible future research direction is to explore the issues in supporting the decentralized specification and enforcement for the other forms of dynamic constraints. Also, we assume that the domains are organized in a hierarchical manner. However, there may be other possibility for the arrangement of the domains. For instance, the domains may form an acyclic graph. Also, a domain may be independent of any other domains in the organization (e.g. a temporary group formed to perform a specific task). The issues of role activation management for the other kinds of settings may also be studied. Finally, we may define some metrics to measure performance related to scalability.

Acknowledgement

This research is supported in part by the Areas of Excellence Scheme established under the University Grants Committee of the Hong Kong Special Administrative Region, China (Project No. AoE/E-01/99), two grants from the Research Grants Council of the HKSAR, China (Project No. HKU/7144/03E and HKU/7136/04E), and two grants from the Innovation and Technology Commission of the HKSAR, China (Project No. ITS/170/01 and UIM/145).

References

1. Venkata Bhamidipat and Ravi Sandhu. Push architectures for user-role assignment. In *National Information Systems Security Conference*, 2000.
2. Rafae Bhatti, James Joshi, Elisa Bertino, and Arif Ghafoor. X-GTRBAC admin: a decentralized administration model for enterprise wide access control. In *SACMAT '04: Proceedings of the ninth ACM symposium on Access control models and technologies*, pages 78–86. ACM Press, 2004.
3. Nicole Dunlop, Jadwiga Indulska, and Kerry Raymond. Dynamic policy model for large evolving enterprises. In *EDOC*, pages 193–, 2001.
4. Jan H. P. Eloff. Separation of duties for access control enforcement in workflow environments. *IBM Systems Journal 40(3): 666-682*, 2001.

5. David F. Ferraiolo, John F. Barklery, and D. Richard Kuhn. A role-based access control model and reference implementation within a corporate intranet. *ACM Transactions on Information and System Security, Vol. 2, No. 1, February 1999, Pages 34-64*, 1999.
6. David F. Ferraiolo, D. Richard Kuhn, and Ramaswamy Chandramouli. Role-based access control. *Boston : Artech House*, 2003.
7. John H. Hine, Walt Yao, Jean Bacon, and Ken Moody. An architecture for distributed oasis services. In *Middleware 2000, LNCS 1795, pp. 104-120*, 2000.
8. Sejong Oh and Ravi Sandhu. A model for role administration using organization structure. In *Proceedings of the 7th ACM Symposium on Access Control Models and Technologies (SACMAT 2002) (Monterey, Calif., June 3–4). ACM, New York, 155–162*, 2002.
9. Joon S. Park, Ravi S. Sandhu, and SreeLatha Ghanta. Rbac on the web by secure cookies. In *DBSec*, pages 49–62, 1999.
10. Najam Perwaiz. Structured management of role-permission relationships. In *ACM Workshop on Role Based Access Control archive Proceedings of the sixth ACM symposium on Access control models and technologies, Chantilly, Virginia, United States, Pages: 163 - 169*, 2001.
11. R. Sandhu R. Chandramouli. Role based access control features in commercial database management systems. In *21st National Information Systems Security Conference, October 6-9, 1998, Crystal City, Virginia*, 1998.
12. David Ferraiolo Ravi Sandhu and Richard Kuhn. The nist model for role-based access control: Towards a unified standard. In *ACM Workshop on Role-Based Access Control*, 2000.
13. R. Sandhu. Transaction control expressions for separation of duties. In *Proc. of the Fourth Computer Security Applications Conference, pp. 282–286*, 1998.
14. Ravi Sandhu. Role activation hierarchies. In *Symposium on Access Control Models and Technologies archive Proceedings of the third ACM workshop on Role-based access control, Fairfax, Virginia, United States, Pages: 33 - 40 , ISBN:1-58113-113-5*, 1998.
15. Ravi S. Sandhu and Qamar Munawer. The ARBAC99 model for administration of roles. In *Annual Computer Security Applicarions Conference*, pages 229–, 1999.
16. Richard T. Simon and Mary Ellen Zurko. Separation of duty in role-based environments. In *IEEE Computer Security Foundations Workshop*, pages 183–194, 1997.

VO-Sec: An Access Control Framework for Dynamic Virtual Organization*

Hai Jin, Weizhong Qiang, Xuanhua Shi, Deqing Zou

Cluster and Grid Computing Lab
Huazhong University of Science and Technology, Wuhan, 430074, China
{hjin, wzqiang, xhshi, deqingzou}@hust.edu.cn

Abstract. In this paper we analyze the access control requirements of the dynamic virtual organization in grid environment and define an absolutely decentralized access control mechanism for the dynamic coalition characteristic of virtual organization. The access control framework in the paper combines the threshold BLS signature schema and the role based access control mechanism to provide a flexible and decentralized mechanism for the VO-based grid applications.

1. Introduction

Grid technology is emerging as a novel form of distributed computing system. With the heterogeneity, dynamic and autonomy characteristics of grid system, the security problem requires much more than the traditional distributed computing system. *Grid Security Infrastructure* (GSI) [1] provides encryption transferring, authentication, and simple authorization mechanism.

To solve the cooperative cross organizations, the *virtual organization* (VO) [2] in grid technology comes up, which is defined as a group of grid entities (e.g., resources, services, and users) from different domains collaborating in order to complete some tasks. Security requirements within the VO-supported grid environment have changed to support scalable, dynamic, autonomous VOs.

As an example for grid virtual organization, there is a hypothetical research project, which may consist of several research teams from different universities or institutes collaborating on some complicated physical problems, which involve some operations of laboratorial instruments and access of some experiment data. The instrument and data may be sensitive, so the operation and access must be restricted.

Above is a typical coalition environment, and all the member domains collaborate into a coalition. Because each member domain (each research team) is *autonomous*, a centralized administrator for the virtual organization is not appropriate. The access control policies for the instruments and data must be jointly administrated by all of the member domains. Some research teams may depart from the virtual organization, and

* This paper is supported by ChinaGrid project funded by Ministry of Education of China and National Science Foundation under grant 90412010.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 370-381, 2005.

new research teams may also join in the virtual organization. Therefore, how to make the access control administration adapt to the *dynamic* characteristic of the virtual organization is another challenge. Also, the research project seems to be a large project and includes many participant research teams. It means that the access control framework of the virtual organization must be *scalable* with the size variability.

In the conventional access control administration for virtual organization [3][4], a centralized administrator in the virtual organization manages the access control of resources/services. Because of the centralized characteristic, the autonomous, dynamic and scalable requirements can not be satisfied.

In this paper, we present an access control framework for dynamic virtual organization. Unlike centralized VO access control management, we utilize the threshold signature mechanism in order to provide a decentralized access control scheme. Also the role based access control technology is used to provide a flexible and autonomous access control method.

The paper arrangement is as follows. Section 2 introduces some related works. Section 3 presents the threshold BLS signature techniques. Section 4 describes the policy model for access control in virtual organization. Section 5 discusses the membership and authorization protocol for VO access control. We conclude this paper in section 6.

2. Related Works

In this section, we discuss the related works in the area of access control in grid systems and dynamic coalitions.

In *Community Authorization Service* (CAS) [3], the owners of resources grant access to a community account as a whole. The CAS server is responsible for managing the policies that govern access to a community's resources. It maintains fine-grained access control information and grants restricted GSI proxy certificates to the users of the community. *Virtual Organization Management System* (VOMS) [4] is similar to CAS. The difference between CAS and VOMS is that CAS specifies the low-level access rights to users, but VOMS specifies the role and VO membership attribute to users. The access control policies in CAS and VOMS are both controlled by a centralized point, which is not appropriate for the decentralized and autonomous requirements in dynamic virtual organization.

Akenti [5] is an access control architecture, which can address issues that all the resources are controlled by multiple authorities. PERMIS [6] is a policy-driven RBAC *Privilege Management Infrastructure* (PMI), in which the policies may be widely distributed. Both Akenti and PERMIS can incorporate multiple stakeholder policies, which means that the access control polices are autonomously managed by the stakeholders. Some efforts focus on integrating Akenti and PERMIS into grid system [7][8]. But as a matter of fact, these two systems only provide policy-decision engine and police repository, they have not provided the actual policy management mechanism. Our system utilizes Akenti and PERMIS for policy repository and decision-making.

Some researches on distributed, dynamic applications are similar to our research. Khurana [9] presents a method for joint administration of the access policies in

coalition resources. Threshold RSA signature techniques are utilized in order to provide threshold signature for each operation request. But the Threshold RSA method can not adapt to the dynamic coalition that member domains may leave and new ones may join in. The dynamic characteristic requires re-keying the RSA keys. The generation of RSA keys requires high computation overhead, which can not satisfy the scalability requirement of dynamic coalitions.

Byrd [10] and Smith [11] provide a framework for joint policy management and auditing in virtual organizations. They also utilize the threshold RSA method for joint control of access control policies. Their efforts have the same shortcoming as [9].

Nita-Rotaru [12] illustrates an access control framework for group communication systems. Centralized group communication server decides and enforces access control policies. The focus of the framework is on group communication control, which controls who can send/receive messages. But the access control of dynamic virtual organization requires controlling the access rights for each VO component.

3. Using Threshold BLS Techniques

In this paper, we use the threshold signature scheme for the access control policy administration to fulfill the requirement of autonomous. Because of the dynamic adaptation characteristic of threshold signature scheme, the dynamic requirement of virtual organization access control policy administration can also be fulfilled.

3.1 Threshold Signature Techniques

The threshold signature scheme is similar to the voting mechanism. If enough votes are collected, the request is approved. Let n ($\geq k$) be the current number of group members. In the (k, n) threshold scheme, a secret x is split into n shares. The secret of group can be reconstructed if k secret share are presented. The group member wants to get the approval of at least k member of the group in order to produce a valid signature. Any malicious member must compromise at least k member of the group in order to disguise a valid signature.

In threshold signature scheme, the signature secret is divided into k parts and shared among the group members. The group members can use the shares of signature key in order to compute a signature on some messages.

The threshold signature scheme can be classified into two types, static threshold and dynamic threshold. Static threshold is a (k, n) threshold scheme where k is a fixed value, and the number of the group n may vary. This scheme is not efficient at the group initiation period or when a large number of members depart from the group. In these situations, the current number of the group n may be less than k.

Dynamic threshold is a (k, n) threshold scheme where k may also vary. A typical dynamic threshold is that k may vary with the fixed fraction of the current group size n. The dynamic threshold scheme is more flexible and can adapt to the variability of the group size. But the main problem for dynamic threshold is that it is difficult to reliably determine the size of the current group.

3.2 Threshold BLS Signature Scheme

In this section we give a brief overview of threshold BLS signature techniques. Then we give the usage of threshold signature for the administration of access control policies. We also consider how the threshold BLS adapts to the dynamic characteristic of virtual organization.

The threshold signature mechanism in this paper is from the BLS signature scheme [13], and the threshold version [14] bases on BLS. Saxena [15] compares among the threshold RSA, threshold DSA, threshold Schnorr and threshold BLS.

The reason why we use the BLS threshold scheme is as follows. Threshold BLS is based on the *elliptic curve discrete logarithm problem* (ECSLP), and 160-bits ECC system is as secure as the 1024-bits RSA system, so the computing cost of BLS threshold scheme is much less than the discrete logarithm schemes, such as RSA and DSA schemes. The signature of BLS scheme is also shorter than the discrete logarithm schemes.

In the threshold BLS signature mechanism, a trust dealer of a group (that means the initiator of the virtual organization) first initiates and generates the elliptic curve domain, some information about initiation step is defined as follows.

Definition1: All the members of a group can communicate with each other by secure point-to-point channel.

Definition2: The elliptic curve domain parameters are (p, F_p, a, b, A, q). The elliptic curve is represented by: $y^2 = x^3+ax+b$.

Definition3: e: $G_1 \times G_1 \rightarrow G_2$, is defined to be a public binary mapping.

Definition4: $H_1:(0,1)^* \rightarrow G_1^*$ is the hash function that maps binary strings to non-zero points in G_1.

All the information above is accessible to each member of the group. After the information definition, the initiator (trust dealer) must compute the secret share for each member in the group. The initiator selects a random polynomial, $f(z)=f_0+f_1z+\ldots+f_kz^k$ over Z_q of degree k, then the group secret is $f(0)=f_0=x$, and the group public key is xA. Furthermore, each secret share ss_i (i=0,…,k) can be computed as $ss_i=f(id_i) \pmod q$, where id_i is the unique identify number of each member. The group membership token GMT_i for existing members can be issued with the group secret x. The initiator is not required thereafter.

4. A Policy Model for Access Control in Virtual Organization

In this section, we study policy model for access control in dynamic virtual organization. First, the example scenario is presented in details in order to discuss the policy requirements of virtual organization. And then the policy model is presented, which includes the VO template policy model, the access control model, and the voting policy for each member in a VO.

4.1 An Example Scenario for Virtual Organization

We still use the example of several research teams from different organizations compose the hypothetical research project. The members (the users of the virtual organization) in the virtual organization may jointly control the access to the resources (documents, instruments and the others). Figure 1 illustrates the composing of a virtual organization. From this figure, some resources/services and users from different security domains compose a virtual organization, and the participant users also jointly control these resources/services by the VO access control policies.

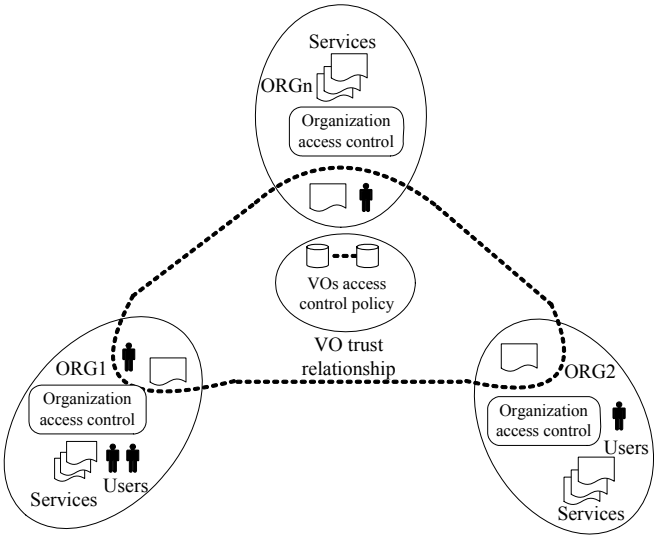


Figure 1. An illustration for a virtual organization

The users may have diverse roles, and each role can accomplish various levels of access permission, which is a typical role based access control model. First, a user who intends to participate in the research project must become a member of the virtual organization. Then the user may try to join one of the roles of the virtual organization, which means that he intends to be assigned certain role and get some authorization of the virtual organization. All users that belong to the virtual organization jointly determine the user-role assignment decision by a voting mechanism, which is a k out of n threshold scheme. In this paper, we use the dynamic threshold signature scheme, which means a fixed factor value must be predefined for the authorization decisions.

From the discussion above, for the policy definition of the virtual organization, we can conclude that besides the access control policies defining the user-role assignment and role-permission assignment, there are requirements to define the virtual organization *template policies* and *voting policies*. The template policies prescribe the specifications for the virtual organization, such as threshold signature algorithm, threshold factor definition. The voting policies prescribe the specifications for the membership decision and authorization decision. The explanations of these policies are as follows.

4.2 The VO Template Policy

The template policy for a virtual organization includes the following aspects:

Definition5:

- *Threshold signature algorithm*: The template policy predefines the threshold signature scheme, such as threshold RSA scheme, threshold DSA scheme, or threshold BLS scheme.
- *Administration role set definition*: Some high privilege roles of the virtual organization may be predefined in the template policies in order to act as the administration role, such as the *project director* or the *privilege manager* of the research project.
- *Authorization lifetime definition*: Authorization in this paper is assignment some role to a specific membership (user) of the virtual organization. In order to guarantee the security of authorization, we specify a lifetime for each role from the role set definition.
- *Threshold factor definition*: We use the dynamic threshold in this paper. So the threshold factor must be specified in the template policies.
- *Role-permission assignment for administration roles*: Some administration permissions are assigned to the administration roles. These constraints define the permissions that *administration roles* may hold, which will be explained in the section 4.3.

In Figure 2 we present a sample template policy for the above research project virtual organization. The threshold signature is predefined as BLS. There are two administration roles, *project director* and *privilege manager*. The lifetime for the two roles is 12 hours and 48 hours, respectively, which means that any user who is assigned one of the roles will lose the authorization after the lifetime period from the authorization assignment.

Threshold signature scheme: BLS
Primary role set: {*project director*, *privilege manager* }
Authorization lifetime definition: {
{ *project director*, *12 hours*}
{ *privilege manager* , *48 hours*}
}
Threshold factor definition: {
{*member*, *45%*}
{ *project director*, *95%*}
{ *privilege manager* , *100%*}
}
Role-permission assignment: {
{ *project director*, modify *RR* set}
{ *privilege manager* , modify role-permission assignment for roles in *RR* set}
}

Figure 2. A sample template policy

The threshold factor definition for each role is 95% and 100%, respectively, which means that the user who wants to be assigned to be the *project director* or *privilege manager* must get 95% and 100% approval votes of all the membership of the research

project, respectively. The security level of the "role" definition determines the difference of the factor value. The higher factor value the "role" has, the more approval votes the user must collect in order to get the assignment of the role. In fact, the threshold factor value is a tradeoff between the security and performance cost. The larger the factor value, the more performance cost will be taken for the authorization, as more approval votes require more computing.

The membership threshold factor is defined to control the users' membership to be an eligible member of the virtual organization.

The template policies are some regulations that must be complied during the lifecycle of the virtual organization. The definition of template policies is out of the scope in this paper. Template policies are securely signed by a trusted third party in order to ensure the integrity and authenticity. Each member of the virtual organization gets the template policies and uses them when trying to get some authorizations or when making some voting decisions.

4.3 Access Control Model for Virtual Organization

When users intend to be a member of a virtual organization and access the resources/services inside the VO, they need to be authenticated first. Some authentication mechanisms, such as user/password, Kerberos, or PKI are available. In this paper, we use the PKI authentication. Each member of a virtual organization gets a *public key certificate* (PKC) from a trusted *certificate authorization* (CA). The certificate authorization also acts as the trusted third party and provides a signature for the template policies.

After the authentication period, a user of the virtual organization may try to get the authorization to access the resources/services, which means getting certain user-role assignment. How the user gets the authorization is presented in section 5.

Here we present the role based access control model for the virtual organization framework. The following is the access control policy model.

Definition6: The virtual organization access control policies consist of the following parts:

- The permission set of a virtual organization is defined as: $P=AP \cup RP$. The permission set consists of two parts. One is the VO resource access permissions, *RP* for short. The *RP* permissions are binary relation of operation and object (*Oper*×*Obj*), and can be exemplified as "modify document1" or "manipulate microscope instrument"; The other is the administration permissions, *AP* for short, such as modifying the role-permission assignment or modifying the role set configuration of the virtual organization. The administration permission must be assigned to some roles with high-level privilege, such as the "*privilege manager*" role in the research project virtual organization.
- The role set of a virtual organization is defined as: $R=AR \cup RR$. The role set also consists of two parts. One is the administration roles, *AR* for short. The administration role set is predefined in the template policies and can not be modified any more in the lifecycle of the virtual organization. The other is the VO resource access roles, *RR* for short. *RR* can only be configured and modified by users with administration role.

- The lifetime and threshold factor constraints for the roles in role set, which is similar to the definition in template policies. In fact, the constraints for roles consists two parts. One is for administration roles, and the other is for VO resources access roles. They are defined to constrain the user-role assignment binary relation $UA \subseteq U \times R$.
- The binary relation $PA \subseteq R \times P$, which determines the role-permission assignment relation. Unlike the centralized access control model in conventional virtual organization security framework, because there is no centralized administrator in our access control framework, the assignment constraints can only be configured and modified by users with higher privilege role. The role-permission for roles in *RR* set is constrained by users that are assigned administration roles in *AR*. The role-permission for roles in *AR* set is constrained by template policies that are illustrated above.

4.4 The Voting Policy Definition

The authorization (user-role assignment) decision is jointly made by all the users (members) of the virtual organization. We use threshold signature scheme for authorization making, which will be discussed in section 5.

Each member of the virtual organization may independently define voting policy for each voting request.

Definition7: The voting policies of each member are defined as follows:

- *VP* is the set of voting policies for each member. Each voting policy maps each role to a set of voting rules. Each voting rule has the form (*constraint_req, vo_context*). If the user satisfies the constraint *constraint_req* and the context of the virtual organization is *vo_context*, the voting rule approves the authorization. Each member of the virtual organization can modify the voting policies that he owns.

In Figure 3 a sample voting policy for a certain member of the research project virtual organization is provided. For the role "project director", one voting rule is that the name of the authorization request starting with "*/O=CHINAGRID/OU=HUST/*", and the context of the virtual organization is "ongoing". The other voting rule is that the name of the authorization request is "*/O=CHINAGRID/OU=Tsinghua/CN=XX*", and the context of the virtual organization is "initial". If any voting rule is satisfied, the member will make an "approval" decision.

All members of the virtual organization make decisions according to the voting policy that he owns. If enough "approval" decisions are made for the authorization request, the request is approved and the user will get the authorization.

The membership voting policy is also defined in order to respond new member request. In the sample voting policy, a membership request started with the distinguished name "*/O=CHINAGRID/*" can get the "approval" membership vote from the member.

For scalability consideration, the members can make voting decision according to the attribute of the requestors, not the direct ID of them. Thus, each member doesn't need to maintain voting policies for each requestor.

```
Voting policy:{
project director:{
{DN="/O=CHINAGRID/OU=HUST/CN=*", VO_context=ongoing},
{DN="/O=CHINAGRID/OU=Tsinghua/CN=XX", true}}
privilege manager:{
{DN="/O=CHINAGRID/OU=HUST/CN=HJ", VO_context=initial}}
analysis scientist:{
{DN="/O=CHINAGRID/OU=ICT/CN=*", VO_context=ongoing}}
member:{
{DN="/O=CHINAGRID/*", true}}
……
}
```

Figure 3. A sample voting policy

5. Membership and Authorization Protocol for VO Access Control

In general, a user who wants to take part in the virtual organization must first get the membership. Then he must get authorization of the virtual organization. In this paper, both the membership-getting and authorization-getting processes are achieved by threshold signature scheme.

In this section, we present the membership protocol and authorization protocol that uses threshold BLS scheme for the virtual organization access control.

5.1 The Membership Protocol

The membership protocol is designed for the admission control of the virtual organization. In this paper, we use the admission control mechanism from [16]. The membership protocol consists of the following steps:

Step 0: *Initiation*. First, the information of current group size n is collected by some approaches in fault detection techniques. Second, the threshold k for the group membership can be computed by n multiplying the membership threshold factor, $k=n\times threshold\ factor$. Third, an initiator computes the security parameters such as the secret share (ss_i) and group membership token (GMT_i) (i=0,…,k) for the random k members of the virtual organization. Then the security parameters are securely transferred to the k members by secure point-to-point channel. The initiator is no longer required after the initiation step. The initiation step will be needed only when threshold changes.

Step 1: *Membership Request*. A new user sends a membership request (*GMT_REQ*) to all the members of the virtual organization. The *GMT_REQ* includes the requestor's *public key certificate* (*PKC*) to identify the requestor.

Step 2: *Voting decision*. The receivers of the *GMT_REQ* extract the requestor's *GMT_REQ* and make voting decision according to the local voting policy and the requestor's identity. If the request is approved, the receiver replies with its membership token GMT_i and an approval vote $vote_i$ (signed by the secret share ss_i of the receiver).

Step 3: Membership Token Acquisition. Once getting t ($t \geq k$) votes, the new requestor randomly choose k out of them. Then the requestor validates the membership of the k members by the GMT_i, verifies the k votes, and computes its own membership token GMT_{new} from the k votes.

In fact, except the k members' membership tokens initially generated by the initiator, the n-k members' membership tokens are all generated by the initial k members, just as the new membership requestor.

Because we use dynamic threshold in this paper, the threshold factor is fixed and the threshold k must vary with new member joining or existing member departure. The secret share of each member will also change with the threshold changing dynamically. All the membership of the members will be updated once more.

But updating the threshold and the following operations are expensive, it is not practical to swiftly update the threshold when the group changes. In this paper, we use a lazy-updating mechanism. In detail, we introduce a lazy factor. If the threshold k_{new} ranges between the two values $n \times (threshold\ factor \pm lazy\ factor)$, the existing threshold k_{old} will still be used for the signature scheme.

Figure 4 presents the membership getting process (step 1 ~ step 3).

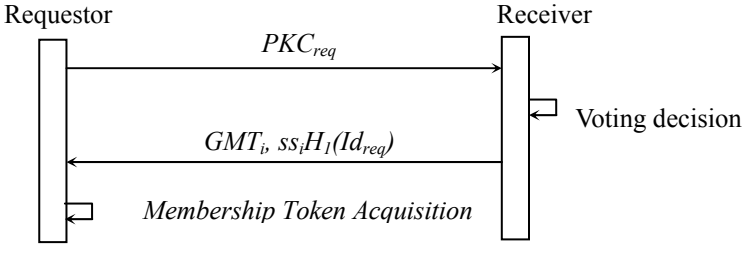


Figure 4. The membership protocol

5.2 The Authorization Protocol

If a member with the membership of the virtual organization must be assigned a role and get authorization, the authorization protocol must be applied.

The authorization protocol is similar to the membership protocol except some differences. We illustrate the authorization protocol as follows:

Step 0: *Initiation*. The initiation process is the same as the corresponding step in membership protocol. But there is no need to compute the security token for k members, such as GMT_i in membership protocol. The threshold of an authorization (authorization mean a user-role assignment relation) k can be computed by n

multiplying the role threshold factor (each role independently has a threshold factor), and each role independently has its own security parameter definition.

Step 1: *Authorization Request*. A member of the virtual organization sends an authorization request (*AUTHZ_REQ*) to all the members. The *AUTHZ _REQ* includes the requestor's group membership token (GMT_{req}), the member's PKC_{req} and the authorization request (for example, the *role1* assignment request).

Step 2: *Voting decision*. The receivers of the *AUTHZ_REQ* validate the *GMT* of the authorization requestor, extract the requestor's user-role assignment request and make voting decision according to the local voting policy and the requestor's identity. If the request is approved, the receiver signs a user-role assignment assertion (a vote for the request) with the share ss_i for the *role1*, and the lifetime of the authorization is configured in the assertion. Then the vote and the GMT_i of the receiver are transferred to the authorization requestor.

Step 3: Authorization Token Acquisition. If the authorization requestor gets at less k votes, k of them is selected and composed to generate the authorization token. The authorization token can be validated by the public secret of the group for *role1*.

Also, the lazy factor is used for the threshold of each role.

Figure 5 describes the authorization getting process (step 1 ~ step 3).

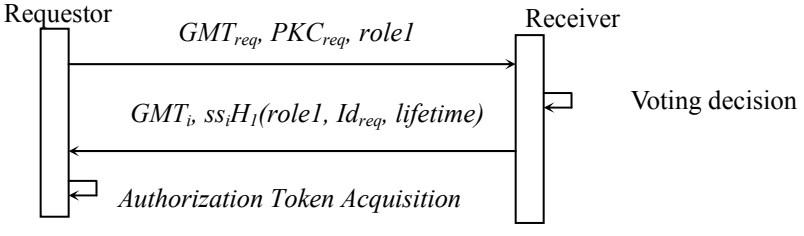


Figure 5. The authorization protocol

There is "lifetime" definition in membership token and authorization token; thus, the revocation of these tokens is implicitly implemented by the "lifetime" constraints.

6. Conclusion and Future Work

In this paper, we propose VO-Sec, an authorization and access control framework for dynamic virtual organization. VO-Sec uses the ideas in threshold signature techniques and role based access control model. We design a policy model for access control in virtual organization, which is composed of the template policy, RBAC access control policy and voting policy. Also, we propose the membership and authorization protocol. The framework presented in this paper can satisfy the dynamic and autonomous requirements of the access control management in virtual organizations.

We plan to use the proposed scheme in some grid systems, such as Globus Toolkit or CGSP (*China Grid Support Platform*) in the future.

References

1. I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke, "A Security Architecture for Computational Grids", *Proceedings of the 5th ACM Conference on Computer and Communications Security*, pp.83-92, San Francisco, CA, USA, 1998.
2. I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", *International Journal of High Performance Computing Applications*, 15 (3): 200-222, 2001.
3. L. Pearlman, V. Welch, I. Foster, C. Kesselman, and S. Tuecke, "A Community Authorization Service for Group Collaboration", *Proceedings of the IEEE 3rd International Workshop on Policies for Distributed Systems and Networks*, Monterey, CA, 2002.
4. R. Alfieri, R. Cecchini, V. Ciaschini, L. dell'Agnello, A. Frohner, A. Gianoli, K. Lörentey, and F. Spataro, "VOMS, an Authorization System for Virtual Organizations", DataGrid Project, 2003, http://grid-auth.infn.it/docs/VOMS-Santiago.pdf.
5. M. Thompson, W. Johnston, S. Mudumbai, G. Hoo, K. Jackson, and A. Essiari, "Certificate-based Access Control for Widely Distributed Resources", *Proceedings of the Eighth Usenix Security Symposium*, Aug. 1999.
6. D. Chadwick and A. Otenko, "The Permis X.509 Role Based Privilege Management Infrastructure", *Proceedings of SACMAT 2002 Conference*, ACM Press, pp.135-140.
7. http://dsd.lbl.gov/akenti/codeDist/GRAMAkentiAuthz.html
8. D. W. Chadwick, "An Authorization Interface for the GRID", Presented at the *E-Science All Hands Meeting 2003*, Nottingham, Sept 2-4, 2003.
9. H. Khurana, V. D. Gligor, and J. Linn, "Reasoning about Joint Administration of Access Policies for Coalition Resources", *Proceedings of ICDCS 2002*, pp.429-443, Vienna, 2002.
10. T. Byrd, F. Gong, C. Sargor, and T. J. Smith, "Yalta: A secure collaborative space for dynamic coalitions", *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security*, pp.30-37, IEEE, United States Military Academy, West Point.
11. T. J. Smith and L. Ramakrishnan, "Joint Policy Management and Auditing in Virtual Organizations", *Proceedings of 4th International Workshop on Grid Computing*, Phoenix, Arizona, 2003.
12. C. Nita-Rotaru and N. Li, "A framework for Role-Based Access Control in Group Communication Systems", *Proceedings of the 17th International Conference on Parallel and Distributed Computing Systems*, San Francisco, CA, 2004.
13. D. Boneh, B. Lynn, and H. Shacham, "Short Signatures from the Weil Pairing", *ASIACRYPT'01*, Volume 2248, LNCS, (2001) 514-532
14. A. Boldyreva, "Efficient threshold signatures, multisignatures and blind signatures based on the Gap-Diffie-Hellman-Group Signature Scheme", *Proceedings of International Workshop on Practice and Theory in Public Key Cryptography*. Volume 2567, LNCS, (2003) 31-46.
15. N. Saxena, G. Tsudik, and J. H. Yi, "Access Control in Ad Hoc Groups", *Proceedings of International Workshop on Hot Topics in Peer-to-Peer Systems*, Volendam, Netherlands, 2004.
16. Y. Kim, D. Mazzocchi, and G. Tsudik, "Admission Control in Peer Groups", *Proceedings of IEEE International Symposium on Network Computing and Applications (NCA)*, 2003.

An Efficient Implementation of a Threshold RSA Signature Scheme

Brian King

Purdue School of Engineering & Technology
Indiana Univ. Purdue Univ. at Indianapolis
briking@iupui.edu

Abstract. Several threshold RSA signature schemes have been proposed, in particular the schemes [4, 8] and [20]. Recent research has shown that the earlier schemes [4, 8] may be in some cases more "efficient" than these later schemes. Here we describe efficient implementations of threshold RSA schemes as well as further enhancements to improve performance of the Desmedt-Frankel scheme. Our conclusion is that if memory is not an issue there will be situations, for example if one can expect shareholders know who will be participating in the signature generation, that the Desmedt-Frankel scheme is very efficient.

1 Introduction

Digital signatures provide a means for binding a message and an identity. In many applications it is desirable that the secret key (signature key) does not reside on a single device, but rather that it is distributed via some sharing mechanism to several devices. For example, in high-level government setting, a single device may become compromised or disabled. If the secret key had been solely safeguarded on that device then authorized people may no longer able to generate signatures with that secret key and/or unauthorized people, that possess the device or key, may be able to sign. There are numerous other examples which illustrate the need for distributing the key to several devices. A t out of n threshold scheme is such that shares of secret k are distributed to n participants so that any set of t participants can compute k, and where any subset of $t-1$ or less participants gain no information about k.

RSA [18] is a popular cryptographic primitive. The development of threshold RSA was problematic due to the fact that the modulus $\phi(N)$, as well any multiple, cannot be leaked to any of the shareholders. Threshold RSA has been examined in [5, 6], then in [4, 8, 13], and most recently in [1, 10, 11, 12, 17, 20]. The Desmedt-Frankel scheme [8] was the first secure threshold RSA sharing scheme. This is a zero-knowledge threshold scheme (for a formal definition of zero-knowledge threshold schemes see [8]). Further this scheme is a group independent scheme. That is, the shareholder's reconstruction of the secret key is independent of the group. Group independent schemes provide a flexible method to achieve threshold secret sharing. However, there is a disadvantage when using

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 382–393, 2005.

the Desmedt-Frankel scheme. The disadvantage is the amount of resources it requires, in the sense of memory (share size) and processing (computational time). The memory requirement is caused by share expansion. The share expansion is such that a single share will consist of $O(n)$ subshares (where n is the number of shareholders) drawn from the keyspace. The processing cost comes from the computing requirements. Computations will need to be performed on these large shares. Moreover, computations will need to be performed in an extension ring. These resource requirements (and interest in development of robust, proactive, and/or verifiable threshold RSA) has led to searching for other schemes. In [20], Shoup described his Practical Threshold Signatures, which is widely regarded as the most efficient threshold RSA signature scheme. Further improvement are being developed, for examples schemes which include key generation [3].

Work has been initiated in realizing the precise computational requirements for the Desmedt-Frankel scheme. In [9], the authors established that within the Desmedt-Frankel scheme, the share size for each participant could be halved. In [14, 15], algorithms were developed to reduce the number of required computations when using the Desmedt-Frankel scheme. Further, a comparison of the computations required by the Desmedt-Frankel signature scheme with Shoup's Practical Threshold Signature scheme was made. It was pointed out that in many cases, it appeared that the Desmedt-Frankel scheme performed better than the signature scheme developed by Shoup (as long as it is assumed that the shareholders who wish to form the signature know all others who are willing to sign). The comparison was developed using complexity theory. In the following, we describe the results of a software implementation of [4, 8]. This work illustrates that with application of algorithms developed in [14, 15] significant performance improvement of the Desmedt-Frankel scheme (from now on we refer to it as the DF scheme) can be achieved. We also discuss further improvements. We provide a comparison between the work required by a shareholder of the DF scheme, compared to the work done by a RSA signer, as well as, compare it to the work done by a shareholder in Shoup's scheme. Although our results are based on comparison of the DF scheme with the Shoup scheme, one would see similar results if they compare the DF scheme to any threshold RSA scheme which avoids using the *Lenstra constant*[1] (see [8]).

2 Background: The Desmedt-Frankel Scheme

In [8], Desmedt and Frankel showed how to share with zero-knowledge a secret over any finite abelian group. In [4], Desmedt, De Santis , Frankel and Yung extended this to zero-knowledge sharing a function over any finite abelian group. Of course the motivation was to develop a zero-knowledge sharing of RSA keys.

[1] The Lenstra constant is the cardinality of the largest set of units in $\mathbf{Z}[u]$ whose differences are also units (here $\mathbf{Z}[u]$ is the algebraic extension of the integers). The Lenstra constant maybe be used if one tries to generalize Shamir's secret sharing, in this case one needs a unit for each participant. Hence, in this case the Lenstra constant needs to be $\geq n$.

Let us describe the setting. Let $\mathcal{K}$ be any finite abelian group (in the case of RSA, $\mathcal{K}= \mathbf{Z}_{\phi(N)}$) and we denote the secret key by k where $k \in \mathcal{K}$. Let q be a prime satisfying $q \geq n+1$, where n is the number of participants. Due to Tchebychev [16], we can assume that $O(q) = O(n)$.

The algorithm is based on extending the Shamir secret sharing scheme [19] to a module. Let u represent a root of the cyclotomic polynomial $\Phi_q(x) = \sum_{j=0}^{q-1} x^j = \frac{x^q-1}{x-1}$. Many computations will be performed in the ring $\mathbf{Z}[u] \cong \mathbf{Z}[x]/\Phi_q(x)$. Because q is a prime, it follows that $\alpha_i = \sum_{j=0}^{i-1} u^j = \frac{u^i-1}{u-1}$ is a unit and that $\alpha_i - \alpha_j$ are units for distinct i, j. The ring $\mathbf{Z}[u]$ has very nice behavior and several interesting properties can be derived. First we can define $\alpha_q = 0$, then for all nonzero x we can define $\alpha_x = (u^x - 1)/(u-1)$. Integral to this definition is the use of the property that $u^q = 1$. Further one can show $\alpha_x = \alpha_{x \bmod q}$. Additional properties include: (1) $\alpha_i - \alpha_j = u^{i-j}\alpha_{i-j}$, (2) $\frac{\alpha_{ay}}{\alpha_y} = 1 + u^a + \cdots + u^{a(y-1)}$, (3) $\frac{\alpha_x}{\alpha_y} = \frac{\alpha_{y(y^{-1}x)}}{\alpha_y}$ (Recall that $\mathbf{Z}_q$ is a field, thus y^{-1} exists for nonzero y), and (4) $\alpha_{-x} = -u^{-x}\alpha_x$

Consider $\mathcal{K}^{q-1} = \mathcal{K} \times \mathcal{K} \times \cdots \times \mathcal{K}$. $\mathcal{K}^{q-1}$ is an additive abelian group. If $\boldsymbol{x} \in \mathcal{K}^{q-1}$ then $\boldsymbol{x} = [x_0, x_1, \ldots, x_{q-2}]$. Let $\mathbf{0} = [0, 0, \ldots, 0]$, where 0 denotes the identity in $\mathcal{K}$. For all $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{K}^{q-1}$ $\boldsymbol{x}_1 + \boldsymbol{x}_2 = [x_{1,0} + x_{2,0}, x_{1,1} + x_{2,1}, \ldots, x_{1,q-2} + x_{2,q-2}]$.

A scalar arithmetic is defined over $\mathcal{K}^{q-1}$, with scalars from $\mathbf{Z}[u]$ as follows. For all $b \in \mathbf{Z}$, define $b\boldsymbol{k} = [bk_0, bk_1, \ldots, bk_{q-2}]$. Define $u\boldsymbol{k} = [0, k_0, k_1, \ldots, k_{q-3}] + [-k_{q-2}, \ldots, -k_{q-2}]$. Inductively define $u^{i+1}\boldsymbol{k} = u^i(u\boldsymbol{k})$. Lastly, for all $f(u) = b_0 + b_1 u + \cdots + b_k u^k \in \mathbf{Z}[u]$, $f(u)\boldsymbol{s}$ is defined by $f(u)\boldsymbol{k} = \sum_{i=0}^{k} b_i(u^i\boldsymbol{k})$. Then $\mathcal{K}^{q-1}$ is a module over $\mathbf{Z}[u]$.

2.1 How Shares Can Be Computed the Distributor/Dealer

Given secret k, we can embed the secret k in $\mathcal{K}^{q-1}$ by setting $\boldsymbol{k} = [k, 0, \ldots, 0] \in \mathcal{K}^{q-1}$. For B a set of t participants we define $y_{i,B} \in \mathbf{Z}[u]$ by $y_{i,B} = \sum_{i \in B} \prod_{\substack{h \in B \\ h \neq i}} \frac{0 - \alpha_h}{\alpha_i - \alpha_h}$.

Each shareholder P_i, for $i = 1, \ldots, n$, is given share $\boldsymbol{s}_i$ in the following manner: $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_{t-1}$ are chosen uniformly random from $\mathcal{K}^{q-1}$. For $i \geq t$, let $C_i = \{1, 2, \ldots, t-1, i\}$ and $\boldsymbol{s}_i$ is computed using $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{t-1}$ as follows: $\boldsymbol{s}_i = y_{i,C_i}^{-1} \cdot \left(\boldsymbol{k} - \sum_{\substack{j \neq i \\ j \in C_i}} y_{j,C_i} \cdot \boldsymbol{s}_j\right)$ where $y_{j,C_i} = \frac{\prod_{\substack{h \in C_i \\ h \neq j}} (0 - \alpha_h)}{\prod_{\substack{h \in C_i \\ h \neq j}} (\alpha_j - \alpha_h)}$ for each $j \in C_i$.

2.2 How the Secret k Is Computed

When a set B of t participants wish to compute $k \in \mathcal{K}$, they can determine $\boldsymbol{k}$, of which the first coordinate is the secret k. $\boldsymbol{k} = \sum_{i \in B} y_{i,B} \cdot \boldsymbol{s}_i = \sum_{i \in B} \prod_{\substack{h \in B \\ h \neq i}} \frac{0 - \alpha_h}{\alpha_i - \alpha_h} \cdot \boldsymbol{s}_i$. Then $k = F_0(\boldsymbol{k})$ where $F_0 : \mathcal{K}^{q-1} \longrightarrow \mathcal{K}$ by $F_0([x_0, \ldots, x_{q-2}]) = x_0$.

To compute the RSA signature of message M

$$\textit{Signature of } M \ = m^k = \prod_{i \in B} m^{F_0(y_{i,B} \cdot \boldsymbol{s}_i)}$$

where $m = hash(M)$ (here k is the RSA private key and $m^{F_0(y_{i,B}\boldsymbol{s}_i)}$ is the partial signature of participant P_i).

It follows then that the DF scheme is a generalization of Shamir's secret sharing scheme. In Shamir's scheme, one is sharing points on a "curve", where each point of the curve belongs to $\mathbf{Z}_p \times \mathbf{Z}_p$. Whereas in the DF scheme, one is sharing a point on a "curve" where each point belongs to $\mathbf{Z}[u] \times \mathcal{K}^{q-1}$. However one's view of that point depends on who they are. For example, consider the case we are using the DF scheme to achieve threshold RSA sharing. Then a shareholder does not know $\mathcal{K}$ (which is $\mathbf{Z}_{\phi(N)}$). Thus they see a point as a point belonging to $\mathbf{Z}[u] \times \mathbf{Z}^{q-1}$. Whereas the distributor knows $\mathcal{K}$ and so they view a point as a point in $\mathbf{Z}[u] \times \mathcal{K}^{q-1}$.

3 Problems That Arise with This Scheme

(1) *Size of share* $= n *$ size of secret key.
(2) *Performance.* In addition to working with expanded shares, shareholders and the distributor will need to perform computations in the ring $\mathbf{Z}[u]$, as well as the module operation $f_\xi \cdot \boldsymbol{s}$.

Regarding shareholder computations, each shareholder has a share $\boldsymbol{s}$ which consists of a $q-1$ subshares. To sign, the shareholder needs to compute $y_{i,B}\boldsymbol{s}$. A possible way to compute $y_{i,B}\boldsymbol{s}$, is to express $y_{i,B} = f_{\xi_{t-1}} \cdots f_{\xi_2} \cdot f_{\xi_1}$ and compute $y_{i,B}\boldsymbol{s}_i = f_{\xi_{t-1}} \cdots f_{\xi_2} \cdot (f_{\xi_1} \cdot \boldsymbol{s}_i)$ [8]. Another way to compute $y_{i,B}\boldsymbol{s}$, is to compute $y_{i,B}$ and then compute $y_{i,B}\boldsymbol{s}$ using the definition of the scalar arithmetic [8].

Regarding distributor computations, a distributor needs to generate $t-1$ random $\boldsymbol{s}_j$ which belong to $\mathcal{K}^{q-1}$. The distributor needs to compute $n-t+1$ many $\boldsymbol{s}_i$, where $\boldsymbol{s}_i = y_{i,C_i}^{-1} \cdot \left(\boldsymbol{k} - \sum_{\substack{j \neq i \\ j \in C_i}} y_{j,C_i} \cdot \boldsymbol{s}_j\right)$.

3.1 How Much Time Is Needed to Perform the Necessary Algebraic Operations

As described above there are two alternatives to compute $y_{i,B}\boldsymbol{s}$: either (1) first compute $y_{i,B}$ and then compute $y_{i,B}\boldsymbol{s}$ using the definition of the scalar arithmetic, or (2) express $y_{i,B} = f_{\xi_{t-1}} \cdots f_{\xi_2} \cdot f_{\xi_1}$ and compute $y_{i,B}\boldsymbol{s}_i = f_{\xi_{t-1}} \cdots f_{\xi_2} \cdot (f_{\xi_1} \cdot \boldsymbol{s}_i)$. In [15] it was shown that this second method is the most efficient method in terms of computational complexity. In terms of computational complexity, the time for the for the shareholder and distributor to perform the required group operations is: Shareholder performs $O(tq)$ group operations. Distributor selects $t-1$ random shares from $\mathcal{K}^{q-1}$ performs $O(tq(q-t))$ group operations.
In a threshold RSA signature scheme we have $\mathcal{K} = \mathbf{Z}_{\phi(N)}$. A significant computation that will need to take place by the shareholder is the modular exponentiation. In fact this computation will be the most time-consuming computation.

The time to compute the exponentiation is not included in the above table. If C is the size of the exponent (in bits) then the time to compute the exponentiation is $O(C(\log_2 N)^2)$.

4 Background: Practical Threshold Signature Scheme

In [20], Shoup described Practical Threshold Signatures. This scheme provides an extremely efficient way to develop a RSA threshold signature scheme. In addition, this scheme is provably secure in the random oracle model. Further, in an efficient manner, one can develop a robust threshold scheme. The share size in Shoup's scheme is on the order of the size of the secret, which is a tremendous improvement over Desmedt-Frankel's scheme. So the Practical Threshold Signature scheme provides many significant benefits. However under certain assumptions, we find that computationally there may be better threshold schemes. Our sole interest is to compare time required to perform the needed computations to achieve threshold RSA signatures. So we will omit comparison between the two schemes outside this realm. For example we will not include time required by a shareholder to perform proof of correctness of partial signatures. The reason for this is that schemes can utilize a proof of correctness (see [13] for a proof of correctness for the DF scheme), in fact one could modify the proof of correctness used in the Shoup scheme and apply it within the DF scheme provided that $N = pq$, $p = 2p' + 1$ and $q = 2q' + 1$ where p' and q' are prime.

We now discuss the Practical Threshold Signature scheme. Our discussion is limited to those details that are required to generate a threshold RSA signatures. Details such as proof of correctness are omitted in our discussion. For the complete details of the Practical Threshold Signature scheme we refer the reader to [20]. Let $N = pq$ where p and q are primes such that $p = 2p' + 1$ and $q = 2q' + 1$ where p' and q' are prime. Let $L = p'q'$. The dealer chooses an RSA exponent e. The dealer chooses $a_1, a_2, \ldots, a_{t-1}$ at random from $\{0, \ldots, L-1\}$ and defines the polynomial $f(x) = d + a_1 x = a_2 x^2 + \cdots + a_{t-1} x^{t-1}$. For each $i \in \{1, \ldots, n\}$ the dealer sends participant P_i the share $s_i = f(i) \pmod{L}$.
Let $\Delta = n!$. For any each subset B of t elements from set $\{0, 1, \ldots, n\}$ and for any $i \in \{0, 1, \ldots, n\} \setminus B$ and $j \in B$ define

$$\lambda_{i,j}^B = \Delta \frac{\prod_{j' \in B \setminus \{j\}} (i - j')}{\prod_{j' \in B \setminus \{j\}} (j - j')}.$$

Then for all $i \in \{1, \ldots, n\}$

$$\Delta \cdot f(i) \equiv \sum_{j \in B} \lambda_{i,j}^B f(j) \pmod{L}.$$

Signature and Signature shares. A hash function H is applied to message m. Let $x = H(m)$ for message m, then a signature for m is y such that $y^e = x$ (modulo N).

Each participant has received s_i from the dealer, so they can compute $x_i = x^{2\Delta s_i}$, which belongs to the subgroup consisting of all squares in $\mathbf{Z}_N^*$.

Each member belonging to set B (a set consisting of t participants) will: first compute $x_{i_j}^2 = x^{4\Delta s_{i_j}}$. (These steps are separated because a proof of correctness is sent using this $x_{i_j}^2$, which is omitted from our discussion.) Next the participant will compute $\mathcal{S}_{i_j} = x_{i_j}^{2\lambda_{0,i_j}^B}$ and send it to the combiner.[2] The combiner computes w by: $w = \prod_{i \in B} \mathcal{S}_i$. It follows then that $w^e = x^{e'}$ where $e' = 4\Delta^2$. Since $gcd(e', e) = 1$, by using the extended Euclidean Algorithm one can find a and b such that $ae' + be = 1$. The signature is $y = w^a x^b$. The signature can be verified by $y^e = x$

Observe the following: in the DF scheme, though there is share expansion, the additional computations is reserved to computations in the secret space. Whereas Shoup's scheme requires additional computations in the message space. Of course when signing with RSA, the message computations is more computationally intensive than computations in the secret space. In [15] the complexity for the shareholder computations for the DF scheme was given by $max\{qt(t \log_2 q + \log_2 \phi(N)), [t \log_2 q + \log_2 \phi(N)](\log_2 N)^2\}$ whereas the complexity of shareholder computations required by Shoup's scheme was given by $(n \log_2 n + \log_2 \phi(N)(\log_2 N)^2)$. Note that the q in the complexity for the DF scheme could be replaced by n, since prime q is on the order of n. Lastly note that in the case of threshold applications t and n will be much smaller than $\phi(N)$ and N. It is difficult to gauge the difference of the two schemes in terms of their performance, due to the fact that the complexity values use several different parameters, as well as, since complexity analysis will omit constant factors. The following benchmarks should be enlightening.

5 Software Implementation

We now describe relevant algorithms to achieve an efficient implementation of the DF scheme. The algorithms demonstrate that using the scalar arithmetic on the module is significantly more efficient than other threshold RSA signature schemes, as long as the shareholders know which shareholders will be sending their partial signature.

Recall that $\alpha_x = \frac{u^x - 1}{u - 1}$ for $x = 1, \ldots, q-1$. By observing that $u^q = 1$, one can extend the definition of α_x to exist for all integers x by defining $\alpha_0 = 0$ and for $x \neq 0$, $\alpha_x = \alpha_{x \bmod q}$. Recall that $\boldsymbol{k} = \sum_{i \in B} y_{i,B} \boldsymbol{s}_i$, where $k = F_0(\boldsymbol{k})$. If participant P_i knows all other members that belong to B, then they can compute $y_{i,B}\boldsymbol{s}_i$, and transmit the first coordinate to the combiner (the first coordinate is $F_0(y_{i,B}\boldsymbol{s}_i)$). In [14] it was shown that $y_{i,B} = \prod_{\substack{h \in B \\ h \neq i}} \frac{0 - \alpha_h}{\alpha_i - \alpha_h} = u^{-(t-1)i} \prod_{\substack{h \in B \\ h \neq i}} \frac{\alpha_h}{\alpha_{h-i}}$.

[2] We are assuming that the participant knows the members in B. If the shareholder does not know B, then they send to the combiner x_i and the combiner will need to compute $w = x_{i_1}^{2\lambda_{0,i_1}^B} \cdots x_{i_t}^{2\lambda_{0,i_t}^B}$.

We can express $\frac{\alpha_h}{\alpha_{h-i}} = \frac{\alpha_{a\tau}}{\alpha_a}$ where $a = h - i \bmod q$ and $\tau = \frac{h}{h-i}$ (here we compute $\frac{h}{h-i}$ in the field $\mathbf{Z}_q$). Lastly observe that $\frac{\alpha_{a\tau}}{\alpha_a} = 1 + u^a + \cdots + u^{a(\tau-1)}$.

In an effort to demonstrate a more distinguishable pattern when computing scalar arithmetic on elements in $\mathcal{K}^{q-1}$, we suggest to view each $\boldsymbol{x} = [x_0, \ldots, x_{q-2}]$ as a vector of length q where the last coordinate x_{q-1} will be by definition 0. That is, $\boldsymbol{x} = [x_0, \ldots, x_{q-2}, 0]$, i.e. we work in $\mathcal{K}^{q-1} \times \{0\}$. In [15] it was shown that if $u^a \boldsymbol{s} = \boldsymbol{x}$ then the j^{th} coordinate of $\boldsymbol{x}$ satisfies that $x_j = s_{j-a} - s_{-a-1}$. Consequently we can compute $u^a \boldsymbol{s}$ as follows. (Here all indices are reduced modulo q.)

Algorithm 1 *Algorithm to compute* $u^a \boldsymbol{s}$

Input=$(a, \boldsymbol{s})$.
Output=$\boldsymbol{z} = u^a \boldsymbol{s}$.

$z_{q-1} = 0$;
for $i = 0$ *to* $q - 2$
$z_i = s_{i-a} - s_{-a-1}$

It was also shown that $\boldsymbol{z} = \frac{\alpha_{a\tau}}{\alpha_a} \boldsymbol{s}$ had as well a recursive relationship such that the $(i+1)a^{th}$ coordinate could easily be computed from the ia^{th} coordinate. That is the coordinate $(i+1)a$ of $\boldsymbol{z}$ can be computed as: $z_{(i+1)a} = z_{ia} + s_{(i+1)a} - s_{ia-(\tau-1)a}$ (remember indices are reduced modulo q and that $z_{q-1} = 0$).

Algorithm 2 *Algorithm to compute* $\frac{\alpha_{a\tau}}{\alpha_a} \boldsymbol{s}$.

Input=$(a, \tau, \boldsymbol{s})$.
Output=$\boldsymbol{z} = \frac{\alpha_{a\tau}}{\alpha_a} \boldsymbol{s}$.

$z_{q-1} = 0$
$j = 0$;
for $i = 0$ *to* $q - 2$
$z_{j+a} = z_j + s_{j+a} - s_{j-(\tau-1)a}$
$j = j + a \ mod \ q$

For a shareholder to compute $y_{i,B} \boldsymbol{s}$ they compute $u^{-(t-1)i} \prod_{\substack{h \in B \\ h \neq i}} \frac{\alpha_h}{\alpha_{h-i}} \boldsymbol{s}$. The manner in which they compute this is $u^{-(t-1)i}(\frac{\alpha_{**}}{\alpha_*}(\cdots(\frac{\alpha_{**}}{\alpha_*}\boldsymbol{s})\cdots))$ (we are using $**$ and $*$ to represent arbitrary indices).

Algorithm 3 *Algorithm to compute* $y_{i,B} \boldsymbol{s}$.

Input=$(i, \boldsymbol{s}, B)$.
Output=$\boldsymbol{z} = y_{i,B} \boldsymbol{s}$.

$\boldsymbol{z} = \boldsymbol{s}$
$j = 0$;
for $h \in B, \ h \neq i$
$a = h - i \ mod \ q$
$\tau = \frac{h}{h-i} \ mod \ q$
$\boldsymbol{z} = \frac{\alpha_{a\tau}}{\alpha_a} \boldsymbol{s}$
$\boldsymbol{z} = u^{-(t-1)i} \boldsymbol{z}$

The last computation a shareholder will have to perform is the modular exponentiation. That is, once the shareholder has computed $\boldsymbol{z} = y_{i,B}\boldsymbol{s}_i$, to complete the computation of the partial signature the shareholder will need to compute $m^{z_0} \bmod N$. To gauge the cost of this exponentiation we need to bound the size of z_0. For $\boldsymbol{z} = [z_0, \ldots, z_{q-1}]$ define $||\boldsymbol{z}|| = max_i|z_i|$. We first show that:

Theorem 1. *If $\boldsymbol{b} = u^a\boldsymbol{s}$, then $||\boldsymbol{b}|| \leq 2||\boldsymbol{s}||$.*

Proof. Recall by Algorithm 1 that $u^a\boldsymbol{s} = (s_{-a}-s_{-a-1}, s_{1-a}-s_{-a-1}, \ldots, s_{q-2-a}-s_{-a-1})$. Therefore

$$\begin{aligned}||b|| &\leq \max\{|s_{-a} - s_{-a-1}|, |s_{1-a} - s_{-a-1}|, \ldots, |s_{q-2-a} - s_{-a-1}|\} \\ &\leq \max\{|s_{-a}| + |s_{-a-1}|, |s_{1-a}| + |s_{-a-1}|, \ldots, |s_{q-2-a}| + |s_{-a-1}|\} \\ &\leq 2||\boldsymbol{s}||.\end{aligned}$$

Theorem 2. *If $\boldsymbol{b} = \frac{\alpha_{a\tau}}{\alpha_a}\boldsymbol{s}$ then $||\boldsymbol{b}|| \leq 2\tau \cdot ||\boldsymbol{s}|| \leq 2q \cdot ||\boldsymbol{s}||$.*

Proof. Suppose $\boldsymbol{b} = \frac{\alpha_{a\tau}}{\alpha_a}\boldsymbol{s}$, then $\boldsymbol{b} = 1 + u^a + u^{2a} + \cdots + u^{a(\tau-1)}\boldsymbol{s}$. By Algorithm 2, $b_{ja} = s_{ja} + \sum_{x=0}^{\tau-1} s_{(j-x)a} - \sum_{x=0}^{\tau-1} s_{q-a-1-xa}$ [3](for a more complete argument we refer the reader to [15]). Thus $||\boldsymbol{b}|| \leq 2\tau \cdot ||\boldsymbol{s}|| \leq 2q \cdot ||\boldsymbol{s}||$.

Theorem 3. *If $\boldsymbol{z} = y_{i,B}\boldsymbol{s}$ where $\boldsymbol{s}$ represents the share of the private key distributed to the participant by the dealer, then $||\boldsymbol{z}|| \leq 2(2q)^{t-1}\phi(N)$*

Proof.

$$\begin{aligned}\boldsymbol{z} = y_{i,B}\boldsymbol{s} &= \prod_{\substack{h\in B \\ h\neq i}} \frac{0-\alpha_h}{\alpha_i - \alpha_h}\boldsymbol{s} \\ &= u^{-(t-1)i} \prod_{\substack{h\in B \\ h\neq i}} \frac{\alpha_h}{\alpha_{h-i}}\boldsymbol{s} \\ &= u^{-(t-1)i}\left(\frac{\alpha_{h_{t-1}}}{\alpha_{h_{t-1}-i}}\left(\frac{\alpha_{h_{t-1}}}{\alpha_{h_{t-1}-i}}\left(\cdots\left(\frac{\alpha_{h_1}}{\alpha_{h_1-i}}\boldsymbol{s}\right)\cdots\right)\right)\right)\end{aligned}$$

Therefore $||\boldsymbol{z}|| \leq 2(2q)^{t-1}||s|| \leq 2(2q)^{t-1}\phi(N)$.

Thus the number of bits required to represent the maximum z_j of $\boldsymbol{z} = y_{i,B}\boldsymbol{s}$ is approximately $\log_2(2(2q)^{t-1}\phi(N)) = t\log_2 q + \log_2 \phi(N)$. Consequently the time to compute $m^{z_0} \bmod N$ can be bounded by $O((t\log_2 q + \log_2(\phi(N))(\log_2 N)^2)$.

To demonstrate the performance of the DF scheme, we implemented a simulation of the shareholder computations of the DF scheme as well as the RSA exponentiation using Montgomery multiplication. All benchmarks were made on a Toshiba 380 MHz laptop. We were interested in making relevant computations,

[3] All arithmetic performed on indices is mod q. Also for simplicity we have introduced a dummy term $s_{q-1} = 0$.

so we utilized 1024 RSA keys. We simulated the shareholder computations for various thresholds: 6 out of 100, 15 out of 16, 31 out of 32, 51 out of 100, and 99 out of 100. The choices were made to reflect possible applications from fault-tolerant computing to group signatures. Our choice of t and n vary, we viewed that these inputs (t, n) are representative of possible thresholds, for example a threshold 51 out of 100 could be seen in a group signature scheme implemented by a legislature like the U.S. Senate (see [7]). To better understand how well the scheme is performing, we simulated the shareholder computations that a shareholder would make under Shoup's Practical Threshold Signature scheme. We limited the calculations to solely those that would be required to perform threshold sharing (not to achieve robustness nor the proof of correctness).That is, we benchmarked the computation $x^{2\Delta s_i} \bmod N$ (which is x_i). Such a computation represents the minimal amount of computations that would be required by a shareholder to compute a partial RSA signature in a threshold RSA signature scheme which avoids using the Lenstra constant (see comment in section 1). Other threshold schemes which avoid using the Lenstra constant include [11,17]. The Practical Threshold Signature shareholder will need to compute $x^{2\Delta s_i}$. We note that the computation of $temp = x^{2s_i}$ is comparable to an RSA exponentiation. Subsequently we raise this to the Δ power. Here $\Delta = n!$. Rather than computing $n!$, we efficiently compute $temp^{\Delta}$ as $(\cdots((temp^n)^{n-1})^{n-2}\cdots)$. We use the same multiplication function (Montgomery) and RSA exponentiation function that we used in the RSA benchmark and in the DF benchmark. The result of the benchmarks are displayed in the following table. The results in the table represent the amount of time for a single shareholder in the threshold scheme to perform their necessary computations when they participate in a signature generation. This data was constructed when simulating only the computations required and do not include any effects due to the network communication that is needed. Each benchmark is compared to the original RSA computation to illustrate the effect of the extra threshold computations required. We do so by taking the ratio of the shareholder's benchmark to the RSA benchmark. The basis of our comparison is the 1024 bit RSA signature which we benchmarked at a time of 445 milliseconds.

t	n	DF ms.	Ratio DF to RSA	Shoup ms.	Ratio Shoup to RSA
15	16	449.9	101.10 %	470.6	105.75 %
31	32	461.3	103.66 %	509.4	114.47 %
6	100	458.0	102.90%	704.4	158.2 %
51	100	504.7	113.4 %	707.6	159.0 %
99	100	533.4	119.8 %	704.4	158.2 %

In all cases q was selected to be the smallest prime to exceed n. Regarding the benchmarks, we do see that the DF scheme has out performed Shoup's scheme. Further there is a widening difference between the two schemes as n grows. Of course for large t we would expect the performance of the DF scheme to approach the performance of Shoup's scheme. (Due to the estimate on the size of $||\boldsymbol{z}||$ as given in Theorem 3.We know that the most time consuming

computation is the modular exponentiation which was estimated at $O((t \log_2 q + \log \phi(N))(\log_2 N)^2)$. Thus t should heavily influence this time. But this estimate was based on the assumption that $\boldsymbol{z} = y_{i,B}\boldsymbol{s}$ will possess coordinates which has grown to the bound given by Theorem 3.As we will soon see that is not the case.

Recall that $y_{i,B} = u^{-(t-1)i} \frac{\alpha_{i_1}}{\alpha_{i_1-i}} \cdot \frac{\alpha_{i_2}}{\alpha_{i_2-i}} \cdot \cdots \cdot \frac{\alpha_{i_{t-1}}}{\alpha_{i_{t-1}-i}}$. There are $t-1$ many distinct i_j's, and there are $t-1$ many distinct $i_j - i$'s. Thus by rearranging the ratios we should find a significant number of cancellations. Especially in the case where t is close to q. For example, in the case of the 99 out of 100 threshold, q was 101, so almost all the ratios should cancel. This implies that we have one (or two) ratios $\frac{\alpha_{a\tau}}{\alpha_a}$ to contend with, as well as implies that $||y_{i,B}\boldsymbol{s}||$ should be the same size as a RSA private key.

6 Some Enhancements

In the case where both t is large and $q - t$ is large, for example a 51 out of 100 threshold, the following idea may provide further reductions. If we do not achieve a significant number of cancellations by rearranging the ratios, we could use the relation $\alpha_{-x} = \frac{u^{-x}-1}{u-1} = -u^{-x}\alpha_x$ to search for more cancellations. Lastly we note that when one applies a ratio $\frac{\alpha_{a\tau}}{\alpha_a}$ to $\boldsymbol{s}$ the size of the output coordinates are affected by the size of τ (see Theorem 2). Thus if one considers those remaining ratios (after cancellations have taken place) one could rearrange the ratios to search for a collection of small "τ". One does not necessarily need to perform this last step to reap the benefit, for the fact that this step can be done implies that our bound on $||y_{i,B}\boldsymbol{s}||$ has been vastly overstated. Consequently the complexity estimate of the time to perform the exponentiation $m^{z_0} \bmod N$ has been overstated. As a side result, we experimented with the number of cancellations, as well as searched for pairing ratios so that τ was ± 2, and found that on a consistent basis that nearly 65% of the ratios fell into these categories.

We developed a second implementation, for which, before the shareholder computed $y_{i,B}\boldsymbol{s}$, they searched for cancellations as well as pairing alpha ratios so that "τ" was ± 2. The table below compares this "improved" implementation of the DF scheme to the previous DF scheme (we represent this "improved scheme as DF$^+$).

t	n	DF ms.	Ratio DF to RSA	DF$^+$ ms.	Ratio DF$^+$ to RSA
15	16	449.9	101.10 %	450.6	101.25 %
31	32	461.3	103.66 %	452.6	101.72 %
6	100	458.0	102.90 %	456.6	102.60 %
51	100	504.7	113.4 %	472.6	106.21 %
99	100	533.4	119.8 %	453.3	101.86 %

Of course for small t, we see little improvement when searching for cancellations. But our results do show that as t grows we do see significant improvement, in particular, when t is approximately n.

The most important comparison is the comparison of resources required for a shareholder to perform the necessary operations for it to participate in the threshold scheme. Obviously if the shareholders do not know in advance, which shareholders will participate in the signature generation then the Practical Threshold Signature scheme is the most efficient scheme. However, if the shareholders have knowledge of which shareholders will participate in the signature generation then we have evidence that computationally the DF scheme is more efficient than Practical Threshold Signature scheme. This does come at a cost of memory.

We could also compare the amount of work required by the combiner for each of the two schemes. We have not implemented the work, but the combiner of the DF scheme should require approximately the same amount of computations as the Practical Threshold Signature scheme, even if a proof of correctness is required.

Lastly, we could compare the amount of distributor work for each of the schemes. In this case, it is clear that the distributor of the DF scheme will be required to perform significantly more work than the Practical Threshold Signature scheme, including generating more random elements and performing more computations. This additional work required by the distributor of the DF scheme is due to the expanded shares in the DF scheme. However, the distributor should be characterized as an entity with a significant amount of resources whose computations will be performed once.

Regarding the expanded shares, in [9], it was shown that the share size of the DF scheme could be reduced by one-half. Possible avenues to consider when trying to achieve a reduction in share size: look for linear dependencies between subshares (this is how [9] achieved the reduction) or search for a ring $\mathcal{R}$ where one may find n units whose difference are as well, units a problem related to the *Lenstra constant*, for which one can define a efficient scalar arithmetic on $\mathcal{K}^X$ without requiring X to be too large. At Crypto 2002, Cramer and Fehr [2] have generated a group independent threshold scheme which is significantly more efficient (in terms of share size)[4] than the DF scheme. The DF scheme will be, in terms of computational complexity, more efficient than this scheme (this assumes that shareholders know who belong to B).

7 Conclusion

Our work provides tangible results that indicate the performance of the DF scheme is not a bottleneck and can be more efficient than the Shoup Practical Threshold Signature scheme and schemes comparable to it under certain conditions. What remains as the bottleneck, is the expanded shares. We have pointed out that this comparison was based on the assumption that shareholders know who belong to B. If such an assumption cannot be made, then the DF scheme is such that the shareholder computations degenerate to the point that they need

[4] The share expansion of the threshold scheme in [2] was $O(n \log n)$ whereas the share expansion of the DF scheme is $O(n^2)$.

to compute $O(n)$ RSA (partial) signatures. Thus if no interaction is allowed between shareholders, the Practical Threshold Signature scheme is vastly more efficient than the DF scheme. Lastly, we have provided further enhancements to improve the performance of the DF scheme.

References

1. D. Boneh and M. Franklin. "Efficient Generation of Shared RSA Keys". In *Advances of Cryptology- CRYPTO 1997*, p. 425-439.
2. R. Cramer and S. Fehr. "Optimal Black-Box Secret Sharing over Arbitrary Abelian Groups". In *Advances of Cryptology- Crypto 2002*.
3. I. Damgard and M. Koprowski. "Practical Threshold RSA Signatures without a Trusted Dealer". In *Advances of Cryptology- EUROCRYPT 2001*, p. 152-165
4. A. De Santis, Y. Desmedt, Y. Frankel, and M. Yung. "How to share a function". In *Proceedings of the twenty-sixth annual ACM Symp. Theory of Computing (STOC)*, pp. 522-533, 1994.
5. Y. Desmedt and Y. Frankel. Threshold Cryptosystems In *Advances of Cryptology-Crypto '89*, pages 307-315, 1989.
6. Y. Desmedt and Y. Frankel. Shared generation of authenticators and signatures. In *Advances of Cryptology- Crypto '91*, 1991.
7. Y. Desmedt and B. King, "Verifiable Democracy - a protocol to secure an electronic legislature", *Electronic Government, First International Conference, EGOV 2002*, Lecture Notes in Computer Science 2456 Springer 2002, pg 400-403.
8. Y. Desmedt and Y. Frankel. "Homomorphic zero-knowledge threshold schemes over any finite abelian group". In *Siam J. Disc. Math. vol 7, no. 4* pp. 667-679, SIAM, 1994.
9. Y. Desmedt, B. King, W. Kishimoto, and K. Kurosawa, "A comment on the efficiency of secret sharing scheme over any finite abelian group", In *Australasian Conference on Information Security and Privacy 1998*, LNCS 1438, 1998, 391-402.
10. Y. Frankel, P. Gemmel, P. Mackenzie, and M. Yung. "Proactive RSA". In *Advances of Cryptology-Crypto '97*, 1997, LNCS 1294, Springer Verlag, 1997, p. 440-454.
11. Y. Frankel, P. Gemmel, P. Mackenzie, and M. Yung. "Optimal-Resilience Proactive Public-key Cryptosystems". In *Proc. 38th FOCS*, IEEE, 1997, p. 384-393.
12. Y. Frankel, P. MacKenzie and M. Yung. "Robust Efficient Distributed RSA-Key Generation". In *STOC 1998*, p. 663-672.
13. R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. "Robust and efficient sharing of RSA functions". In *Advances of Cryptology-Crypto '96,* LNCS 1109, Springer Verlag, 1996, p. 157-172.
14. B. King. Algorithms to speed up computations in threshold RSA, In *Australasian Conference on Information Security and Privacy 2000*, p. 443-456.
15. B. King. "Improved Methods to Perform Threshold RSA". In *Advances in Cryptology - ASIACRYPT 2000*, LNCS 1976, Springer Verlag, 2000,p. 359-372 .
16. H.L. Keng. *Introduction to Number Theory.* Springer Verlag, NY 1982
17. T.Rabin. A Simplified Approach to threshold and proactive RSA. In *Advances of Cryptology- Crypto '98*, 1998
18. R. Rivest, A. Shamir, and L. Adelman, A method for obtaining digital signatures and public key cryptosystems, *Comm. ACM,* 21(1978), pp 294-299.
19. A. Shamir, How to share a secret, *Comm. ACM*, 22(1979), pp 612-613.
20. V. Shoup. "Practical Threshold Signatures". In *Advances in Cryptology - EUROCRYPT 2000*, LNCS 1807, Springer Verlag 2000, p. 207-220.

GBD Threshold Cryptography with an Application to RSA Key Recovery*

Chris Steketee[1], Jaimee Brown[2], Juan M. González Nieto[2], and Paul Montague[3]

[1] Advanced Computing Research Centre, University of South Australia
Chris.Steketee@cs.unisa.edu.au
[2] Information Security Institute, Queensland University of Technology
{j2.brown,j.gonzaleznieto}@qut.edu.au
[3] Motorola Australia
Paul.Montague@motorola.com

Abstract. We present protocols for threshold decryption and threshold key generation in the GBD public-key cryptosystem in the "honest-but-curious" setting. These allow GBD computations to be performed in a distributed manner during both key generation and decryption, without revealing the private key to any party. GBD threshold decryption is similar to El-Gamal threshold decryption. GBD threshold key generation is based on adaptations of protocols for RSA key generation by Boneh and Franklin, and Catalano et al, and includes a new protocol for efficiently computing the inverse of a shared secret modulo another shared secret. We also show an application of GBD threshold cryptography to RSA key recovery, and point out two open problems in this application.

1 Introduction

This paper applies techniques of threshold cryptography to the GBD public-key cryptosystem of González et al [1,2]. We develop both GBD threshold decryption and GBD threshold key generation, and we consider finally an application to key recovery for the RSA cryptosystem.

Threshold cryptography [3] shares the private key amongst a number, ℓ, of players, with a threshold, $t < \ell$, such that any subset of $t + 1$ or more players can compute (decrypt or sign) with the private key, but no subset of t or fewer players can do so (or indeed deduce any information about either the plaintext or the private key). It has been widely studied since [4] developed the first practical scheme for ElGamal and [5] for RSA.

Initially these schemes offered protection only against passive adversaries, often referred to as the "honest-but-curious" model of security. In this model, the corruption of up to t players does not compromise the security of the private key or allow an adversary to operate with the private key, provided that the

* This work was partially funded by Australian Research Council Linkage Project Grant LP0347128.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 394–405, 2005.

adversary is limited to viewing the state and communication messages of the corrupted players. A subsequent development was the addition of *robustness*, providing protection also against active adversaries that can alter the behaviour of corrupted players in arbitrary ways, and *pro-active security*, which changes the key-shares (but not the key) periodically in order to reduce the time available for an adversary to corrupt $t+1$ players.

Threshold methods have been applied also to the generation of private keys, thereby eliminating the need for a trusted dealer to generate the keys, compute the shares, and communicate these securely to all players. In *threshold key generation*, the players jointly generate the shares of the private key without that private key becoming known to any party. This was achieved by [6] for ElGamal, but was more difficult for RSA, with the first practical solution being produced by Boneh and Franklin [7], and a robust version by [8].

In this paper, we apply the methods of threshold cryptography to the GBD cryptosystem. The motivation for this work is two-fold. Firstly, we wish to show that applications using GBD can benefit from threshold methods. Secondly, we demonstrate how, by using GBD as a "master" cryptosystem whose security is enhanced by threshold cryptography, it is possible for individuals in an organisation to generate RSA public and private keys that are then used in standard fashion, but with the added capability of allowing the RSA private key to be recovered by a sufficiently large subset of a group of designated key recovery entities.

Communication and Security Model For both threshold decryption and threshold key generation, we denote the number of players by ℓ and the threshold by t. The protocols for key generation require $t \leq \lfloor (\ell - 1)/2 \rfloor$.

In common with other work in this area, we assume a private communication channel between each pair of players, and an authenticated broadcast channel[4].

We restrict ourselves to the "honest-but-curious" model, in which players may collude to pool their information in an attempt to discover secrets, but all players follow the protocol. Moreover, we use a static model, in which the adversary chooses the set of players to corrupt before the computation begins.

2 Summary of the GBD Cryptosystem

González, Boyd and Dawson [1] presented a semantically secure public key cryptosystem which operates in $\mathbb{Z}_P^*$, the multiplicative group of integers modulo a large prime P such that $P = 2N + 1$, where $N = Q_0 Q_1$ and Q_0, Q_1 are also prime. The security of the cryptosystem is based on the difficulty of determining whether an element $x \in \mathbb{Z}_P^*$ is a member of the subgroup G_{Q_i} of order Q_i (for $i = 0, 1$) given P and two elements g_0, g_1 of order Q_0, Q_1 respectively. The authors conjecture that the best attack against the GBD scheme is factoring N,

[4] These may be implemented using standard cryptographic protocols for privacy and authentication.

hence the primes need to be large enough such that factoring N is hard. In this way GBD is similar to RSA since the key lengths need to be of similar size.

For the following description of the GBD constituent algorithms, we denote G_i as the proper subgroup of $\mathbb{Z}_P^*$ of order i. All operations are assumed to be reduced modulo P unless otherwise instructed.

Key Generation $\mathcal{G}(1^k)$: Input security parameter k

1. Generate the modulus P such that $P = 2N + 1$, where $N = Q_0Q_1$ and Q_0, Q_1 are each random primes of binary size k.
2. Select elements g_0, g_1 of order Q_0, Q_1 respectively.
3. Compute $\alpha_i \equiv Q_{1-i}(Q_{1-i}^{-1} \bmod Q_i)$.
4. Output the public key $pk = (P, g_0, g_1)$ and the secret key $sk = (\alpha_0, \alpha_1)$.

Encryption $\mathcal{E}(pk, m)$: Input message $m \in G_N$, public key $pk = (P, g_0, g_1)$.

1. Choose two integers r_0, r_1 uniformly at random in $\mathbb{Z}_N$.
2. Compute $v_i = g_i^{r_i}$, an element of G_{Q_i} for $i = 0, 1$.
3. Compute $c_i = mv_{1-i}$, an element of G_N for $i = 0, 1$.
4. Output the ciphertext $c = (c_0, c_1)$.

It can be shown that the unique projection of any element y of G_N onto G_{Q_i} is given by $y_i = y^{\alpha_i}$, which allows us to decrypt an encrypted message, as shown below.

Decryption $\mathcal{D}(sk, c)$: Input secret key $sk = (\alpha_0, \alpha_1)$, ciphertext $c = (c_0, c_1)$

1. Compute $m_i = c_i^{\alpha_i}$ for $i = 0, 1$.
2. Calculate $m = m_0m_1$.
3. Output m.

It can be seen that encryption takes two modular exponentiations and two modular multiplications, and decryption requires computing the product of two exponentiations. This decryption can actually be implemented using algorithms that are far more efficient than performing the two exponentiations separately. Brown *et al.* [9] examine the implementation of the GBD cryptosystem, and show that the efficiency is comparable to that of a semantically secure ElGamal implementation for equal key lengths.

In fact, the decryption algorithm can be simplified in such a way that it requires only one exponentiation and a division instead of two exponentiations.

Decryption $\mathcal{D}'(sk, c)$: Input secret key $sk = (\alpha_0)$, ciphertext $c = (c_0, c_1)$

1. Compute $u = c_0/c_1$
2. Calculate $m = c_1 * u^{\alpha_0}$.

Here, exponentiating $u = g_1^{r_1}g_0^{-r_0}$ $(\in G_N)$ to α_0 projects it onto the subgroup G_{Q_0}, giving $g_0^{-r_0}$. Hence, $c_1u^{\alpha_0} = mg_0^{r_0}g_0^{-r_0} = m$ and decryption is correct.

3 GBD Threshold Decryption

The task of distributively decrypting a GBD ciphertext is very similar to that of ElGamal. As we have seen in section 2, a simplified GBD decryption requires a division of the ciphertext components and one exponentiation using the private key exponent α_0.

We achieve *t-out-of-ℓ* threshold GBD decryption using a polynomial sharing of the secret α_0 by choosing a polynomial $f(x)$ of degree t in a field such that $f(0) = \alpha_0$. Each player i is given the secret share $K_i = f(x_i)$, where x_i is chosen to be i. Decryption of a ciphertext is done by reconstructing the secret α_0 implicitly in the exponent from $(t+1)$ secret shares using Lagrange interpolation. This means that the secret is never actually explicitly reconstructed and therefore remains secret to all players.

Given a ciphertext $c = (c_0, c_1)$, a subset T of $t+1$ players can collaboratively compute the decryption as follows. Each player i in T first computes $u = c_0/c_1$, then

$$a_i = K_i \prod_{j \in T, j \neq i} \frac{(0 - x_j)}{(x_i - x_j)}$$

and finally $u'_i = u^{a_i}$. If these u'_i values are then sent to some combiner, or alternatively broadcast to other players, the message can be computed by the combiner, or each player in the latter case by:

$$\begin{aligned} c_1 \times \prod_{i \in T} u'_i &= c_1 . u^{\sum_{i \in T} a_i} \\ &= c_1 . u^{\alpha_0} = m \end{aligned}$$

For our application to key recovery to be detailed in section 5, we also need to perform GBD threshold projection. As stated in section 2, any element $y \in G_N$ can be uniquely projected onto the subgroup elements $y_0 \in G_{Q_0}$ and $y_1 \in G_{Q_1}$ such that $y = y_0 y_1$. To compute these projections, firstly one computes $y_0 = y^{\alpha_0} \mod P$ and then $y_1 = y/y_0 \mod P$. When α_0 is shared polynomially as described above, each player can compute y^{a_i} and make this value known to the other t players. The calculation of y_0 can then be performed simply by computing the product of these y^{a_i} values. The calculation of y_1 can then be performed by each player.

4 GBD Threshold Key Generation

The value of threshold decryption is reduced if it depends on a trusted dealer to generate the initial keys and securely distribute the private key shares: since the trusted dealer knows all the secrets, it introduces a single point of vulnerability. This has motivated the development of key generation schemes which generate the secret shares as a shared computation, without the secrets ever being reconstructed.

The GBD threshold key generation described here is based on the work of Boneh and Franklin [7] for shared generation of the modulus $N = Q_0Q_1$, plus a new protocol to invert a shared value modulo a shared value, based on Catalano et al [10].

4.1 Building Blocks for Computations on Shared Values

We use well-known secure techniques — eg [11] — for polynomial sharing of values and for performing shared computations on shared values. These are not detailed here, but may be found in the full version of the paper [12]. A result by Canetti [13] shows that low-level protocols for secure shared computations may be composed non-concurrently to provide a higher-level protocol which is also secure. Like other work in this field, the security of our solution depends on this result.

4.2 Shared Generation of the GBD Modulus

Boneh and Franklin, in a landmark paper [7], present protocols for the shared generation of shared RSA keys, consisting of two parts: firstly the generation of an RSA modulus $n = pq$ where p and q are shared primes, and secondly the generation of the private exponent in shared form. For GBD, we adapt the first part to generate the GBD modulus $N = Q_0Q_1$, adding a test that $P = 2N+1$ is prime. The second part is specific to RSA; our equivalent for GBD, the generation of the GBD private key, is described in 4.3.

The shared generation of the GBD modulus obtained by adapting [7] is summarised in Figure 1. Details of the shared computation steps 1b, 2 and 5 may be found in [7]. That paper also contains proofs of the security of each step of the protocol. Since the only change we have made to the protocol is the addition of step 3, which operates on a public value, these proofs apply also to our protocol.

The successful completion of this protocol results in an additive sharing over the integers of Q_0 and Q_1, ie player P_s has shares $Q_{i,s}$ such that $Q_i = \sum_{s=1}^{\ell} Q_{i,s}$. All players also know the value of $N = Q_0Q_1$ and thence $P = 2N + 1$.

The reliance on a biprimality test leads to a larger number of iterations than would be the case for testing each Q_i for primality in isolation. [7] quotes an expected number of iterations of 484 to generate a 1024-bit RSA modulus, versus 44 for normal modulus generation (both assuming trial divisions up to $B_1 = 8103$ in step 1b). A subsequent implementation paper for RSA [14] describes further optimisations, including a technique of distributed sieving and implementation optimisations based on concurrency. The paper cites total generation times around 1.5 minutes for a 1024-bit modulus on a local network for three players ($\ell = 3$), and 5.6 minutes for $\ell = 5$, using 333 MHz Pentium II systems running Solaris 2.5.1, a 10 MB Ethernet network, and SSL to protect the privacy of communications. These times are completely dominated by modulus generation; the contribution of the subsequent private key generation step is insignificant.

Input: security parameter k.

1. The following steps are performed for Q_0 first and then repeated for Q_1:
   (a) Each player P_s chooses a secret k-bit integer $Q_{i,s}$. The value $\sum_s Q_{i,s}$ is a shared candidate for Q_i.
   (b) The players perform a shared trial division of Q_i by small primes less than some bound B_1. If a factor is found, repeat from step 1a.
2. The players perform a shared computation to reveal the value of $N = Q_0 Q_1$.
3. One of the players performs a primality test on $P = 2N + 1$ as a local computation. If this fails, repeat from step 1.
4. One or more players perform further trial divisions on N for small primes in the range $[B_1, B_2]$ for some bound B_2. If a factor is found, repeat from step 1.
5. The players perform a shared computation to test that N is the product of two primes; if this fails, repeat from step 1.

Fig. 1. Shared Generation of the GBD Modulus

A worst-case estimate for shared GBD modulus generation is obtained by multiplying these times by the expected number of iterations required to find a prime P, about 350 iterations for P of size 1024 bits, potentially giving a GBD modulus generation time of many hours. In practice, and with further optimisations, these times would be reduced significantly. Even so, GBD threshold key generation using this protocol may be suitable only for applications where it is required infrequently — for example the generation of the GBD keys for the RSA key recovery example (section 5).

Note also that [11] presents a shared primality test as a possibly more efficient replacement for the biprimality test of [7], but we are not aware of any implementations or experimental results.

4.3 GBD Private Key Calculation

The GBD private key consists of the projection exponent α_0, and is computed from the primes Q_i as $\alpha_0 = Q_1(Q_1^{-1} \bmod Q_0)$. For GBD threshold key generation, the input values Q_i are shared between the players, and we require that the output value α_0 is also shared, to enable GBD threshold decryption and projection. Shares of α_0 may be kept modulo $N = Q_0 Q_1$, since they will be used to exponentiate values $y \in G_N$, which satisfy $y^N \equiv 1 \pmod{P}$. We also require that, in the "honest-but-curious" security model, the shared computation reveals no information to an adversary that it could not obtain from the public key alone.

The secure computation of the private key will be performed as a shared inversion operation followed by a shared multiplication. The multiplication is a standard protocol (section 4.1). We describe the inversion below. Note that the computations require that Q_0 and Q_1 be converted from additive sharing

over the integers to polynomial sharing in $\mathbb{Z}_N$: this is achieved using a standard protocol (section 4.1).

The inspiration for our inversion protocol is that of Catalano et al [10] which was developed for the case where the value to be inverted is known and the modulus shared, and in which computations are done in the integers. We extend the protocol for our case, in which both values are shared, and specialise it for this application by performing computations modulo N. The ability to run this protocol modulo N has attractive consequences: the shares of α_0 have a smaller size, and the security proof is straightforward[5].

Our protocol is based on the shared computation of two random linear combinations of Q_0 and Q_1 modulo N:

$$F = (RQ_0 + SQ_1) \bmod N$$
$$E = (TQ_0 + UQ_1) \bmod N$$

where R, S, T and U are independent shared random values in $\mathbb{Z}_N$. The values of F and E are then revealed to all players, and each player performs a local computation, using the extended GCD algorithm to compute integers a, b and d satisfying $aF + bE = d$. Substituting for F and E, we obtain:

$$\begin{aligned}
(aR + bT)Q_0 + (aS + bU)Q_1 &\equiv d \pmod N \\
e(aR + bT)Q_0 + e(aS + bU)Q_1 &\equiv 1 \pmod N \text{ where } e \equiv d^{-1} \pmod N \\
e(aS + bU)Q_1 &\equiv 1 \pmod{Q_0} \text{ since } Q_0 \text{ is a divisor of } N \\
\text{that is, } e(aS + bU) &\equiv Q_1^{-1} \pmod{Q_0} \text{ which is the desired result.}
\end{aligned}$$

Each player P_s therefore computes its share of C, the inverse of Q_1, from its shares of S and U as $C_s = e(aS_s + bU_s) \bmod N$. The probability that the inverse $d^{-1} \bmod N$ does not exist is $O(2^{-k})$, which is negligible for realistic values of N. The complete protocol is given in Figure 2.

Security of this Protocol The full version of the paper [12] includes a proof that the protocol is private. In this published version, we confine ourselves to a very brief outline. Firstly, the polynomial sharing of Q_0, Q_1, R, S, T and U is obtained using standard techniques that are known not to reveal any information to an adversary having access to t or fewer shares. Secondly, the proof shows that the values F, E, F_s and E_s are, independently, uniformly distributed over $\mathbb{Z}_N$. Therefore, publishing F and E reveals no information about Q_0 and Q_1, and publishing F_s and E_s reveals no information about $Q_{0,s}$ and $Q_{1,s}$. Thirdly, it shows that the shares C_s are uniformly distributed independently of F_s and E_s, and t or fewer shares reveal no information about the inverse C.

4.4 Computing the GBD Public Key

The public key consists of (P, g_0, g_1) where g_0 and g_1 are randomly-chosen elements of the GBD subgroups G_{Q_0} and G_{Q_1}. These may be computed most

[5] An extension of the protocol for the general case where computations must be done in the integers is also feasible, but not required for this application.

1. Polynomially share a random value $R \in \mathbb{Z}_N$, resulting in shares R_s (section 4.1).
2. Similarly share random values S, T, and $U \in \mathbb{Z}_N$, resulting in shares S_s, T_s and U_s.
3. Each player broadcasts the value $F_s = (R_s Q_{0,s} + S_s Q_{1,s}) \bmod N$. These values are shares of a random polynomial $F(x)$ of degree $2t$ which has $F(0) = F = (RQ_0 + SQ_1) \bmod N$.
4. Each player broadcasts the value $E_s = (T_s Q_{0,s} + U_s Q_{1,s}) \bmod N$. These values are shares of a random polynomial $E(x)$ of degree $2t$ which has $E(0) = E = (TQ_0 + UQ_1) \bmod N$.
5. Each player uses Lagrange interpolation (requiring $2t + 1$ shares) to compute the values F and E.
6. Each player performs the extended GCD protocol to find a and b such that $aF + bE = d$, and computes $e \equiv d^{-1} \pmod N$.
7. Each player P_s computes its share of $C = Q_1^{-1} \bmod Q_0$ as $C_s = e(aS_s + bU_s) \bmod N$.

An efficient implementation would combine into one round the separate communication rounds implied in steps 1 to 2, and also the rounds in steps 3 to 4. This protocol therefore requires two rounds of communication.

Fig. 2. Shared Inversion Protocol

straightforwardly by choosing a random g in subgroup G_N, projecting g onto G_{Q_0} using $g_0 = g^{\alpha_0} \bmod P$, and computing $g_1 = g \cdot g_0^{-1} \bmod P$. g is most easily obtained by having one player, P_1 say, choose a random h in $[2, P-2]$, computing $g = h^2 \bmod P$ and broadcasting its value. Computing g_0 and g_1 is an application of GBD threshold projection (section 3) using the shares $\alpha_{0,s}$ of the private key as computed above.

5 Self-escrowed RSA

The notion of self-escrowed public keys is due to Paillier and Yung [15]. A public key encryption scheme $\mathcal{S} = \langle \mathcal{G}, \mathcal{E}, \mathcal{D} \rangle$ is said to be (perfectly) *self-escrowed* when there exists a master encryption scheme $\mathcal{S}' = \langle \mathcal{G}', \mathcal{E}', \mathcal{D}' \rangle$ and a master key pair (e', d') of $\mathcal{S}'$ such that all key pairs of $\mathcal{S}$ satisfy $e = \mathcal{E}'(e', d)$. In other words, a public key is self-escrowed if the public key is itself an encryption of the private key under some master cryptosystem. In the typical application of self-escrowed cryptosystems, a trusted authority known as the key recovery authority (KRA) owns the master key pair. System users employ $\mathcal{S}$ and generate their key pairs according to $\mathcal{G}$. Normally, these keys are then certified by a certification authority (CA). If for whatever reason a private key needs to be restored, e.g. a user loses her private key, the KRA can do so given the corresponding public key. The main advantage of using self-escrowed keys when compared with other proposals

(e.g. [16,17]) is that the only information that needs to be stored by the KRA is the master key. When the key recovery capability is to be enforced during certification of public keys, it may be required that users prove to the CA that the public keys submitted for certification are indeed self-escrowed, i.e. that they are valid keys from the range of $\mathcal{G}$.

In a self-escrowed public key infrastructure the compromise of the master key is likely to be catastrophic. The secrecy of all users' private keys relies on the secrecy of the master key. Furthermore, users must trust that the KRA will not misuse its key recovery capability. Because of this, it is desirable to distribute the key recovery function among a threshold of authorities.

Paillier and Yung [15] described a self-escrowed discrete log based cryptosystem (a variation of ElGamal's cryptosystem over groups of composite order). The master encryption scheme is a deterministic version of Paillier's encryption scheme [18], for which threshold decryption schemes are known [19,20]. In [21] Gonzalez *et al.* observed that the trap-door function upon which the GBD is built can be used as a trap-door to the prime factorisation of composite numbers of a certain form. This allows the generation of *self-escrowed* factorisation-based public keys (e.g. RSA keys). Now the master encryption is a deterministic version of GBD, for which there was no threshold decryption equivalent until now.

In what follows we apply the threshold GBD key generation and decryption of Sections 3 and 4 to the self-escrowed scheme of González *et al.* [21] for integer factorisation based cryptosystems. For simplicity we focus on self-escrowed RSA, noting that self-escrowed versions of other factorisation-based cryptosystems can be devised similarly.

Master Cryptosystem $\mathcal{S}'$: The master key generation algorithm $\mathcal{G}'$ is the t-out-of-ℓ threshold GBD key generation algorithm described in Sects. 4.2 and 4.3. Let the public key be $e' = (P, g_0, g_1)$, where $P = 2N+1$, and $N = Q_0Q_1$. The private key is $\alpha_0 = Q_1(Q_1^{-1} \mod Q_0)$, which is shared among the KRAs as described in Sect. 4.3. The encryption algorithm $\mathcal{E}'$ is a deterministic version of the GBD encryption algorithm. The message space consists of the elements of $G_{Q_0} \times G_{Q_1}$. On input a message $(m'_0, m'_1) \in G_{Q_0} \times G_{Q_1}$ and public key e', it outputs a ciphertext $c' = m'_0 m'_1$. It is easy to see that one-wayness is still based on the projection problem assumption. Given a ciphertext c', any subset of more than t KRAs can run the decryption algorithm $\mathcal{D}'$ to compute ${c'}^{\alpha_0} \mod P$, as shown in Sect. 3, and recover the plaintext message $(m'_0, m'_1) = ({c'}^{\alpha_0} \mod P, c'/m'_0 \mod P)$.

User Cryptosystem $\mathcal{S}$: The only difference with respect to plain RSA is that the key generation algorithm is modified to output the prime factors p and q lying in G_{Q_0} and G_{Q_1}. On input a master public key e', the user key generation $\mathcal{G}$ is as follows:

- Choose random r_0, r_1 in $\mathbb{Z}_{(P-1)/2}$ such that $p = {g_0}^{r_0} \mod P$ and $q = {g_1}^{r_1} \mod P$ are both distinct primes.
- Compute $n = pq$ in $\mathbb{Z}$.

– Choose $a \in \mathbb{Z}_{\phi(n)}$.
– Compute $d = a^{-1} \mod \phi(n)$.

The user public key is $e = (a, n)$ and the private key is d. The encryption algorithm $\mathcal{E}$, given a message $m \in \mathbb{Z}_n$ and public key $e = (a, n)$ outputs a ciphertext $c = m^a \mod n$. The decryption algorithm $\mathcal{D}$ computes the plaintext back as $m = c^d \mod n$.

Since for large P the density of prime numbers in the subgroups G_{Q_0}, G_{Q_1} can be considered the same as in the whole group $\mathbb{Z}_P^*$, the prime number theorem ensures that the amount of primes $(p, q) \in G_{Q_0} \times G_{Q_1}$ is large, and that therefore they can be found efficiently. The RSA moduli generated by $\mathcal{G}$ are ciphertexts of $\mathcal{S}'$ and therefore can be decrypted into its factors by any subset of at least $t + 1$ KRAs.

Key-length Doubling: As noted in [21], the self-escrowed RSA cryptosystem described above has an important shortcoming that reduces its usefulness. The effective security of RSA moduli n generated by $\mathcal{G}$ is not greater than the security of the integer N generated by $\mathcal{G}'$, while the typical length of n is double that of N. Note that the factors p and q of n are chosen randomly from the subgroups G_{Q_0} and G_{Q_1}, and hence their typical length will be approximately $|P|$. It is an open question whether it is possible to modify $\mathcal{G}$ to efficiently generate factors of significantly shorter lengths.

We note that this key-length doubling problem is a consequence of the fact that the GBD modulus P is publicly known. Any party can attempt on its own to recover the private key of all self-escrowed RSA public keys by factoring N, which is much easier than attempting factoring an individual RSA modulus. If P were kept secret, then an attacker would have to fall back to factoring individual RSA moduli employing standard factoring algorithms. Applications where this is the case, i.e. where the modulus P is never made publicly available, are conceivable. For example consider the situation where a group of parties needs to generate a large set of shared RSA keys for themselves, but do have severe storage constraints that do not allow them to store all the corresponding private keys. This may well be the case when using secure hardware tokens. A solution to this can be achieved by modifying the above master and user cryptosystems. The group of parties would set up a (modified) master cryptosystem, which they themselves use to generate a set of (shared) self-escrowed RSA keys. Only the shares of the master key pair need to be kept secret in the tokens, whereas the shares of RSA private keys can be computed from the public keys, which do not require hardware protection. The modifications, which we leave as future work, needed to the master and user cryptosystem are as follows:

– The algorithm for the generation of the GBD modulus would need to be changed so that N is never revealed to any single party.
– Since P would be a value shared by the group of parties, the inversion algorithm in Section 4.3 would need to be extended to the case where computations are done over the integers — this appears doable in a way similar to the inversion protocol of Catalano *et al.* [10].

– The key-generation algorithm of the self-escrowed RSA cryptosystem would need to be distributed in a threshold manner. This would be the same algorithm as in plain threshold RSA, but complicated by the requirement that the factors must be chosen from G_{Q_0} and G_{Q_1}, with the modulus P being a shared value.

Finally, we note that even when the master public key is never exposed, a concern remains as to the security of RSA keys generated by $\mathcal{G}$ due to the sparsity of the distribution of primes from which the factors are chosen. Only a small fraction $O(\frac{1}{\sqrt{P}})$ of primes in $\mathbb{Z}_P$ is eligible to be chosen. It is an open problem to devise a factoring algorithm that takes advantage of the special form of these factors.

6 Conclusion and Further Work

Application to GBD of the techniques of threshold cryptography has been shown to be feasible, thus extending the applicability of GBD to those contexts where it is required to secure the private key information by sharing it between several servers. The threshold key generation protocol may require running times in the order of hours, which would restrict the circumstances in which it is practical. Experimental estimates of the time taken to generate the GBD modulus, and possible improvements in the protocol to reduce this time, are potential areas for future study, as is the development of an efficient robust version of the protocols.

We have demonstrated an application of GBD threshold cryptography to the problem of RSA key recovery by using a self-escrow method. This suffers from the drawback that it requires the RSA key to be double the normal length in order to maintain equivalent security. We are investigating ways to overcome this drawback. In addition there is an open problem of whether a factoring algorithm can be devised to take advantage of the special form of RSA modulus generated by this self-escrow method; clearly this would affect the practical utility of this approach to key recovery.

References

1. González-Nieto, J. M., C. Boyd, and E. Dawson. A public key cryptosystem based on a subgroup membership problem. In *ICICS 2001*, volume 2229 of *LNCS*, pages 352–363. Springer-Verlag, 2001.
2. González-Nieto, J. M., C. Boyd, and E. Dawson. A public key cryptosystem based on a subgroup membership problem. *Designs, Codes and Cryptography*, Accepted for publication 2004.
3. P. Gemmell. An introduction to threshold cryptography. *CryptoBytes*, 2(3):7–12, 1997.
4. Y. Desmedt and Y. Frankel. Threshold cryptosystems. In *Crypto '89*, volume 435 of *LNCS*, pages 307–315. Springer-Verlag, 1990.
5. A. De Santis, Y. Desmedt, Y. Frankel, and M. Yung. How to share a function securely. In *STOC 94*, pages 522–533, 1994.

6. T. Pedersen. A threshold cryptosystem without a trusted party. In *Eurocrypt '91*, volume 547 of *LNCS*, pages 522–526. Springer-Verlag, 1991.
7. D. Boneh and M. Franklin. Efficient generation of shared RSA keys. *J. ACM*, 48(4):702–722, 2001.
8. Y. Frankel, P. MacKenzie, and M. Yung. Robust efficient distributed RSA-key generation. In *STOC '98*, pages 663–672. ACM Press, 1998.
9. J. Brown, E. Dawson, and González-Nieto, J. M. Implementation of the GBD cryptosystem. In *Cryptographic Algorithms and their Uses*, pages 94–109. QUT Publications, 2004.
10. D. Catalano, R. Gennaro, and S. Halev. Computing inverses over a shared secret modulus. In *Eurocrypt 2000*, volume 1807 of *LNCS*, pages 190–206. Springer-Verlag, 2000.
11. J. Algesheimer, J. Camenisch, and V. Shoup. Efficient computation modulo a shared secret with application to the generation of shared safe-prime products. In *Crypto'02*, volume 2442 of *LNCS*, pages 417–432. Springer-Verlag, 2002.
12. C. Steketee, J. Brown, González Nieto, J., and P. Montague. GBD threshold cryptography with an application to RSA key recovery, 2005. http://eprints.qut.edu.au/.
13. R. Canetti. Security and composition of multiparty cryptographic protocols. *J. Cryptology*, 13(1):143–202, 2000.
14. M. Malkin, T. Wu, and D. Boneh. Experimenting with shared generation of RSA keys. In *SNDSS '99*, pages 43–56, 1999.
15. P. Paillier and M. Yung. Self-escrowed public-key infrastructures. In *ICISC '99*, LNCS, pages 249–261. Springer-Verlag, December 1999.
16. A. Young and M. Yung. Auto-recoverable and auto-certifiable cryptosystems. In *EUROCRYPT '98*, number 1403 in LNCS, pages 17–31. Springer-Verlag, 1998.
17. S. Micali. Fair public-key cryptosystems. In *Crypto '92*, volume 740 of *LNCS*, pages 113–128. Springer-Verlag, 1993.
18. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Eurocrypt '99*, number 1599 in LNCS, pages 223–238. Springer-Verlag, 1999.
19. P. Fouque, G. Poupard, and J. Stern. Sharing decryption in the context of voting or lotteries. In *FC '00*, volume 1962 of *LNCS*, pages 90–104. Springer-Verlag, 2001.
20. I. Damgard and M. Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In *PKC '01*, volume 1992 of *LNCS*, pages 119–136. Springer-Verlag, 2001.
21. González-Nieto, J. M., K. Viswanathan, C. Boyd, and E. Dawson. A self-escrowed integer factorisation based public key infrastructure. In *VII Spanish Meeting on Cryptology and Information Security*, pages 315–328, Oviedo, Spain, 2002. Universidad de Oviedo.

An $(n-t)$-out-of-n Threshold Ring Signature Scheme

Toshiyuki Isshiki[1] and Keisuke Tanaka[2]

[1] NEC Corporation,
1753, Shimonumabe, Nakahara-Ku, Kawasaki, Kanagawa, 211-8666, Japan
t-issiki@bx.jp.nec.com
[2] Dept. of Mathematical and Computing Sciences
Tokyo Institute of Technology,
W8-55, 2-12-1 Ookayama Meguro-ku, Tokyo 152-8552, Japan
keisuke@is.titech.ac.jp

Abstract. In CRYPTO2002, Bresson, Stern, and Szydlo proposed a threshold ring signature scheme. Their scheme uses the notion of *fair partition* and is provably secure in the random oracle model. Their scheme is efficient when the number t of signers is small compared with the number n of group members, i.e., $t = \mathcal{O}(\log n)$ (we call this scheme **BSS scheme**). However, it is inefficient when t is $\omega(\log n)$.
In this paper, we propose a new threshold ring signature scheme which is efficient when the number of signers is large compared with the number n of group members, i.e., when the number t of non-signers in the group members is small compared with n. This scheme is very efficient when $t = \mathcal{O}(\log n)$. This scheme has a kind of dual structure of BSS scheme which is inefficient when the number of signers is large compared with the number of group members. In order to construct our scheme, we modify the trap-door one-way permutations in the ring signature scheme, and use the combinatorial notion of fair partition. This scheme is provably secure in the random oracle model.

keywords: threshold ring signature, random oracle model.

1 Introduction

Anonymity is required to ensure that the information about the user is not revealed in some multi-user cryptographic applications. The notion of group signature was introduced by Chaum and van Heijst [8], allows a registered member of a predefined group to produce anonymous signatures on behalf of the group. However, this anonymity can be revoked by an authority if necessary. The distinct but related concept of ring signature has been formalized by Rivest, Shamir, and Tauman [14]. This concept is of particular interest when the members do not agree to cooperate since the scheme requires neither a group manager, nor a setup procedure, nor the action of a non-signing member.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 406–416, 2005.

A ring signature specifies a set of possible signers and a proof that is intended to convince any verifier that the author of the signature belongs to this set, while hiding her identity. The scheme is said to be signer ambiguous in the sense that the verifier cannot tell which user in this set actually produces the signature.

There are many schemes proposed for group signatures with additional properties [11,12,13] as well as increasing efficiency [2,6,7]. Cramer, Damgård, and Schoenmakers [9] treated witness hiding zero-knowledge proofs, and an application to group signatures without a group manager was discussed at the end of this article. Their scheme can be seen as ring signature. Recently, Dodis, Kiayias, Nicolosi, and Shoup [10] proposed a variant of ring signature schemes based on the notion of an accumulator with a one-way domain [3,5]. The signature size of their scheme is independent of the number of group members and provably secure in the random oracle model.

Assume that in order to create a certain signature at least t out of the n parties need to combine their knowledge. Combining the shares must not reveal the actual secret key. The correctness of the signature would be verifiable using the public keys. Any t out of the n parties can perform some cryptographic operation jointly, whereas it is infeasible for at most $t-1$ parties to do so. In 2002, Bresson, Stern, and Szydlo [4] proposed a threshold ring signature scheme. Their scheme uses the notion *fair partition*, and is provably secure in the random oracle model and efficient when the number t of signers is small compared with n, i.e., $t = \mathcal{O}(\log n)$ (we call this scheme **BSS scheme**). However, it is inefficient when t is $\omega(\log n)$.

In this paper, we propose a new threshold ring signature scheme which is efficient when the number of signers is large compared with the number n of group members, i.e., when the number t of non-signers is small compared with n. This scheme is very efficient when $t = \mathcal{O}(\log n)$.

Consider the case that a majority of members in some section of a company wishes to claim something for a director of the company. BSS scheme does not work well for this simple case. Our scheme has a kind of dual structure of BSS scheme. They used a structure of so-called *super-ring*, which has standard 1-out-of-n ring signatures as nodes. In our scheme, we still employ a simple structure of ring (not super-ring), and modify the trap-door one-way permutations for it. Our scheme uses the fair partitions for proving correctness and unforgeability, while BSS scheme uses the fair partitions for correctness and anonymity.

The organization of the paper is as follows: in section 2, we present some definitions and background useful for this paper. In section 3, we present our threshold ring signature scheme using fair partition which is more efficient when the threshold is large compared with n and prove its security.

2 Preliminaries

2.1 Formulations of Standard and Threshold Ring Signature

In this paper, we follow the formalization proposed by Rivest, Shamir, and Tauman [14]. They proposed the notion of ring signature, which allows a member

of an ad-hoc collection of users S to prove that a message is authenticated by a member of S without revealing which member actually produced the signature.

We assume that each user has received a public key PK_k, for which the corresponding secret key is denoted by SK_k. A ring signature scheme consists of the following algorithms.

- **Ring-sign**: A probabilistic algorithm outputs a ring signature σ for the message m, with input a message m, the public keys $PK_1, \ldots, PK_r$ of the r ring members, together with the secret key SK_s of a signer.
- **Ring-verify**: A deterministic algorithm outputs either "`ACCEPT`" or "`REJECT`" with input (m, σ) (where σ includes the public key of all the possible signers).

A ring signature scheme must satisfy the correctness (i.e. a correct ring signature should be accepted as valid with overwhelming probability) and unforgeability (i.e. it must be infeasible for any non-ring member to generate a valid ring signature, except with negligible probability). We also require anonymity that nobody should be able to guess the actual signer's identity with probability greater than $1/r + \epsilon$, where r is the number of the ring members, and ϵ is negligible.

In [4], Bresson, Stern, and Szydlo introduced the definition and the security requirements for threshold ring signature. We briefly review them:

A t-out-of-n threshold ring signature scheme consists of the following algorithms:

- **T-ring-sign:** A probabilistic algorithm outputs a t-out-of-n threshold ring signature σ on the message m (where σ includes the value of t as well as the n public keys of all ring members), with input a message m, the public keys $PK_1, \ldots, PK_n$ of the n ring members, together with the t secret keys $SK_{i_1}, \ldots, SK_{i_t}$ of t signers.
- **T-ring-verify:** A deterministic algorithm outputs either "`ACCEPT`" or "`REJECT`" with input (m, σ).

Unforgeability on the threshold ring signature schemes is defined similar to the notion of existential unforgeability against adaptive chosen message attacks on the standard signature schemes. However, since threshold ring signature allows the signers to choose an ad-hoc collection of users, there could be public keys of users who were corrupted by the adversary (e.g., there could be public-keys generated by the adversary). We formalize this unforgeability property by considering the following adversarial model. The adversary $\mathcal{A}$ is given the public keys $PK_1, \ldots, PK_n$ of the n ring members, and can access to the hash function $\mathcal{H}$. Also, $\mathcal{A}$ is given access to the signing oracle. We define that t-forger against a threshold ring signature is a probabilistic polynomial-time Turing machine $\mathcal{A}$, that can sign a message on behalf of t users, with up to $t-1$ corrupted users, under the adaptive chosen message attack.

Definition 1 (Unforgeability). *We say a t-out-of-n threshold ring signature scheme is t-CMA-secure if no t-forger $\mathcal{A}$ can succeed to forge a signature with non-negligible probability.*

We require the signature to have anonymity (i.e. nobody should be able to guess the actual signer's identity) and unforgeability (i.e. the scheme is t-CMA-secure).

2.2 BSS Scheme

In this paper, we propose an $(n-t)$-out-of-n ring signature scheme, where t is small compared with n. In our scheme, we use a kind of dual structure of BSS scheme, and employ the combinatorial notion called *fair partition* that is used in BSS scheme. We briefly review its definition and (n,t)-complete partitioning systems introduced in [4] (see also [1]).

Let $\pi = (\pi^1, \cdots, \pi^t)$ be a partition of $[1,n]$ in t subsets and $I = \{i_1, \cdots, i_t\}$ a set of t indices in $[1,n]$. If all integers in I belong to t different subsets, we say that π is a *fair partition for* I.

Definition 2. *Let $\pi = (\pi^1, \cdots, \pi^t)$ be a partition of $[1,n]$ in t subsets and $I = \{i_1, \cdots, i_t\}$ a set of t indices in $[1,n]$. We say that π is a* ***fair partition*** *for I if for all $j \in [1,t]$,*

$$\#(I \cap \pi^j) = 1.$$

$\#(A)$ denotes the number of elements in A.

To ensure anonymity, we need to provide a set Π of partitions such that there exists a fair partition for any set I of t indices in $[1,n]$.

Definition 3. *Let $t < n$. We say that a set Π of partitions of $[1,n]$ is an (n,t)-* ***complete partitioning system*** *if for any set I of cardinality t, there exists a fair partition in Π for I.*

A perfect hash function for a set I is a mapping $h : [1,n] \to [1,t]$ which is 1-1 on I. We say H is an (n,t)-family of perfect hash functions if for any I of size t, there exists $h \in H$ which is perfect on I. It is clear that defining a partition in t sub-groups for each member of an (n,t)-family makes an (n,t)-complete partitioning system. In [1], Alon, Yuster, and Zwick has been proved that there exists an (n,t)-family of perfect hash functions which have size of $2^{\mathcal{O}(t)} \log n$. Moreover each of these functions is efficiently computable.

We briefly describe the idea of BSS scheme. Consider a ring of n members, and among them t users who want to sign for a message. Let $I = \{i_1, \ldots, i_t\}$ be a set of t indices in $[1,n]$ such that $P_{i_1}, \ldots, P_{i_t}$ are signers. The idea is to split the group into t disjoint sub-groups regard to a fair partition for I, and to show that each of these sub-groups contains one signer by producing sub-rings. However doing so may compromise perfect anonymity since such split restricts the anonymity of each user to a sub-ring. To ensure anonymity, their scheme needs to split the group regard to an (n,t)-complete partitioning system for which any t users are in different sub-rings. Then all of these splits are used as nodes in a super-ring. The super-ring proves that at least one split has been solved.

The size of the signature in this scheme is $\mathcal{O}(\ell 2^t n \log n)$, the time complexity for a signing is t computations for g_i^{-1} and $\mathcal{O}(2^t n \log n)$ computations for g_i.

3 Our Scheme

In this section, we propose an efficient $(n-t)$-out-of-n threshold ring signature scheme where the number t of non-signers is small compared with the number n of ring members.

3.1 Idea and Description

We briefly describe the idea of our scheme. Consider a ring of n members, and among them $n-t$ users who want to sign for a message. Note that there are t non-signers in the ring members. Let $I = \{i_1, \ldots, i_t\}$ be a set of t indices in $[1, n]$ such that $P_{i_1}, \ldots, P_{i_t}$ are non-signers. The idea is to split the group of ring members by using a fair partition. Suppose we split the group into $t+1$ disjoint sub-groups by using some partition. Then, there is at least one sub-group which does not contain any non-signer in the partition (we call this sub-group *legal* sub-group). However doing so may compromise unforgeability since with such split to non-signers can be in the same sub-group. To ensure unforgeability, our scheme needs to split the group regard to an $(n, t+1)$-complete partitioning system for which any $t+1$ users are in different sub-groups in the partitioning system.

We now describe formally our $(n-t)$-out-of-n threshold ring signature scheme. We denote by ℓ a security parameter and by $\Pi_n^{t+1} = \{\pi_1, \ldots, \pi_p\}$ where $p = 2^{t+1} \log n$ an $(n, t+1)$-complete partitioning system. Let $\pi_i = (\pi_i^1, \ldots, \pi_i^{t+1})$, where each π_i^j is a set of indices. Let $\{P_1, \ldots, P_n\}$ be a set of n ring members. For each $i = 1, 2, \ldots, n$, let g_i be a trapdoor one-way permutation with the anonymity property (see Appendix A).

For each i, j, we denote by q_i^j the number of elements of π_i^j, and by Q the maximum value of q_i^j. Let $\pi_i^j = \{p_i^{j,1}, \ldots, p_i^{j,q_i^j}\}$.

We assume that for all integer n and $t \leq n$, an $(n, t+1)$-complete partitioning system is publicly available, and that each user P_i uses a standard signature scheme built on a trapdoor one-way permutation g_i over $\{0,1\}^\ell$. We say π_i^j is *legal* if for all $k \in \pi_i^j$, P_k is a signer.

For each sub-group π_i^j, we define a trapdoor one-way permutation G_i^j:

If $q_i^j = Q$, then let $S_i^j = \pi_i^j$, else let $S_i^j = \{\pi_i^j \cup \{p_i^{j,q_i^j+1}, p_i^{j,q_i^j+2}, \ldots, p_i^{j,Q}\}\}$, where $p_i^{j,q_i^j+1} = p_i^{j,q_i^j+2} = \ldots = p_i^{j,Q} = p_i^{j,q_i^j}$.

$$G_i^j(x_1, \ldots, x_Q) = g_{p_i^{j,1}}(x_1) || \cdots || g_{p_i^{j,Q}}(x_Q).$$

Thus, each G_i^j has a Q-tuple of ℓ-bit strings as input and outputs a $(Q \times \ell)$-bit string, since each $g_{p_i^{j,k}}$ $(k = 1, 2, \ldots, Q)$ is a trapdoor one-way permutation of $P_{p_i^{j,k}}$ over $\{0,1\}^\ell$. The trapdoor of G_i^j is a set of all $g_{p_i^{j,k}}$'s trapdoors. It is clear that G_i^j is a trapdoor one-way permutation since one have to invert all $g_i^{j,k}$'s in order to invert G_i^j. For example, we assume that each $g_{p_i^{j,k}}$ is an extended

RSA permutation [14] and let $(n_{p_i^{j,k}}, e_{p_i^{j,k}})$ be the public key of $P_{p_i^{j,k}}$ and $d_{p_i^{j,k}}$ to be the secret key of $P_{p_i^{j,k}}$. Then, the trapdoor of G_i^j is $(d_{p_i^{j,1}}, \ldots, d_{p_i^{j,Q}})$ (see Appendix A).

Signing algorithm.

The signers execute the following steps for each π_i $(i = 1, 2, \ldots, p)$. We assume that there are at most t non-signers. Then, there are at least one legal sub-group in each partition, since the ring members are split into $t+1$ disjoint sub-groups. We assume that S_i^j is legal. Let m be a message to sign and $\mathcal{H}$ a hash function that maps strings of arbitrary length to $(Q \times \ell)$-bit strings. Note that the length of the message is arbitrary.

- **Choose random seeds:** The signers choose random seeds $s^1, \ldots, s^Q \in \{0,1\}^\ell$ and compute
$$v_{j+1} = \mathcal{H}(m, s^1, \ldots, s^Q).$$
- **Pick random x's:** For each $k = j+1, \ldots, t+1, 1, 2, \ldots, j-1$, the signers choose $x_k^1, \ldots, x_k^Q \in \{0,1\}^\ell$ at random, and compute
$$v_{k+1} = \mathcal{H}(m, v_k \oplus G_i^k(x_k^1, \ldots, x_k^Q)).$$
- **Invert the legal S_i^j's trapdoor permutation:** The signers use their knowledge of trapdoors of each $g_{p_i^{j,k}}$ in order to invert G_i^j to obtain $x_j^1, \ldots, x_j^Q$ such that $v_{j+1} = \mathcal{H}(m, v_j \oplus G_i^j(x_j^1, \ldots, x_j^Q))$.
- **Output the signature for π_i:** The signers choose at random an index $i_0 \in \{1, 2, \ldots, t+1\}$, then the signature on the message m for π_i is defined as the $(2(t+1)Q+2)$-tuple:
$$\sigma_i = (PK_{p_i^{1,1}}, \ldots, PK_{p_i^{1,Q}}, PK_{p_i^{2,1}}, \ldots, PK_{p_i^{t+1,Q}}; i_0; v_{i_0}; x_1^1, \ldots, x_1^Q, x_2^1, \ldots, x_{t+1}^Q).$$

  Thus, the signature on the message m is defined to be the p-tuple:
$$\sigma = (\sigma_1, \ldots, \sigma_p).$$

Verifying algorithm.

The verifier can verify each σ_i $(i = 1, 2, \ldots, p)$ on the message m as follows.

- **Apply the trapdoor permutations:** For $k = i_0 + 1, i_0 + 2, \ldots, t + 1, 1, 2, \ldots, i_0 - 1$, the verifier computes
$$v_k = \mathcal{H}(m, v_{k-1} \oplus G_i^j(x_k^1, \ldots, x_k^Q)),$$
  and checks the equation:
$$v_{i_0} = \mathcal{H}(m, v_{i_0-1} \oplus G_i^j(x_{i_0-1}^1, \ldots, x_{i_0-1}^Q)).$$
  If this equation is satisfied, then σ_i is "`TRUE`", otherwise "`FALSE`".

If for all $k = 1, 2, \ldots, p$, σ_i is "`TRUE`", then the verifier outputs "`ACCEPT`", otherwise "`REJECT`".

3.2 Security Analysis

We prove that the above scheme has the required property of threshold ring signature.

Completeness. If there are at most t non-signers among n ring members then there is at least one legal sub-group for each π_i. Therefore, it is straightforward that $n-t$ members are able to produce $p = 2^{t+1} \log n$ signatures for all partitions which are accepted as `TRUE` on behalf of $n-t$ signers.

Unforgeability. In order to prove unforgeability, we use the $(n, t+1)$-complete partitioning system. We have to prove that a correct signature is necessarily produced by at least the claimed number of signers. We consider an adversary $\mathcal{A}$ that can mount an adaptive chosen message attack, and is able to succeed with probability $\mathtt{Succ}^{\mathcal{H},sign}_{forge,\mathcal{A}}(\ell)$. We construct from it an adversary $\mathcal{B}$ against the one-wayness of some g_i.

In particular, we show that in the random oracle model, if there exists a forging algorithm $\mathcal{A}$ which can generate a new $(n-t)$-out-of-n ring signature for a message with $(n-t-1)$ corrupted signers with non-negligible probability then there exists an algorithm $\mathcal{B}$ which outputs x_0 such that $y_0 = g_0(x_0)$ on input the public key of g_0 and a random value y_0 with non-negligible probability.

Algorithm $\mathcal{A}$ accepts the public keys $PK_1, \ldots, PK_n$, is given oracle access to the random oracle $\mathcal{H}$ and to the $(n-t)$-out-of-n ring signing oracle, and can corrupt $(n-t-1)$ signers. We assume that algorithm $\mathcal{A}$ can produce a valid $(n-t)$-out-of-n threshold ring signature on a new message which was not queried to the signing oracle, with non-negligible probability.

Algorithm $\mathcal{B}$ with input y_0 and the public key PK_0 of g_0 chooses at random an index $i_0 \in [1, n]$ and a subset $I_0 \subset [1, n]$ of cardinality $n-t-1$ such that $i_0 \notin I_0$. Algorithm $\mathcal{B}$ hopes that algorithm $\mathcal{A}$ corrupts all of the members in I_0. Algorithm $\mathcal{B}$ sets P_{i_0}'s public key to PK_0, and generates $n-t-1$ pairs of matching secret/public keys for all members in I_0, and sets other members' public keys at random (i.e., the corresponding secret keys will not be used). Algorithm $\mathcal{B}$ chooses at random an integer t_0 in $[1, Q]$. It also chooses at random q_0, q_0' in $[1, q(\ell)]$ such that $q_0 < q_0'$, where $q(\ell)$ is the total number of queries for $\mathcal{H}$.

Algorithm $\mathcal{B}$ simulates the random oracle $\mathcal{H}$ in the usual way, answering a random value for each new query, and maintaining a list of already queried messages. However, we add the following rule. Let Γ_0 be the XOR between the argument of the q_0-th query for the oracle $\mathcal{H}$ and the answer to the q_0'-th query, and γ_0 the substring of Γ_0, corresponding to $[\Gamma_0]_{\ell(t_0-1)+1}$ through $[\Gamma_0]_{\ell t_0}$, where $[x]_i$ denotes the i-th bit of the string x. On the q_0'-th query for $\mathcal{H}$, algorithm $\mathcal{B}$ answers $y_0 \oplus r_0$, where r_0 is the value the q_0-th query was made on.

Algorithm $\mathcal{A}$ is allowed to query for the secret key of up to $n-t-1$ members, in an adaptive way. Such queries are answered in a straightforward way, except if asked to P_j, $j \notin I_0$. In that case, algorithm $\mathcal{B}$ halts and outputs "`FAIL`". The probability that algorithm $\mathcal{A}$ guesses correctly the subset I_0 of corrupted members is at least $1/\binom{n}{n-t-1}$.

Algorithm $\mathcal{B}$ can also simulate the signing oracle, simply by choosing randomly the components of the signature, and by adapting its answers to the queries for $\mathcal{H}$.

Algorithm $\mathcal{A}$ outputs a forgery:

$$(m^*, \sigma_1, \ldots, \sigma_p),$$

where $p = 2^{t+1} \log n$ and each $\sigma_i = (PK_{p_i^{1,1}}, \ldots, PK_{p_i^{1,Q}}, PK_{p_i^{2,1}}, \ldots, PK_{p_i^{t+1,Q}}; i_0; v_{i_0}; x_1^1, \ldots, x_1^Q, x_2^1, \ldots, x_{t+1}^Q)$. If there exist indices i, j for which $g_{i_0}(x_i^j) = y_0$, then algorithm $\mathcal{B}$ outputs x_i^j, otherwise, it outputs "FAIL".

However, algorithm $\mathcal{B}$ does not know which queries will be these queries in the forged $(n-t)$-out-of-n threshold ring signature, and thus algorithm $\mathcal{B}$ has to guess their identity for slipping y_0 as the gap. Hence, the probability that these queries were the q_0-th and q_0'-th queries is at least $1/\{q(\ell)\}^2$. Moreover, since algorithm $\mathcal{B}$ does not know which g_0 is used in G_i^j, the probability of guessing t_0 correctly is at least $\frac{1}{Q}$, where Q is the maximum number of the elements in the partition.

Let $\bar{I}_0$ be the complement set of I_0. We note that the size of $\bar{I}_0$ is $t+1$ and $\bar{I}_0$ contains a signer. Since Π_n^{t+1} is an $(n, t+1)$-complete partitioning system, there exists the fair partition π_{i^*} ($i^* \in [1, p]$) for $\bar{I}_0$. Therefore, there is a sub-group in π_{i^*}, such that the sub-group consists of only signers including one in $\bar{I}_0$.

If algorithm $\mathcal{B}$ guesses correctly I_0, q_0, q_0', and i_0, algorithm $\mathcal{B}$ can "slip" y_0 and succeed to invert g_{i_0}. Therefore, algorithm $\mathcal{B}$ outputs a valid x_0 such that $x_0 = g_0^{-1}(y_0)$ with the probability at least $\frac{1}{\binom{n}{n-t-1}} \cdot \frac{1}{q(\ell)^2} \cdot \frac{1}{t+1} \cdot \mathtt{Succ}_{forge,\mathcal{A}}^{\mathcal{H},sign}(\ell)$, where $\mathtt{Succ}_{forge,\mathcal{A}}^{\mathcal{H},sign}(\ell)$ is the probability that algorithm $\mathcal{A}$ produces a valid $(n-t)$-out-of-n threshold ring signature on a new message.

By the assumption, $\mathtt{Succ}_{forge,\mathcal{A}}^{\mathcal{H},sign}(\ell)$ is non-negligible. Therefore, since algorithm $\mathcal{B}$ can invert g_0 with non-negligible probability, this is contradiction that g_i is a one-way permutation.

Anonymity. For each signature σ_i for partition π_i, no one can guess which the sub-group consists of only signers with the probability $\frac{1}{t+1} + \epsilon$, where ϵ is negligible. This is due to the fact that for any message m and any v_i, the simultaneous system of equations:

$$\begin{aligned}
v_2 &= \mathcal{H}(m, v_1 \oplus G_i^1(x_1^1, \ldots, x_1^Q)),\\
v_3 &= \mathcal{H}(m, v_2 \oplus G_i^2(x_2^1, \ldots, x_2^Q)),\\
&\vdots\\
v_{t+1} &= \mathcal{H}(m, v_t \oplus G_i^t(x_t^1, \ldots, x_t^Q)),\\
v_1 &= \mathcal{H}(m, v_{t+1} \oplus G_i^{t+1}(x_{t+1}^1, \ldots, x_{t+1}^Q))
\end{aligned}$$

has exactly $2^{\ell \cdot Q \cdot t}$ solutions which can be obtained with equal probability. This implies the anonymity of user's identity.

3.3 Efficiency

We discuss the efficiency of our scheme. Let n be the number of ring members, t the number of non-signers, and ℓ the security parameter. Recall that a threshold ring signature can be stated as $\sigma = (\sigma_1, \ldots, \sigma_p)$, where $p = 2^t \log n$ and each $\sigma_i = (PK_{p_i^{1,1}}, \ldots, PK_{p_i^{1,Q}}, PK_{p_i^{2,1}}, \ldots, PK_{p_i^{t+1,Q}}; i_0; v_{i_0}; x_1^1, \ldots, x_1^Q, x_2^1, \ldots, x_{t+1}^Q)$. Each size of PK_i, x_i^j, and v_{i_0} are ℓ. Therefore, the size of an $(n-t)$-out-of-n threshold ring signature is $2^t \log n \times \{2((t+1) \times Q \times \ell) + \ell \times Q\} = \mathcal{O}(t \cdot Q \cdot \ell 2^t n \log n)$. The time complexity for signing is $Q \times 2^t \log n$ computations of g_i^{-1} and $Q \times (t+1) \times 2^t \log n = \mathcal{O}(t \cdot Q \cdot 2^t n \log n)$ computations of g_i. That is, in the worst case (i.e., $Q = n$), the size of an $(n-t)$-out-of-n threshold ring signature is $\mathcal{O}(t \cdot \ell 2^t n^2 \log n)$ and the time complexity for signing is $n \times 2^t \log n$ computations of g_i^{-1} and $\mathcal{O}(t \cdot 2^t n^2 \log n)$ computations of g_i. Note that the size of an $(n-t)$-out-of-n signature of BSS scheme is $\mathcal{O}(\ell 2^{(n-t)} n \log n)$ and the time complexity for signing of BSS scheme is $n - t$ computations of g_i^{-1} and $\mathcal{O}(2^{(n-t)} n \log n)$ computations of g_i. Therefore, our scheme is more efficient than BSS scheme when the number of signers is large compared with the number of ring members.

Also note that our scheme is more efficient than generic solution such that making ring signatures for all subgroups cardinality $n - t + 1$ since this would lead to $\binom{n}{t-1} = \mathcal{O}(n^{t-1})$ size.

4 Acknowledgments

We thank the anonymous referees for valuable comments.

References

1. Alon, N., Yuster, R., and Zwick, U. Color-coding. *Electronic Colloquium on Computational Complexity (ECCC) 1*, 009 (1994). Full paper appears in J.ACM 42:4, July 1995, 844-856.
2. Ateniese, G., Camenisch, J., Joye, M., and Tsudik, G. A practical and provably secure coalition-resistant group signature scheme. In *Advances in Cryptology – CRYPTO 2000* (Santa Barbara, California, USA, August 2000), M. Bellare, Ed., vol. 1880 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 255–270.
3. Barić, N., and Pfizmann, B. Collision-free accumulators and fail-stop signature schemes without trees. In *Advances in Cryptology – EUROCRYPT '97* (Konstanz, Germany, May 1997), W. Fumy, Ed., vol. 1233 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 480–494.
4. Bresson, E., Stern, J., and Szydlo, M. Threshold Ring Signatures and Applications to Ad-hoc Groups. In Yung [15], pp. 465–480.
5. Camenisch, J., and Lysyanskaya, A. Efficient revocation of anonymous group membership. In Yung [15], pp. 465–480.
6. Camenisch, J., and Michels, M. Separability and efficiency for generic group signature schemes. In *Advances in Cryptology – CRYPTO '99* (Santa Barbara, California, USA, August 1999), M. Wiener, Ed., vol. 1666 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 106–121.

7. Camenisch, J., and Stadler, M. Efficient group signatures schemes for large groups. In *Advances in Cryptology – CRYPTO '97* (Santa Barbara, California, USA, August 1997), B. S. Kaliski, Jr., Ed., vol. 1294 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 410–424.
8. Chaum, D., and van Heijst, E. Group Signatures. In *Advances in Cryptology – EUROCRYPT '91* (Brighton, UK, April 1991), D. Davies, Ed., vol. 547 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 257–265.
9. Cramer, R., Damgård, I., and Schoenmakers, B. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Advances in Cryptology – CRYPTO '94* (Santa Barbara, California, USA, August 1994), Y. G. Desmedt, Ed., vol. 839 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 174–187.
10. Dodis, Y., Kiayias, A., Nicolosi, A., and Shoup, V. Anonymous identification in ad hoc groups. In *Advances in Cryptology – EUROCRYPT 2004* (Interlaken, Switzerland, May 2004), C. Cachin and J. Camenisch, Eds., vol. 3027 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 609–626.
11. Kim, S., Park, S., and Won, D. Convertible group signatures. In *Advances in Cryptology – ASIACRYPT '96* (Kyongju, Korea, November 1996), K. Kim and T. Matsumoto, Eds., vol. 1163 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 311–321.
12. Kim, S., Park, S., and Won, D. Group signatures for hierarchical multigroups. In *ISW '97* (1997), vol. 1396 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 273–281.
13. Petersen, H. How to convert any digital signature scheme into a group signature. In *Security protocols '97* (1997), vol. 1361 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 67–78.
14. Rivest, R. L., Shamir, A., and Tauman, Y. How to Leak a Secret. In *Advances in Cryptology – ASIACRYPT 2001* (Gold Coast, Australia, December 2001), C. Boyd, Ed., vol. 2248 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 552–565.
15. Yung, M., Ed. *Advances in Cryptology – CRYPTO 2002* (Santa Barbara, California, USA, August 2002), vol. 2442 of *Lecture Notes in Computer Science*, Springer-Verlag.

A The Primitive Ring Signature Scheme

Bresson, Stern, and Szydlo proposed a modification of the original Rivest–Shamir–Tauman ring signature scheme. In this section, we briefly review this modification proposed by Bresson, Stern, and Szydlo [4], based on the random oracle model, while the original Rivest–Shamir–Tauman scheme uses the ideal cipher model.

We denote by ℓ, ℓ_b two security parameters. We consider a hash function $\mathcal{H}$ that maps arbitrary strings to ℓ_b-bit strings. We assume that each user P_i uses a regular signature scheme built on a trapdoor one-way permutation f_i on $\mathbb{Z}_{n_i}^*$: $f_i(x) = x^{e_i} \bmod n_i$ where $|n_i| = \ell_b < \ell$. Then,

$$g_i(x) = \begin{cases} q_i n_i + f_i(r_i) & \text{if } (q_i+1)n_i \leq 2^\ell \\ x & \text{otherwise} \end{cases} \tag{1}$$

where $x = q_i n_i + r_i$ and $0 \leq r_i < n_i$. Intuitively, the input x is sliced into ℓ_b-bit long parts which then go through the trap-door permutation f_i. If we choose a sufficiently large ℓ (e.g., 160 bits larger than any of the n_i), the probability that a random input is unchanged becomes negligible.

Generating a ring signature

Given the message m to be signed, her secret key SK_s, and the sequence of public keys $PK_1, PK_2, \ldots, PK_r$ of all the ring members, the signer computes a ring signature as follows.

1. **Choose a random seed:** The signer picks a random seed σ in $\{0,1\}^{\ell_b}$, and computes
$$v_{s+1} = \mathcal{H}(m, \sigma).$$
2. **Pick random x_i's:** The signer picks random x_i for all the other ring members $1 \leq i \leq r$, $i \neq s$ uniformly and independently from $\{0,1\}^{\ell_b}$, and computes for $i = s+1, s+2, \ldots, n, 1, 2, \ldots, s-1$,
$$v_{i+1} = \mathcal{H}(m, v_i \oplus g_i(x_i))$$
where
$$g_i(x) = \begin{cases} q_i n_i + f_i(r_i) & \text{if } (q_i+1)n_i \leq 2^\ell \\ x & \text{otherwise} \end{cases}$$
with $x = q_i n_i + r_i$ and $0 \leq r_i < n_i$.
3. **Solve for x_s:** The signer solves the following equation for x_s by using her knowledge of trap-door permutation:
$$\sigma = v_s \oplus g_s(x_s).$$
4. **Output the signature:** The signer chooses at random an index $i_0 \in \{1, 2, \ldots, r\}$, then the signature on the message m is defined as the $(2r+2)$-tuple:
$$(PK_1, PK_2, \ldots, PK_r; i_0; v_{i_0}; x_1, x_2, \ldots, x_r).$$

Verifying a ring signature

A verifier can verify an alleged signature
$$(PK_1, PK_2, \ldots, PK_r; i_0; v_{i_0}; x_1, x_2, \ldots, x_r)$$
on the message m as follows.

1. **Apply the trapdoor permutations:** For $i = i_0 + 1, i_0 + 2, \ldots, n, 1, 2, \ldots, i_0 - 1$, the verifier computes
$$v_i = \mathcal{H}(m, v_{i-1} \oplus g_{i-1}(x_{i-1})).$$
2. **Verify the equation:** The verifier checks that the v_i's satisfy the equation:
$$v_{i_0} = \mathcal{H}(m, v_{i_0-1} \oplus g_{i_0-1}(x_{i_0-1})).$$
If this equation is satisfied, the verifier outputs "`ACCEPT`", otherwise "`REJECT`".

Deposit-Case Attack Against Secure Roaming

Guomin Yang, Duncan S. Wong*, and Xiaotie Deng

Department of Computer Science
City University of Hong Kong
Hong Kong, China
{csyanggm,duncan,deng}@cs.cityu.edu.hk

Abstract. A secure roaming protocol involves three parties: a roaming user, a visiting foreign server and the user's home server. The protocol allows the user and the foreign server to establish a session key and carry out mutual authentication with the help of the home server. In the mutual authentication, user authentication is generally done in two steps. First, the user claims that a particular server is his home server. Second, that particular server is called in by the foreign server for providing a 'credential' which testifies the user's claim. We present a new attacking technique which allows a malicious server to modify the user's claim in the first step without being detected and provide a fake credential to the foreign server in the second step in such a way that the foreign server believes that the malicious server is the user's home server. We give some examples to explain why it is undesirable in practice if a roaming protocol is vulnerable to this attack. We also show that there are three roaming protocols proposed previously which are vulnerable to this attack.

Keywords: Protocol Security Analysis, Authenticated Key Exchange, Roaming

1 Introduction

With the rapid development of mobile technologies, user mobility is becoming an important network feature nowadays. People can travel around with their mobile devices without being limited by the geographical coverage of their home networks. They can access different foreign networks, identify themselves as subscribers of their home networks and get access to the foreign networks after passing some authentication procedures. This scenario is called *roaming*.

A typical roaming scenario involves three parties: a *roaming user*, A, a visiting *foreign server*, V, and the user's *home server*, H. The roaming user A subscribed to the home server H is now in a network operated by the foreign server V. A communicates directly with V but does not have direct link with H. On the other hand, V has direct link with H. Before allowing A to connect to V, the

* The work was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. 9040904 (RGC Ref. No. CityU 1161/04E)).

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 417–428, 2005.

foreign server V first finds out the identity of A's home server and obtains a valid 'credential' from the home server which testifies that A is a legitimate subscriber of the home server. In other words, V does not authenticate A directly. Instead, H authenticates A via V and then provides a credential to V for testifying the authentic subscription of A.

Roaming services have been widely deployed in cellular networks such as [8, 11] and 3GPP[1]. Besides mobile communications, there are many other systems and applications that can be considered as roaming in protocol perspective. Some of them are actually roaming on wired networks. In [1], Ateniese, et al. gave two examples. One is the inter-bank ATM networks and the other is the credit card payment systems. In an inter-bank ATM network, a customer (a roaming user) comes to an ATM machine[2], which is not operated by the customer's bank, and accesses his bank account. Financial transactions such as withdrawing and depositing money are provided by the ATM machine after the machine has obtained enough assurance on the customer's good standing with respect to his ATM card. Similar roaming environment exists in a credit card payment system by considering the merchant's bank as the foreign server and the credit card issuing bank as the home server of the credit card holder. There are some emerging technologies which can also be modeled as roaming. For example, hopping across meshed WLANs (Wireless Local Area Networks) administered by different individuals, joining and leaving various wireless ad hoc networks operated by different foreign operators, etc.

On the security of roaming, almost all secure roaming protocols support *Subscription Validation* and *Key Establishment*.

Subscription Validation is satisfied if the following conditions are satisfied.

1. The foreign server is sure about the home server of the user.
2. The foreign server gets some 'credential' from the home server of the user which testifies that the user is a legitimate subscriber of the home server.

Key Establishment allows the foreign server and the roaming user to share a session key which is used to secure the communication channel between them. There are some other security requirements for some specific roaming protocols. For example, the latest cellular system 3GPP requires *Server Authentication* which allows the roaming user to authenticate the visiting foreign server. Some other roaming protocols [1, 10, 12, 5] also consider *User Anonymity and Untraceability* as required security objectives. These additional security requirements enable users to roam anonymously without being located or tracked.

Among these security requirements, Subscription Validation is intuitively related to the financial interests of the foreign server and the home server of the user. By getting a credential from the home server of the user, a foreign server is able to request the user's home server for service charge as the credential becomes a proof for payment request. In order to protect the financial interests of both the foreign server and the home server, the credential is required to

[1] http://www.3gpp.org

[2] For example, an ATM terminal with Visa/PLUS or Mastercard/Cirrus sign on.

be secure against forgery and it should also be one-time so that the credential cannot be replayed.

In this paper, we show that Subscription Validation is also related to the financial interest of the user. We present a new attacking technique which incurs the following two results simultaneously.

1. The attack allows a malicious server to persuade the visiting foreign server of a roaming user that the malicious server is the user's home server without being noticed by the user nor the real home server.
2. The roaming user, however, believes that the foreign server has obtained the correct value about the identity of his home server.

We call this attack the **Deposit-case Attack** as such attack is profitable to the malicious server in the case when the user is accessing the foreign server to 'deposit' some information of value (such as electronic cash) to his home server.

The first impression one may have on the deposit-case attack is that it is similar to an Unknown Key Share Attack [3]. In some cases, they cause similar damage. However in some other cases, they are different.

An unknown key share attack applies to a key agreement protocol [2]. It makes one party A believe that a session key is shared with a party B when it is in fact shared with another party C. If party B is the adversary, then the unknown key share attack causes similar damage on a key agreement protocol to that of the Deposit-case Attack on a roaming protocol.

However, a roaming protocol is not simply a kind of key agreement protocols. A roaming protocol can also be an authentication protocol when the Key Establishment between the roaming user and the foreign server is not required. In this case, the Deposit-case Attack against an authentication-only roaming protocol will make the user A believe that the foreign server V has obtained the identity of A's home server (i.e. H) when it has in fact obtained the identity of another server which is malicious.

The Deposit-case Attack is not well captured in the security requirements of current roaming protocols. Apparently, the attack may not even be considered in many of such protocols as we have found three roaming protocols [10, 5, 7] that are vulnerable to this attack. In the following, we give details of the deposit-case attack and explain how this attack could bring very undesirable consequences in practice (Sec. 2). Then we show that there are three roaming protocols that can be compromised by the deposit-case attack in Sec. 3, 4 and 5, respectively. We conclude the paper in Sec. 6 by discussing a corrective approach against this attack.

2 Deposit-Case Attack

In most of the current roaming protocols, Subscription Validation is done in two steps.

1. The roaming user A claims that a particular server H is his home server.
2. That particular server, H, is then called in as a guarantor by the visiting foreign server V for giving a promise (as a one-time unforegable credential) that A is one of H's legitimate subscribers.

In the second step above, H generates a credential only after authenticating A. This effectively prevents the following attack.

Consider a malicious user B, who is not subscribed to any server, claims that a server, H, is his home server and manage to create a fake credential which results to have the foreign server V believe that H is B's home server. This attack directly conflicts with the interest of H if the Subscription Validation protocol is vulnerable to this attack. Most of the current roaming protocols have the two-step Validation Subscription mechanism described above implemented to thwart this attack.

We now consider a new attacking scenario which is called the Deposit-case Attack against roaming protocols. In this scenario, the user is honest while there is a malicious server[3], M. Suppose the user's home server is H. The malicious server M will make the foreign server V believe that the home server of the user is M without being detected by the user nor the real home server H of the user.

Notice that the two-step Subscription Validation mechanism described above may not be able to prevent the Deposit-case Attack because when the foreign server receives a valid credential from the malicious server, there is no guarantee that the user's claim in the first step is not modified. Suppose the malicious server M modifies the user's claim in the first step and produces a one-time unforgeable credential to the foreign server in the second step. This can be done by M as M is also a server in the system. Consequently, the foreign server believes that M is the user's home server. In this attack, the user believes that he has correctly informed the foreign server that his home server is H while the foreign server believes that the home server of the user is the malicious server M.

2.1 Practical Impacts of the Deposit-Case Attack

It is undesirable if a roaming protocol is vulnerable to the Deposit-case Attack. This attack is profitable to the malicious server in the case when the user is accessing the foreign server to 'deposit' some information of value (such as electronic cash) to his home server. Since the foreign server believes that the user is a subscriber of the malicious server, credit for this deposit will go to the malicious server.

Consider the roaming environment of an inter-bank ATM system described in Sec. 1. Suppose there is a roaming user using an ATM machine operated by a foreign bank (i.e. a foreign server) to deposit money to his bank account located at the user's bank (i.e. the user's home server). If the ATM system is vulnerable

[3] The malicious server can also be viewed as a malicious 'insider' [6] of the underlying roaming system.

to the deposit-case attack, we can see that it would allow the foreign bank to transfer money to a malicious bank instead of the user's bank account.

To some extent, the Deposit-case Attack causes similar damage on a roaming protocol to that of an unknown key share attack on a key agreement protocol [3]. But they are also different as explained in Sec. 1. In the following, we will see that protocols of [10, 5, 7] cannot defend themselves against the Deposit-case Attack.

3 An Anonymous Roaming Protocol

In [10], Samfat et al. proposed a suite of protocols for secure roaming. Besides Server Authentication, Subscription Validation and Key Establishment, their protocols also support certain degrees of User Anonymity and Untraceability. All of their protocols are derived from one basic protocol. In the following, we first review their basic protocol and show that it is vulnerable to the deposit-case attack. The attacking technique can be applied directly to all of their other protocols.

Let E_K be the encryption function under the symmetric key K. The symmetric key encryption function is assumed to be a block cipher (e.g. AES [9]). We use PKE_A to denote the public key encryption function of party A and Sig_A to denote the signature generation function of A. The $\oplus$ symbol indicates a bitwise exclusive-OR operation and the $||$ symbol represents the binary string concatenation.

3.1 The Basic Protocol of Samfat et al.

There are two functions used as building blocks in the protocol: $Token_K$ and $TICK_K$. $Token_K$ is computed by applying a block cipher E_K over three inputs: m_1, m_2 and m_3.

$$Token_K(m_1, m_2, m_3) = E_K(m_1 \ \oplus \ E_K(m_2 \ \oplus \ E_K(m_3))).$$

$TICK_K$ is called a ticket which is used by an initiator A for sending a session key σ to a responder B. The key is also intended to be shared with a third party C. This is denoted by

$$TICK_K(A, B, C, \sigma) = Token_K(N_1 \oplus C, \ N_2, \ N_1 \oplus A) \oplus \sigma$$

where N_1 and N_2 are nonces that are randomly generated.

Let A be a roaming user, V be a foreign server and H be the home server of the roaming user. The Basic Protocol of Samfat et al. consists of four message flows among these three parties. The fourth message flow is optional. In the following, we first describe the protocol with the first three message flows only. We will consider the fourth message later.

$$
\begin{aligned}
&A \rightarrow V : H,\ alias = PKE_H(N_1 \ ||\ N_1 \oplus A),\\
&\qquad\quad AUTH_1 = \langle\ N_2,\ T,\ Token_{K_{av}}(alias, T, N_2)\ \rangle\\
&V \rightarrow H : alias,\ PKE_H(N_3 \ ||\ N_3 \oplus V),\\
&\qquad\quad AUTH_2 = \langle\ N_4,\ AUTH_1,\ Token_{K_{vh}}(V, AUTH_1, N_4)\ \rangle\\
&H \rightarrow V : PKE_V(N_3),\ TICK_{K_{vh}}(H, V, alias, K_{av})
\end{aligned}
$$

In the protocol, N_1, N_2, N_3, N_4 are nonces. T is a timestamp generated by A. $K_{av} = \mathcal{H}(A||V||K_{ah})$ where K_{ah} is a long-term key shared by A and H, and $\mathcal{H}$ is a one-way hash function. K_{vh} is a long-term key shared by V and H. By $\langle m_1, m_2 \rangle$, we mean some appropriate encoding of two messages m_1 and m_2.

3.2 Deposit-Case Attack

The attack described below follows directly the attacking technique delineated in Sec. 2. In the attack, we consider that there exists a malicious server M. The malicious server M first modifies the user's claim by replacing H with M in the first message flow from A to V. Then when V asks M for a credential, which corresponds to the second message flow, M generates and sends back the third message flow as a credential. As a result, V believes that M is the user's home server without being known by the user A. Below are the details of the attack.

The malicious server M intercepts the message from A to V and launches the following attack.

$$
\begin{aligned}
&A \rightarrow M : H,\ alias = PKE_H(N_1 \ ||\ N_1 \oplus A),\\
&\qquad\quad AUTH_1 = \langle\ N_2,\ T,\ Token_{K_{av}}(alias, T, N_2)\ \rangle\\
&M \rightarrow V : M,\ alias,\ AUTH_1' = \langle\ N_2',\ T',\ Token_{K'}(alias, T', N_2')\ \rangle\\
&V \rightarrow M : alias,\ PKE_M(N_3 \ ||\ N_3 \oplus V),\\
&\qquad\quad AUTH_2' = \langle\ N_4,\ AUTH_1',\ Token_{K_{vm}}(V, AUTH_1', N_4)\ \rangle\\
&M \rightarrow V : PKE_V(N_3),\ TICK_{K_{vm}}(M, V, alias, K')
\end{aligned}
$$

N_2' is a nonce, T' is a timestamp and K' is a random symmetric key generated by M. K_{vm} is a long-term key shared by V and M.

In the attack, A believes that he has informed V that his home server is H while V believes that the home server of A is M.

We now consider the optional fourth message flow. The purpose of this message flow is to allow V to send its public key to A so that the public key can be used for authentication in the future. Let the public key of V be P_V. The fourth message is denoted by

$$TICK_{K_{av}}(V, alias, V, P_V).$$

In the deposit-case attack, the fourth message will become

$$TICK_{K'}(V, alias, V, P_V).$$

Due to the lack of message authentication, we can see that A will still accept, but just get the wrong P_V. Therefore, the deposit-case attack still works.

4 Another Anonymous Roaming Protocol

In [5], Go and Kim proposed a different roaming protocol which targets to achieve the similar set of security goals to that of Samfat et al. reviewed in Sec. 3.

Let (G, g, q) be the domain parameters where $G = \langle g \rangle$ and the order of G is a large prime q. Assume the discrete logarithm problem in G is hard. Let A, V, H denote a roaming user, a foreign server and the home server of the user, respectively. We use the same set of notations as in Sec. 3. Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be some cryptographically strong hash functions. By $x \in_R X$, we mean that an element x is randomly chosen from the set X. Let $(\hat{S}_H, P_H) \in \mathbb{Z}_q \times G$ be H's private key/public key pair such that $P_H = g^{\hat{S}_H}$. Let $(\hat{S}_V, P_V) \in \mathbb{Z}_q \times G$ be V's private key/public key pair such that $P_V = g^{\hat{S}_V}$. Let T_1, T_2 and T_3 be timestamps. Assume the public keys of all parties are publicly known. The Go-Kim protocol is shown as follows.

$$
\begin{array}{ll}
A & : r_a \in_R \mathbb{Z}_q,\ K_{ah} = P_H^{r_a},\ alias = E_{K_{ah}}(\mathcal{H}_1(A) \oplus g^{r_a}) \\
A \rightarrow V & : H,\ alias,\ g^{r_a} \\
V & : r_v \in_R \mathbb{Z}_q \\
V \rightarrow H & : alias,\ g^{r_v},\ g^{r_a},\ Sig_V(g^{r_v}, g^{r_a}, alias, V),\ T_1 \\
H & : r_h \in_R \mathbb{Z}_q,\ K_{hv} = \mathcal{H}_2(g^{r_v r_h}, P_V^{r_h}) \\
H \rightarrow V & : g^{r_h},\ E_{K_{hv}}(Sig_H(g^{r_h}, g^{r_v}, \mathcal{H}_1(A) \oplus g^{r_a}, H), \mathcal{H}_1(A) \oplus g^{r_a}),\ T_2 \\
V & : alias' = \mathcal{H}_1(g^{r_v r_a}, \mathcal{H}_1(A)),\ K_{av} = \mathcal{H}_2(g^{r_v r_a}, g^{\hat{S}_V r_a}) \\
V \rightarrow A & : g^{r_v},\ E_{K_{av}}(\mathcal{H}_1(g^{r_v}, g^{r_a}, alias', V), T_2),\ T_3 \\
A \rightarrow V & : E_{K_{av}}(Sig_A(g^{r_a}, g^{r_v}, T_2, V), T_3)
\end{array}
$$

4.1 Deposit-Case Attack

Direct application of the attacking technique outlined in Sec. 2 would not work over here. This is because the malicious server M has to decrypt *alias* and obtain the real identity of A in order to deliver the correct value to V and let A accept when A receives a commitment of $alias'$ in the second last message flow. However, M does not know K_{ah} which is needed to decrypt *alias*.

Note that *alias* is used to hide the real identity of A so that the Go-Kim protocol can provide user anonymity and untraceability against eavesdroppers. Hence before launching the deposit-case attack, M should find out the real identity of A. Below are the details on how M can find out A's real identity[4] and launch the deposit-case attack. Let $P_M \in G$ be M's public key.

[4] Precisely, M finds out the value of $\mathcal{H}_1(A)$ in the attack. However, the commitment $\mathcal{H}_1(A)$ has already provided enough information for an adversary to trace and reveal the identity of the user.

$$
\begin{array}{ll}
A & : r_a \in_R \mathbb{Z}_q,\ K_{ah} = P_H^{r_a},\ alias = E_{K_{ah}}(\mathcal{H}_1(A) \oplus g^{r_a}) \\
A \rightarrow M & : H,\ alias,\ g^{r_a} \\
M & : r_1 \in_R \mathbb{Z}_q \\
M \rightarrow H & : alias,\ g^{r_1},\ g^{r_a},\ Sig_M(g^{r_1}, g^{r_a}, alias, M),\ T_0 \\
H & : r_h \in_R \mathbb{Z}_q,\ K_{hm} = \mathcal{H}_2(g^{r_1 r_h}, P_M^{r_h}) \\
H \rightarrow M & : g^{r_h},\ E_{K_{hm}}(Sig_H(g^{r_h}, g^{r_1}, \mathcal{H}_1(A) \oplus g^{r_a}, H), \mathcal{H}_1(A) \oplus g^{r_a}),\ T_2 \\
M \rightarrow V & : M,\ alias,\ g^{r_a} \\
V & : r_v \in_R \mathbb{Z}_q \\
V \rightarrow M & : alias,\ g^{r_v},\ g^{r_a},\ Sig_V(g^{r_v}, g^{r_a}, alias, V),\ T_1 \\
M & : r_2 \in_R \mathbb{Z}_q,\ K_{mv} = \mathcal{H}_2(g^{r_v r_2}, P_V^{r_2}) \\
M \rightarrow V & : g^{r_2},\ E_{K_{mv}}(Sig_M(g^{r_2}, g^{r_v}, \mathcal{H}_1(A) \oplus g^{r_a}, M), \mathcal{H}_1(A) \oplus g^{r_a}),\ T_2 \\
V & : alias' = \mathcal{H}_1(g^{r_v r_a}, \mathcal{H}_1(A)),\ K_{av} = \mathcal{H}_2(g^{r_v r_a}, g^{\hat{S}_V r_a}) \\
V \rightarrow A & : g^{r_v},\ E_{K_{av}}(\mathcal{H}_1(g^{r_v}, g^{r_a}, alias', V), T_2),\ T_3 \\
A \rightarrow V & : E_{K_{av}}(Sig_A(g^{r_a}, g^{r_v}, T_2, V), T_3)
\end{array}
$$

In this attack, the malicious server M first pretends to be a foreign server, contacts A's home server H, and claims that A is communicating with M. H then innocently sends A's real identity to M. After that, M launches the deposit-case attack by impersonating A and sending a modified message to V (illustrated as the first message from M to V in the diagram above). This message makes V believe that M is the home server of A while A believes that he has informed V that H is his home server. The attack is then carried out in the same way as described in Sec. 2.

Notice that A and V will still agree on the same key K_{av} when the attack completes. Hence the attack is carried out successfully and will not be discovered by any of the three honest parties.

5 A Self-encryption Based Roaming Protocol

We now describe the third roaming protocol which is found to be vulnerable under the deposit-case attack. The protocol was proposed by Hwang and Chang [7] in 2003. The parties involved in the roaming protocol include a roaming user A, a visiting foreign server V and the user's home server H. It is a symmetric key based protocol which requires a secure symmetric key encryption algorithm such as AES [9]. We use the same set of notations as previous sections. Let K_{ah} denote the long-term secret key shared by A and H. Let K_{vh} denote the long-term secret key shared by V and H. Let f be a secure hash function kept secretly by H. We review their protocol as follows.

1. A generates a random value r_0 and sends the message below to V.

$$A \rightarrow V \ : \ A,\ H,\ E_{K_{ah}}(K_{ah}||r_0)$$

2. V generates a random value r_1 and sends the following to H for verification.

$$V \rightarrow H \ : \ E_{K_{ah}}(K_{ah}||r_0),\ E_{K_{vh}}(A||r_1||t)$$

Here t denotes a timestamp.

3. H decrypts the received message. If t is fresh and K_{ah} is equal to $f(A)$, H sends the following message back to V.

$$H \rightarrow V \ : \ E_{K_{vh}}(r_1),\ C = E_{k_{ah}}(r_0||r_1||V)$$

   Otherwise, H rejects the connection.
4. V decrypts $E_{K_{vh}}(r_1)$. If the decrypted value equals r_1, V sets r_1 as the authentication key K_{auth} and passes C to A. Otherwise, V rejects the connection.
5. On receiving C from V, A checks whether the decrypted message contains r_0. If it is false, A terminates the connection. Otherwise, A also sets r_1 as the authentication key K_{auth} and sends the following to V for authentication.

$$A \rightarrow V \ : \ E_{K_{auth}}(r_1)$$

6. V accepts if the decryption of the incoming message is equal to r_1. Otherwise, V rejects the connection.

After establishing K_{auth} between A and V, the authentication process of all subsequent sessions between these two parties can be simplified in such a way that V does not have to ask H for verifying A. Instead, V can talk directly to A and carry out mutual authentication using K_{auth}. For simplicity and without contradicting any of the assumptions made in [7], we hereafter assume that the lengths of all the random numbers and the identities of A, H and V are equal to the block size of the underlying block cipher.

5.1 Deposit-Case Attack

In the following, we describe the deposit-case attack launched by a malicious server M against the Hwang-Chang roaming protocol reviewed above. The attack is slightly different from the one described in Sec. 4.1. This time, the malicious server M uses the user's home server H as an encryption oracle for generating some message which is expected by A from his home server H.

Let K_{mv} be the long-term secret key shared by M and V and K_{mh} be the long-term secret key shared by M and H. M intercepts the first message from A to V and launches the following attack.

$$\begin{aligned}
&A \rightarrow M : A,\ H,\ E_{K_{ah}}(K_{ah}||r_0)\\
&M \rightarrow V : A,\ M,\ E_{K_{ah}}(K_{ah}||r_0)\\
&V \rightarrow M : E_{K_{ah}}(K_{ah}||r_0),\ E_{K_{mv}}(A||r_1||t)\\
&M \rightarrow H : E_{K_{ah}}(K_{ah}||r_0),\ E_{K_{mh}}(A||r_1||t)\\
&H \rightarrow M : E_{K_{mh}}(r_1),\ C' = E_{K_{ah}}(r_0||r_1||M)\\
&M \rightarrow V : E_{K_{mv}}(r_1),\ C'\\
&V \rightarrow A \ : C'\\
&A \rightarrow V \ : E_{K_{auth}}(r_1)
\end{aligned}$$

Note that C' contains the identity of M. The crucial issue of arguing whether the attack works or not is to determine whether A will check the encrypted identity in C in Step 5 of Sec. 5. This is not mentioned in Hwang-Chang's protocol description [7]. We now consider the two possible cases.

1. If A does not check the identity (i.e. the last component) after decrypting C in Step 5 of Sec. 5, then the attack succeeds. The authors of [7] seem not checking it according to the description of their protocol.
2. If A checks the identity encrypted in C, then there are two sub-cases.
   (a) A finds out which foreign server he is talking to by checking the identity after decrypting C.
   (b) A intends to talk to V at the very beginning when A initiates the protocol execution.

Depending on whether A knows if he is talking to V or M at the very beginning of the protocol execution, in Case 2(a), A believes that he is talking to M after checking the identity in C' while V believes that he is talking to A. In addition, V believes that A's home server is M. Hence in Case 2(a), the deposit-case attack works.

For Case 2(b), A will reject the connection with failure if the deposit-case attack is launched. However, we will see that under some assumptions, the malicious server can still launch the deposit-case attack successfully by modifying the last two message flows of the attack described above slightly. Also, the assumptions made do not contradict any of the restrictions or assumptions made in [7].

In order to make the deposit-case attack work in Case 2(b), the malicious server M has to modify C' in such a way that it contains V as the last component of the corresponding plaintext. However, M does not know the value of K_{ah}.

This can be solved by looking into the implementation details of the underlying block cipher $E_{K_{ah}}$. For simplicity, let us assume that the operation mode [4] of the underlying block cipher is ECB (Electronic Codebook). The computation of C in Hwang-Chang protocol becomes

$$C = E_{K_{ah}}(r_0) \mid\mid E_{K_{ah}}(r_1) \mid\mid E_{K_{ah}}(V).$$

It is also the case for computing C' but with the last component being changed to $E_{K_{ah}}(M)$. Suppose there has been a successful protocol execution among A, V and H before M launches the Deposit-case Attack. Then M gets $E_{K_{ah}}(V)$ from the protocol execution through eavesdropping.

To launch the deposit-case attack, M intercepts the last message flow from V to A and replaces the last component of C' by $E_{K_{ah}}(V)$. We can see that A will accept and complete the protocol without early termination. Also notice that M knows the authentication key K_{auth} as M knows the value of r_1.

In this modification, we have made two assumptions.

1. There is at least one successful protocol execution among A, V and H, and M is able to eavesdrop that protocol execution.
2. The underlying block cipher $E_{K_{ah}}$ is operated in ECB mode.

None of these assumptions contradicts the restrictions or assumptions made in [7]. In addition, the second assumption can also be extended to other commonly used operation modes. It is obvious that if the operation mode is CBC (Cipher

Block Chaining), CFB (Cipher Feedback) or OFB (Output Feedback) [4], the malicious server M can still manage to make all the parties accept and complete the deposit-case attack. This is because M knows the last two components of the plaintext corresponding to C'. They are r_1 and M.

6 Concluding Remarks

We present a new attacking technique against secure roaming protocols. The attack allows a malicious server to make a user believe that the visiting foreign server has been informed about the true identity of the user's home server while the foreign server believes that the malicious server is the home server of the user. We explain that this attack is profitable to the malicious server if the protocol is used by the user to deliver some information of value (such as some electronic cash) to his home server via the foreign server. We also show that there are three roaming protocols proposed previously which are vulnerable to this attack.

There is no universal solution for these three roaming protocols so that they can thwart the deposit-case attack. However, there is a plausible approach which can be adopted when modifying these protocols. The approach is to have the roaming user check if the foreign server has obtained a valid credential from his real home server before accepting the connection. None of the three roaming protocols reviewed in this paper has done this checking. As more and more new roaming-like systems and applications are emerging, we believe that this new attack should be checked against if the corresponding systems and applications have related concerns discussed in this paper.

References

[1] G. Ateniese, A. Herzberg, H. Krawczyk, and G. Tsudik. On traveling incognito. In *Proc. of the IEEE Workshop on Mobile Systems and Applications*, December 1994.

[2] C. Boyd and A. Mathuria. *Protocols for Authentication and Key Establishment.* Springer-Verlag, 2003.

[3] Whitfield Diffie, Paul C. Van Oorschot, and Michael J. Wiener. Authentication and authenticated key exchanges. *Designs, Codes, and Cryptography*, 2(2):107–125, June 1992.

[4] M. Dworkin. *Recommendation for Block Cipher Modes of Operation.* NIST Special Publication 800-38A 2001 Edition, US Department of Commerce / NIST, December 2001.

[5] J. Go and K. Kim. Wireless authentication protocol preserving user anonymity. In *Proc. of the 2001 Symposium on Cryptography and Information Security (SCIS 2001)*, pages 159–164, January 2001.

[6] D. Gollmann. Insider fraud (position paper). In *Security Protocols Workshop*, pages 213–219. Springer, 1998. LNCS 1550.

[7] K. F. Hwang and C. C. Chang. A self-encryption mechanism for authentication of roaming and teleconference services. *IEEE Trans. on Wireless Communications*, 2(2):400–407, March 2003.

[8] Michel Mouly and Marie-Bernadette Pautet. *The GSM System for Mobile Communications*. Published by the authors, 1992.
[9] NIST FIPS PUB 197. *Announcing the ADVANCED ENCRYPTION STANDARD (AES)*, November 2001.
[10] D. Samfat, R. Molva, and N. Asokan. Untraceability in mobile networks. In *Proc. of MobiCom '95*, pages 26–36, 1995.
[11] The Telecommunications Industry Association (TIA). *Mobile Station-Base Station Compatibility Standard for Wideband Spread Spectrum Cellular Systems (TIA/EIA-95-B-99)*, Feb 1999.
[12] V. Varadharajan and Y. Mu. Preserving privacy in mobile communications: A hybrid method. In *IEEE International Conference on Personal Wireless Communications*, pages 532–536, 1997.

Security Requirements for Key Establishment Proof Models: Revisiting Bellare–Rogaway and Jeong–Katz–Lee Protocols*

Kim-Kwang Raymond Choo and Yvonne Hitchcock

Information Security Institute
Queensland University of Technology
GPO Box 2434, Brisbane, QLD 4001, Australia
{k.choo,y.hitchcock}@qut.edu.au

Abstract. We observe that the definitions of security in the computational complexity proof models of Bellare & Rogaway (1993) and Canetti & Krawczyk (2001) require two partners in the presence of a malicious adversary to accept the same session key, which we term a key sharing requirement. We then revisit the Bellare–Rogaway three-party key distribution (3PKD) protocol and the Jeong–Katz–Lee two-party authenticated key exchange protocol $\mathcal{TS}2$, which carry claimed proofs of security in the Canetti & Krawczyk (2001) model and the Bellare & Rogaway (1993) model respectively. We reveal previously unpublished flaws in these protocols where we demonstrate that both protocols fail to satisfy the definition of security in the respective models. We present a new 3PKD protocol as an improvement with a proof of security in the Canetti & Krawczyk (2001) model and a simple fix to the specification of protocol $\mathcal{TS}2$. We also identify several variants of the key sharing requirement and present a brief discussion.

1 Introduction

The treatment of computational complexity analysis for key establishment protocols was made popular by Bellare & Rogaway [5] in 1993, who provided the first formal definition for a model of adversary capabilities with an associated definition of security (which we refer to as the BR93 model in this paper). An extension of the BR93 model was used to analyse a three-party server-based key distribution (3PKD) protocol by Bellare & Rogaway [6], which we refer to as the BR95 model. A more recent revision to the model was proposed in 2000 by Bellare, Pointcheval and Rogaway [4], hereafter referred to as the BPR2000 model. In independent yet related work, Bellare, Canetti, & Krawczyk [3] build on the BR93 model and introduce a modular proof model. However, some drawbacks with this formulation were discovered and this modular proof model was

* This work was partially funded by the Australian Research Council Discovery Project Grant DP0345775.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 429–442, 2005.

subsequently modified by Canetti & Krawczyk [8], and will be referred to as the CK2001 model in this paper.

We observe that the definitions of security in the BR93, BR95, BPR2000 and CK2001 models have two basic requirements, namely: two parties who have completed matching sessions (i.e., partners) are required to accept the same session key (which we term a key sharing requirement) and the key secrecy requirement (also known as implicit key authentication [12, Definition 12.6]) whereby no adversary or anyone other than the legitimate parties involved will learn about the session key at the end of a protocol run. Although the key sharing requirement seems straight-forward, there are actually a number of possible variants of this requirement. We identify several variants of the key sharing requirement and present a brief discussion.

In this work, we revisit the Bellare–Rogaway 3PKD protocol [6] and the authenticated key exchange protocol $\mathcal{TS}2$ due to Jeong, Katz, & Lee [11]. The 3PKD protocol was proven secure in the BR95 model and subsequently Tin, Boyd, & Gonzalez-Nieto [15] provided a claimed proof of security for the same protocol in the CK2001 model. Protocol $\mathcal{TS}2$ carries a claimed proof of security in the BR93 model, but uses a different definition of partnership than that given in the original model description.

We reveal previously unpublished flaws in these protocols, whereby we demonstrate that both protocols violate the definition of security in the CK2001 and BR93 models respectively. The attack we present on the 3PKD protocol is similar to the attack on the Otway–Rees key establishment protocol [13] revealed by Fabrega, Herzog, & Guttman [10], in which they showed that a malicious adversary is able to make the initiator and the responder agree on a different session key by asking a trusted third party (i.e., server) to create multiple session keys in response to the same message.

This paper is organized into the following sections. Section 2 provides an informal overview of the proof models. Section 3 describes the 3PKD protocol, describes an example execution of the protocol to demonstrate how the 3PKD protocol is insecure in the CK2001 model, and presents a new provably-secure 3PKD protocol in the CK2001 model. Section 4 describes protocol $\mathcal{TS}2$, describes an example execution of the protocol to demonstrate how protocol $\mathcal{TS}2$ is insecure in the BR93 model, and provides a simple fix to the protocol specification. Section 5 presents a discussion on the four variants of the key sharing requirement that we have identified. Section 6 presents the conclusions.

2 The Proof Models

2.1 Bellare–Rogaway Models

In the BR93, BR95, and BPR2000 models, the adversary $\mathcal{A}$ is defined to be a probabilistic machine that is in control of all communications between parties by interacting with a set of $\Pi^i_{U_1,U_2}$ oracles (i.e., $\Pi^i_{U_1,U_2}$ is defined to be the i^{th} instantiation of a principal U_1 in a specific protocol run and U_2 is the principal

with whom U_1 wishes to establish a secret key). The oracle queries are shown in Table 1.

$\mathsf{Send}(U_1, U_2, i, m)$	This query to oracle $\Pi^i_{U_1,U_2}$ computes a response according to the protocol specification and decision on whether to accept or reject yet, and returns them to the adversary $\mathcal{A}$. If the client oracle, $\Pi^i_{U_1,U_2}$, has either accepted with some session key or terminated, this will be made known to $\mathcal{A}$.
$\mathsf{Reveal}(U_1, U_2, i)$	The client oracle, $\Pi^i_{U_1,U_2}$, upon receiving this query and if it has accepted and holds some session key, will send this session key back to $\mathcal{A}$. This query is known as a Session-Key Reveal in the CK2001 model.
$\mathsf{Corrupt}(U_1, K_E)$	This query allows $\mathcal{A}$ to corrupt the principal U_1 at will, and thereby learn the complete internal state of the corrupted principal. The corrupt query also gives $\mathcal{A}$ the ability to overwrite the long-lived key of the corrupted principal with any value of her choice (i.e. K_E).
$\mathsf{Test}(U_1, U_2, i)$	This query is the only oracle query that does not correspond to any of $\mathcal{A}$'s abilities. If $\Pi^i_{U_1,U_2}$ has accepted with some session key and is being asked a $\mathsf{Test}(U_1, U_2, i)$ query, then depending on a randomly chosen bit b, $\mathcal{A}$ is given either the actual session key or a session key drawn randomly from the session key distribution.

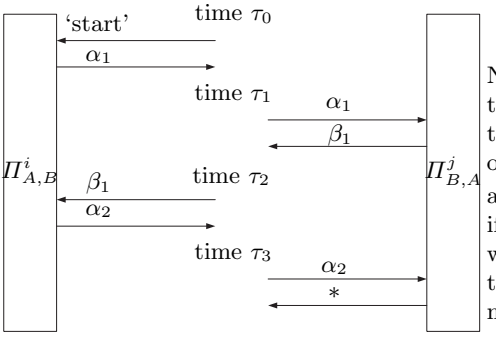
Table 1. Informal description of the oracle queries

BR93 partnership is defined using the notion of matching conversations, where a conversation is defined to be the sequence of messages sent and received by an oracle. The sequence of messages exchanged (i.e., only the Send oracle queries) are recorded in the transcript, T. At the end of a protocol run, T will contain the record of the Send queries and the responses as shown in Figure 1. Definition 1 gives a simplified definition of matching conversations for the case of the protocol shown in Figure 1.

Definition 1 (BR93 Definition of Matching Conversations [5]) *Let n be the maximum number of sessions between any two parties in the protocol run. Run the protocol shown in Figure 1 in the presence of a malicious adversary $\mathcal{A}$ and consider an initiator oracle $\Pi^i_{A,B}$ and a responder oracle $\Pi^j_{B,A}$ who engage in conversations C_A and C_B respectively. $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ are said to be partners if they both have matching conversations, where*

$$C_A = (\tau_0, '\,start', \alpha_1), (\tau_2, \beta_1, \alpha_2)$$
$$C_B = (\tau_1, \alpha_1, \beta_1), (\tau_3, \alpha_2, *), \text{ for } \tau_0 < \tau_1 < \ldots$$

BR95 partnership is defined using a partner function, which uses the transcript to determine the partner of an oracle. However, no explicit definition of partnership was provided in the original paper since there is no single partner function fixed for any protocol. Instead, security is defined predicated on



Note that the construction of conversation shown in Definition 1 depends on the number of parties and the number of message flows. Informally, both $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ are said to be BR93 partners if each one responded to a message that was sent unchanged by its partner with the exception of perhaps the first and last message.

Fig. 1. Matching conversation [5]

the existence of a suitable partner function. However, such a partner definition can easily go wrong. One such example is the partner function described in the original BR95 paper for the 3PKD protocol [6], which was later found to be flawed [9].

BPR2000 partnership is defined using session identifiers (SIDs) where SIDs are suggested to be the concatenation of messages exchanged during the protocol run. In this model, an oracle who has accepted will hold the associated session key, a SID and a partner identifier (PID). Definition 2 describes the definition of partnership in the BPR2000 model.

Definition 2 (BPR2000 Definition of Partnership [4]) *Two oracles, $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$, are partners if, and only if, both oracles have accepted the same session key with the same SID, have agreed on the same set of principals (i.e. the initiator and the responder of the protocol), and no other oracles besides $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ have accepted with the same SID.*

2.2 Canetti–Krawczyk Model

In the CK2001 model, there are two adversarial models, namely the UM and the AM. Let $\mathcal{A}_{\mathrm{UM}}$ denote the adversary in the UM, and $\mathcal{A}_{\mathrm{AM}}$ denote the adversary in the *AM*. The difference between $\mathcal{A}_{\mathrm{AM}}$ and $\mathcal{A}_{\mathrm{UM}}$ lies in their powers. Table 2 provides an informal description of the oracle queries allowed for $\mathcal{A}_{\mathrm{AM}}$ and $\mathcal{A}_{\mathrm{UM}}$.

A protocol that is proven to be secure in the AM can be translated to a provably secure protocol in the UM with the use of an authenticator. We require the definitions of an emulator, and an authenticator as given in Definitions 3 and 4 respectively.

Definition 3 (Definition of an Emulator [3]) *Let π and π' be two n-party protocols where π and π' are protocols in the AM and the UM respectively. π' is said to emulate π if for any $\mathcal{A}_{UM}$, there exists an $\mathcal{A}_{AM}$, such that for all input vectors $\vec{m}$, no polyomial time adversary can distinguish the cumulative outputs*

Session-State Reveal	An oracle, upon receiving this query and if it has neither accepted nor held some session key, will return all its internal state (including any ephemeral parameters but not long-term secret parameters) to the adversary.
Send	Equivalent to the Send query in Table 1. However, $\mathcal{A}_{\mathrm{AM}}$ is restricted to only delay, delete, and relay messages but not to fabricate any messages or send a message more than once.
Session − Key Reveal, Corrupt, and Test queries are equivalent to those queries listed in Table 1.	

Table 2. Informal description of the oracle queries allowed for $\mathcal{A}_{\mathrm{AM}}$ and $\mathcal{A}_{\mathrm{UM}}$

of all parties and the adversary between the AM and the UM with more than negligible probability.

Definition 4 (CK2001 Definition of an Authenticator [8]) *An authenticator is defined to be a mapping transforming a protocol* π_{AM} *in the* AM *to a protocol* π_{UM} *in the* UM *such that* π_{UM} *emulates* π_{AM}*.*

In other words, the security proof of a UM protocol depends on the security proofs of the MT-authenticators used and that of the associated AM protocol. If any of these proofs break down, then the proof of the UM protocol is invalid. Partnership in the CK2001 model can be defined using the notion of matching sessions, as desribed in Definition 5.

Definition 5 (Matching Sessions [8]) *Two sessions are said to be matching if they have the same session identifiers (SIDs) and corresponding partner identifiers (PIDs).*

In the Bellare–Rogaway and the CK2001 models, SIDs are unique and known to everyone (including $\mathcal{A}$). Hence, session keys cannot be included as part of SIDs in the protocols. In the CK2001 model, $\mathcal{A}$ chooses unique SIDs for each pair of participants, although, in practice, SIDs are generally agreed using some unique contributions from each participant.

2.3 Definition of Freshness

Freshness is used to identify the session keys about which $\mathcal{A}$ ought not to know anything because $\mathcal{A}$ has not revealed any oracles that have accepted the key and has not corrupted any principals knowing the key. Definition 6 describes freshness, which depends on the notion of partnership. Note that we do not consider the notion of forward secrecy in this paper, otherwise, the definition of freshness would be slightly different.

Definition 6 (Definition of Freshness) *Oracle* $\Pi^i_{A,B}$ *is fresh (or holds a fresh session key) at the end of execution, if, and only if, (1)* $\Pi^i_{A,B}$ *has accepted with*

or without a partner oracle $\Pi^j_{B,A}$, (2) both $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ oracles have not been sent a Reveal *query (or* Session-State Reveal *in the CK2001 model), and (3) A and B have not been sent a* Corrupt *query.*

2.4 Definition of Security

Security in the four models is defined using the game $\mathcal{G}$, played between $\mathcal{A}$ and a collection of player oracles. $\mathcal{A}$ runs the game $\mathcal{G}$, whose setting is explained in Table 3.

Stage 1:	$\mathcal{A}$ is able to send any oracle queries at will.
Stage 2:	At some point during $\mathcal{G}$, $\mathcal{A}$ will choose a fresh session on which to be tested and send a Test query to the fresh oracle associated with the test session. Depending on the randomly chosen bit b, $\mathcal{A}$ is given either the actual session key or a session key drawn randomly from the session key distribution.
Stage 3:	$\mathcal{A}$ continues making any oracle queries at will but cannot make Corrupt or Reveal queries that trivially expose the test session key.
Stage 4:	Eventually, $\mathcal{A}$ terminates the game simulation and outputs a bit b', which is its guess of the value of b.

Table 3. Setting of game $\mathcal{G}$

Success of $\mathcal{A}$ in $\mathcal{G}$ is quantified in terms of $\mathcal{A}$'s advantage in distinguishing whether $\mathcal{A}$ receives the real key or a random value. $\mathcal{A}$ wins if, after asking a $\mathsf{Test}(U_1, U_2, i)$ query, where $\Pi^i_{U_1,U_2}$ is fresh and has accepted with the same session key, $\mathcal{A}$'s guess bit b' equals the bit b selected during the $\mathsf{Test}(U_1, U_2, i)$ query. Let the advantage function of $\mathcal{A}$ be denoted by $\mathsf{Adv}^{\mathcal{A}}(\mathsf{k})$, where $\mathsf{Adv}^{\mathcal{A}}(\mathsf{k}) = 2 \times Pr[b = b'] - 1$.

Definitions 7, 8, and 9 describe the definition of security for the BR95 model, the BPR2000 model, and both the BR93 and CK2001 models respectively.

Definition 7 (BR95 Definition of Security [6]) *A protocol is secure in the BR95 model if both the following requirements are satisfied:*

1. *When the protocol is run between two oracles $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ in the absence of a malicious adversary, both $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ accept and hold the same session key.*
2. *For all probabilistic, polynomial-time (PPT) adversaries $\mathcal{A}$, $\mathsf{Adv}^{\mathcal{A}}(\mathsf{k})$ is negligible.*

Definition 8 (BPR2000 Definition of Security [4]) *A protocol is secure in the BPR2000 model if both the following requirements are satisfied:*

1. *When the protocol is run between two oracles $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ in the absence of a malicious adversary, both $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ accept and hold the same session key.*

2. *For all PPT adversaries $\mathcal{A}$, (a) the advantage that $\mathcal{A}$ has in violating entity authentication is negligible, and (b) $\mathsf{Adv}^{\mathcal{A}}(\mathsf{k})$ is negligible.*

Definition 9 (BR93 and CK2001 Definitions of Security [5, 8]) *A protocol is secure in the BR93 and CK2001 models if both the following requirements are satisfied:*

1. *When the protocol is run between two oracles $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ in the absence of a malicious adversary, both $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ accept and hold the same session key.*
2. *For all PPT adversaries $\mathcal{A}$, (a) If uncorrupted oracles $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ complete matching sessions, then both $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ must hold the same session key, and (b) $\mathsf{Adv}^{\mathcal{A}}(\mathsf{k})$ is negligible.*

For the BR93 model, if both oracles $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ have accepted, then the probability that orcle $\Pi^j_{B,A}$ does not engage in a matching conversation with oracle $\Pi^i_{A,B}$ is negligible.

3 Bellare–Rogaway 3PKD Protocol

3.1 3PKD Protocol

The 3PKD protocol in Figure 2 involves three parties, a trusted server S and two principals A and B. The notations $\{\cdot\}_{K^{enc}_{AS}}$ and $[\cdot]_{K^{MAC}_{AS}}$ denote the encryption of some message under K^{enc}_{AS} and the computation of a MAC digest under K^{MAC}_{AS} respectively. K^{enc}_{AS} is the encryption key shared between A and S, K^{MAC}_{AS} is the MAC key shared between A and S, and both keys are independent of each other.

A	S	B
$R_A \in_R \{0,1\}^k$	$\xrightarrow{A, R_A}$ Randomly generate SK_{AB} $\xleftarrow{B, R_B}$	$R_B \in_R \{0,1\}^k$
	$\alpha_a = \{SK_{AB}\}_{K^{enc}_{AS}}$	
	$\beta_a = [A, B, R_A, \alpha_a]_{K^{MAC}_{AS}}$	
	$\alpha_b = \{SK_{AB}\}_{K^{enc}_{BS}}$	
	$\beta_b = [A, B, R_B, \alpha_b]_{K^{MAC}_{BS}}$	
	$\xleftarrow{\alpha_a, \beta_a}$ $\quad$ $\xrightarrow{\alpha_b, \beta_b}$	
Decrypt α_a		Decrypt α_b
If β_a verifies true, then		If β_b verifies true, then
Accept SK_{AB}		Accept SK_{AB}

Fig. 2. 3PKD protocol

Tin *et al.* [15] suggest that SIDs can be constructed on the fly using unique contributions from both the initiator and the responder (i.e., $sid_A = (R_A, R_B)$ and $sid_B = (R_A, R_B)$ respectively).

3.2 New Attack on 3PKD Protocol

Figure 3 depicts an example execution of the 3PKD protocol in the presence of a malicious adversary $\mathcal{A}$. Let $\mathcal{A}_U$ denote $\mathcal{A}$ impersonating some user U. At the end of the protocol execution shown in Figure 3, both uncorrupted principals A and B have matching sessions according to Definition 5. However, they have accepted different session keys (i.e., A and B accept SK_{AB} and $SK_{AB,2}$ respectively). This violates requirement 2a of Definition 9.

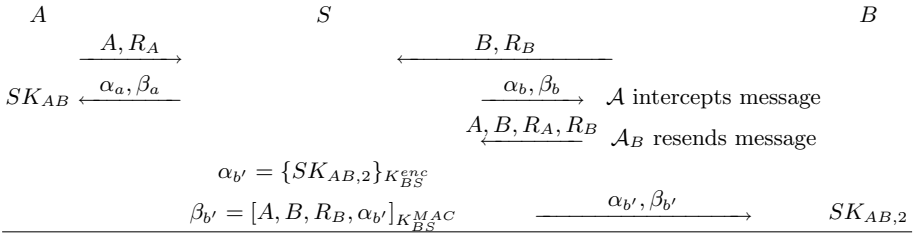


Fig. 3. Execution of 3PKD protocol in the presence of a malicious adversary

We observe that the existing proof fails because the current construction of SIDs does not guarantee uniqueness. Our observation supports the findings of Choo *et al.* [9] that it does not seem possible to define a unique SID in the existing 3PKD protocol.

3.3 A New Provably-Secure 3PKD Protocol in the CK2001 (UM)

A quick fix to the 3PKD protocol is to require the server to store every message processed and not issue different session keys for the same input message received, similar to the approach taken by Backes [1] in his proof of security for the Otway–Rees protocol in the cryptographic library, which has a provably secure cryptographic implementation. However, we argue that this assumption only works well within a confined implementation and will not scale well to a more realistic environment with a large number of participating parties and a substantial level of traffic to any one server.

Another possible fix would be to introduce two extra messages for key confirmation, which would ensure that both parties have the assurance that the other (partner) party is able to compute the (same) session key. However, this would increase the computational load of both the initiator and the responder.

As an improvement, we present an improved provably-secure protocol in the CK2001 model by applying the Canetti–Krawczyk MAC-based MT-authenticator to the Tin–Boyd–Gonzalez-Nieto protocol AM-3PKD. Figures 4 and 5 describe the MAC-based MT-authenticator [8] and the protocol AM-3PKD (which is proven secure in the AM) [15] respectively.

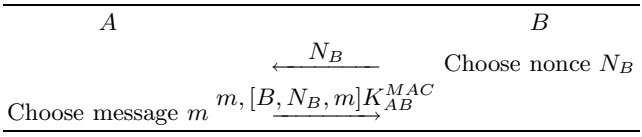


Fig. 4. Canetti–Krawczyk MAC-based MT-authenticator

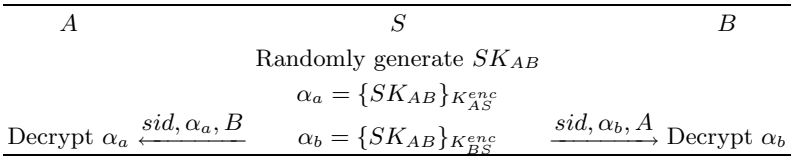


Fig. 5. Tin–Boyd–Gonzalez-Nieto protocol AM-3PKD

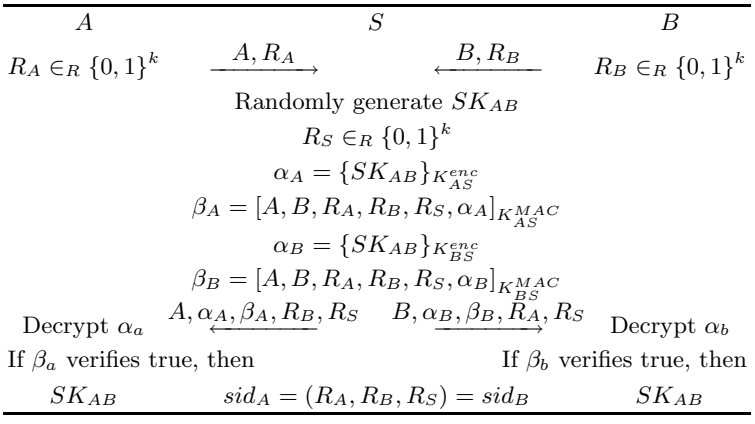


Fig. 6. A new provably-secure 3PKD protocol in the CK2001 (UM)

Figure 6 describes the resultant UM protocol. In this protocol, S will generate a random nonce R_S each time a session key is generated. R_S will be sent together with the associated session key to both A and B together with the contributions by both A and B (i.e., R_A and R_B). Within the new protocol, the only values that A and B can be sure are unique are R_A, R_B, and R_S, and hence SIDs are constructed using these values (i.e., uniqueness of SIDs is ensured).

Intuitively, the attack outlined in Figure 3 will no longer be valid, since a new nonce is generated each time a new session key is generated. Note that there is a subtle difference between our new 3PKD protocol (as shown in Figure 6) and the fix proposed by Choo *et al.* [9]. In their fix, S does not generate a random nonce R_S each time a session key is generated. Hence, the attack outlined in Figure 3 is still valid against their fix. However, their protocol is secure in the BPR2000 model (in which their protocol is proven secure), since the BPR2000 partnership (i.e., Definition 8) requires two parties to have matching SIDs, agreeing PIDs,

and the same session key in order to be partners. Clearly, in the context of our attack, the two oracles are not BPR2000 partners. Hence, the BPR2000 security is not violated.

Table 4 presents a comparison of the computational loads between our new 3PKD protocol and three other similar server-based three-party key establishment protocols. We observe that the three other protocols are unable to satisfy the key share requirement in the presence of a malicious adversary (without making some "impractical" assumption – requiring the server to store every message processed and not issuing different session keys for the same message). From Table 4, we also observe that the computational load of our new 3PKD protocol is comparable to those of the other protocols, yet provides a tighter definition of security (i.e., secure in the sense of Definition 9).

Computational Operation	New 3PKD protocol			Yahalom protocol [7] / Otway-Rees protocol [13] / Bauer–Berson–Feiertag protocol [2]		
	A	B	S	A	B	S
Encryption and Decryption	1	1	2	2/2/1	3/2/1	3/4/2
MAC generation	0	0	2	0	0	0
Messages	4					
Proof of Security	Yes			No, except for the Otway-Rees protocol.		
Security Goal	Key establishment			Key establishment (however, parties who complete matching sessions (partners), are not guaranteed to share the same session key.)		

Table 4. Comparison of the computational loads

4 Jeong–Katz–Lee Protocol $\mathcal{TS}2$

Figure 7 describes protocol $\mathcal{TS}2$ [11]. All arithmetic is performed modulo a large prime p with q being the prime order of g. The protocol uses a different partnering function, as described in Definition 10.

Definition 10 (Modified Definition of Partnership) *Two oracles, $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$, are partners if, and only if, they have agreed on the same set of principals (i.e. the initiator and the responder of the protocol), and no other oracles besides $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ have accepted with the same SID.*

Both the initiator and responder principals, A and B, are assumed to have a public/private key pair (P_A, S_A) and (P_B, S_B) respectively. At the end of the protocol execution, both A and B accept with the session key $SK_{AB} = \mathcal{H}_0(A||B||sid||g^{R_A R_B}||g^{S_A S_B}) = SK_{BA}$.

$A\ (P_A, S_A)$		$B\ (P_B, S_B)$
$R_A \in_R \mathbb{Z}_q$	$\xrightarrow{g^{R_A}}$	$R_B \in_R \mathbb{Z}_q$
$sid_A = g^{R_A}\|\|g^{R_B}$	$\xleftarrow{g^{R_B}}$	$sid_B = g^{R_A}\|\|g^{R_B}$

Fig. 7. Jeong–Katz–Lee protocol $\mathcal{TS}2$

4.1 New Attack on Protocol $\mathcal{TS}2$

Figure 8 desribes the execution of protocol $\mathcal{TS}2$ in the presence of a malicious adversary $\mathcal{A}$, where $\mathcal{A}$ intercepts both messages and sends fabricated messages $g^{R_A}||1$ and $g^{R_B}||1$ to both B and A respectively.

$A\ (P_A, S_A)$		$\mathcal{A}$		$B\ (P_B, S_B)$
$R_A \in_R \mathbb{Z}_q$	$\xrightarrow{g^{R_A}}$		$\xleftarrow{g^{R_B}}$	$R_B \in_R \mathbb{Z}_q$
$sid_A = g^{R_A}\|\|1\|\|g^{R_B}$	$\xleftarrow{1\|\|g^{R_B}}$		$\xrightarrow{g^{R_A}\|\|1}$	$sid_B = g^{R_A}\|\|1\|\|g^{R_B}$

Fig. 8. Execution of protocol $\mathcal{TS}2$ in the presence of a malicious adversary

At the end of the protocol execution shown in Figure 8, both A and B have accepted with $sid_A = g^{R_A}||1||g^{R_B} = sid_B$. Hence, according to Definition 10, both $\Pi_{A,B}^{sid_A}$ and $\Pi_{B,A}^{sid_B}$ are partners since they have accepted with the same SID and $PID(A) = B$ and $PID(B) = A$. However, both $\Pi_{A,B}^{sid_A}$ and $\Pi_{B,A}^{sid_B}$ have accepted with different session keys

$$SK_{AB} = \mathcal{H}_0(A||B||sid_A||(1||g^{R_B})^{R_A}||g^{S_A S_B})$$
$$SK_{BA} = \mathcal{H}_0(A||B||sid_B||(g^{R_A}||1)^{R_B}||g^{S_A S_B}) \neq SK_{BA},$$

in violation of requirement 2a in Definition 9.

A simple fix to protocol $\mathcal{TS}2$ is to include validity checking of the received messages by the recipient, as shown in Figure 9. The validity checking ensures that the messages received by each party are in the group and that the bit lengths of the messages received by each party are correct. Intuitively, the attack outlined in Figure 8 will no longer be valid since the fabricated messages sent by the adversary will fail the validity check. Let $BL(\cdot)$ denote the bit length of some message.

We may speculate that if the protocol designers fail to spot this inadequancy in the specification of their protocols, the protocol implementers are also highly unlikely to spot this inadequancy. Flaws in security protocol proofs or protocol specifications themselves certainly will have a damaging effect on the credibility of provably-secure protocols in the real world [14].

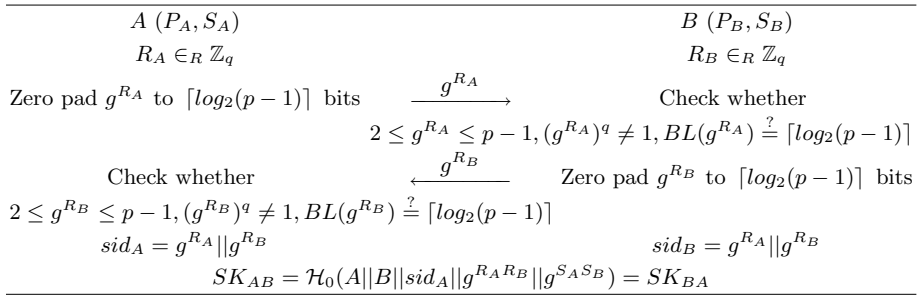


Fig. 9. A possible fix to Jeong–Katz–Lee protocol $\mathcal{TS}2$

5 The Key Sharing Requirement

The key sharing requirement varies between the BR93, BR95, BPR2000, and CK2001 models. In this section, we identify four possible variants of the key sharing requirement, as shown in Table 5.

Variant		Required in
KSR1	Two communicating parties completing matching sessions in the absence of a malicious adversary accept the same session key.	BR95 model.
KSR2	Two communicating parties completing matching sessions in the presence of a malicious adversary accept the same session key.	BR93, BPR2000, CK2001 models.
KSR3	One party is assured that a second (possibly unidentified) party is able to compute a particular secret session key.	Optional in any of the BR93, BR95, BPR2000, or CK2001 models.
KSR4	One party is assured that a second (possibly unidentified) party actually has possession of a particular secret session key.	Not achievable in reductionist proof approach for protocols, as shown below.

Table 5. Variants of key sharing requirement

KSR1 is a completeness requirement, which ensures that a key establishment protocol is behaving correctly. We advocate that KSR2 is a practical functional requirement and depending on the individual implementation, KSR2 requirement can be as important as the key secrecy requirement. Consider the scenario of a real world implementation of one key establishment protocol that does not provide the KSR2 requirement: two partners after completing matching sessions, are unable to share the same session key. From the protocol implementers' perspective, the usefulness (or practicality) of such a key establishment protocol will be questionable.

KSR3 is a weaker version of KSR4, where KSR4 is the key confirmation goal given in [12, Definition 12.7]. KSR4 is generally not achievable in the setting of the reductionist proof approach for protocols for the following reason. In order for one party, A to be assured that a second (possibly unidentified) party, B, actually has possession of the secret session key, A would need to send to B some information derived from the key, such as the encryption of some message with the secret session key. However, in the context of the proof simulation, $\mathcal{A}$ can ask a Send query using the test session key obtained from a Test query, and determine whether the test session key it was given (by the simulator) was real or random. Consequently, such information renders the protocol insecure as $\mathsf{Adv}^{\mathcal{A}}(k)$ will be non-negligible.

We would recommend that the proof models allow different options for the key sharing requirement in their formulation. KSR1 is a minimum requirement as it ensures the (basic) correctness of a protocol, KSR3 implies KSR2, and KSR2 implies KSR1. Protocols proven secure in such a model must indicate which variant of the key sharing requirement is satisfied.

6 Conclusion

A detailed study of the Bellare–Rogaway 3PKD protocol and the Jeong–Katz–Lee protocol $\mathcal{TS}2$ was made. We demonstrated that both protocols fail to achieve the key sharing requirement in the presence of a malicious adversary, in violation of the definition of security in their respective models. Despite the importance of proofs in assuring protocol implementers of the security properties of protocols, we conclude that specifying correct proofs remains a difficult problem.

As an improvement, we presented a new 3PKD protocol with a proof of security in the CK2001 model. A comparison with three existing three-party server-based protocols reveals that the computational load of our new 3PKD protocol is no more than that of the three other protocols, yet ensures that a stronger version of the key sharing requirement is satisfied. We also proposed a simple fix to the specification of protocol $\mathcal{TS}2$ and identified four possible variants of the key sharing requirement. As a result of this work, we would recommend that the proof models for key establishment protocols allow the various options of the key sharing requirement, depending on the individual needs of the protocol implementations and applications.

References

1. Michael Backes. A Cryptographically Sound Dolev-Yao Style Security Proof of the Needham–Schroeder–Lowe Public–Key Protocol. *IEEE Journal on Selected Areas in Communications*, 22(10):2075–2086, 2004.
2. R. K. Bauer, Thomas A. Berson, and Richard J. Feiertag. A Key Distribution Protocol Using Event Markers. *ACM Transactions on Computer Systems*, 1(3):249–255, 1983.

3. Mihir Bellare, Ran Canetti, and Hugo Krawczyk. A Modular Approach to The Design and Analysis of Authentication and Key Exchange Protocols. In Jeffrey Vitter, editor, *30th ACM Symposium on the Theory of Computing - STOC 1998*, pages 419–428. ACM Press, 1998.
4. Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated Key Exchange Secure Against Dictionary Attacks. In Bart Preneel, editor, *Advances in Cryptology – Eurocrypt 2000*, pages 139 – 155. Springer-Verlag, 2000. Volume 1807/2000 of Lecture Notes in Computer Science.
5. Mihir Bellare and Phillip Rogaway. Entity Authentication and Key Distribution. In Douglas R. Stinson, editor, *Advances in Cryptology - Crypto 1993*, pages 110–125. Springer-Verlag, 1993. Volume 773/1993 of Lecture Notes in Computer Science.
6. Mihir Bellare and Phillip Rogaway. Provably Secure Session Key Distribution: The Three Party Case. In F. Tom Leighton and Allan Borodin, editors, *27th ACM Symposium on the Theory of Computing - STOC 1995*, pages 57–66. ACM Press, 1995.
7. Michael Burrows, Martín Abadi, and Roger Needham. A Logic of Authentication. In *ACM Transactions on Computer Systems*, pages 18–36. ACM Press, 1990.
8. Ran Canetti and Hugo Krawczyk. Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels (Extended version available from http://eprint.iacr.org/2001/040/). In Birgit Pfitzmann, editor, *Advances in Cryptology - Eurocrypt 2001*, pages 453–474. Springer-Verlag, 2001. Volume 2045/2001 of Lecture Notes in Computer Science.
9. Kim-Kwang Raymond Choo, Colin Boyd, Yvonne Hitchcock, and Greg Maitland. On Session Identifiers in Provably Secure Protocols: The Bellare-Rogaway Three-Party Key Distribution Protocol Revisited (Extended version available from http://eprint.iacr.org/2004/345). In Blundo Carlo and Stelvio Cimato, editors, *4th Conference on Security in Communication Networks - SCN 2004*, pages 352–367. Springer-Verlag, 2004. Volume 3352/2005 of Lecture Notes in Computer Science.
10. F. Javier Thayer Fabrega, Jonathan C. Herzog, and Joshua D. Guttman. Strand Spaces: Proving Security Protocols Correct. *Journal of Computer Security*, 7:191–230, 1999.
11. Ik Rae Jeong, Jonathan Katz, and Dong Hoon Lee. One-Round Protocols for Two-Party Authenticated Key Exchange. In Markus Jakobsson, Moti Yung, and Jianying Zhou, editors, *Applied Cryptography and Network Security - ACNS 2004*, pages 220–232. Springer-Verlag, 2004. Volume 3089/2004 of Lecture Notes in Computer Science.
12. Alfred J. Menezes, Paul C. Van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. The CRC Press Series On Discrete Mathematics And Its Applications. CRC Press, 1997.
13. Dave Otway and Owen Rees. Efficient and Timely Mutual Authentication. *ACM Operating Systems Review*, 21(1):8–10, 1987.
14. Jacques Stern, David Pointcheval, John Malone-Lee, and Nigel Smart. Flaws in Applying Proof Methodologies to Signature Schemes. In Moti Yung, editor, *Advances in Cryptology - Crypto 2002*, pages 93–110. Springer-Verlag, 2002. Volume 2442/2002 of Lecture Notes in Computer Science.
15. Yiu Shing Terry Tin, Colin Boyd, and Juan Manuel Gonzalez-Nieto. Provably Secure Key Exchange: An Engineering Approach. In *Australasian Information Security Workshop Conference on ACSW Frontiers 2003*, pages 97–104. Australian Computer Society, 2003. Volume 21 of Conferences in Research and Practice in Information Technology.

Group Signature Schemes with Membership Revocation for Large Groups

Toru Nakanishi, Fumiaki Kubooka, Naoto Hamada, and Nobuo Funabiki

Dept. of Communication Network Engineering, Okayama Univ., Japan
nakanisi@cne.okayama-u.ac.jp

Abstract. Group signature schemes with membership revocation have been intensively researched. However, signing and/or verification of some existing schemes have computational costs of $O(R)$, where R is the number of revoked members. Existing schemes using a dynamic accumulator or a similar technique have efficient signing and verifications with $O(1)$ complexity. However, before signing, the signer has to modify his secret key with $O(N)$ or $O(R)$ complexity, where N is the group size. Therefore, for larger groups, signers suffer from enormous costs. On the other hand, an efficient scheme for middle-scale groups with about 1,000 members is previously proposed, where the signer need not modify his secret key. However this scheme also suffers from heavy signing/verification costs for larger groups. In this paper, we adapt the middle-scale scheme to the larger groups. At the sacrifice of the group manager's cost, our signing/verification has only $O(1)$ complexity.

1 Introduction

Group signature schemes allow a group member to anonymously sign a message on behalf of a group, where a group manager controls the membership of members. In case of disputes, the group manager (or a designated third party) can trace the signer from a targeted signature. The main stream of group signature schemes is certificate type [1,6,2,7,13,12,3,4]. In this type, the manager issues a joining member a membership certificate, and the group signature is a non-interactive zero-knowledge proof of knowledge of the membership certificate. This type of scheme has an advantage that the size of signatures and public keys is independent from the group size. However, the membership revocation is not easy. The simplest revocation method is that the manager reissues the certificates of all members except the revoked member. However, the other members' loads are enormous.

Some schemes [6,2,7,13,12,3,4] deal with the membership revocation. However, in the schemes [6,2,4], signing and/or verification requires a computation with $O(R)$ complexity, where R is the number of revoked members.

In [7], an approach using a dynamic accumulator is proposed, which is followed by [13] with the efficiency improvement. The accumulator allows the manager to hash a large set of effective certificates into a short value. In the group signature, the signer has to prove that own certificate is accumulated into the

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 443–454, 2005.

short value. Therefore, signing/verification is $O(1)$ w.r.t. N and R, where N is the group size. However, whenever making a signature, the signer has to modify a secret key for the accumulator. The modification requires the certificates of joining and removed members since the last time he signed. To obtain the certificates, the signer must fetch the certificates of all joining and removed members from a public directory with the list of the certificates, as pointed out in [2]. In addition, the computation of the modification is linear on the number of joining and removed members since the last time he signed. In the worst case, it is $O(N)$. In [3], a group signature scheme from bilinear maps is proposed, where a similar revocation approach is adopted. This scheme also requires secret key modification. The number of modifications is the number of revocations since the last time he signed. Thus, in the worst case, the signer is required $O(R)$ computation. The public revocation information consists of $O(R)$ certificates.

On the other hand, a scheme with efficient revocation suitable for middle-scale groups with about 1,000 members is proposed in [12]. For such groups, signing/verification are independent from N and R, and in addition this scheme has advantages: (1) Any signer does not need to modify the secret key, (2) the signer fetches only 1 public information with about 1,000 bits when the signer signs, and (3) the manager only performs 1 bit operation for the public information in a revocation. However, for larger groups, signing/verification requires the cost that depends on N/ℓ_n, where ℓ_n is a security parameter of strong RSA assumption, which is currently 1,024 or 2,048.

In this paper, we adapt the middle-scale scheme [12] to larger groups. At first, we propose a basic scheme, where the large group is partitioned into middle-scale sub-groups. The public information of the sub-groups falls into a reasonable size, and thus signing/verification cost also becomes $O(1)$. In addition, the signer need not modify his secret key. In the basic scheme, the identity of the sub-group is concealed by using a sub-group certificate that ensures membership situation of the sub-group and a zero-knowledge-like proof of the certificate. However, since the group manager has to re-compute all sub-group certificates whenever a join or revocation happens, the load of the manager is heavy for much larger groups. Therefore, we also propose an extended scheme using a tree structure of sub-group certificates, where the manager only has to re-compute less certificates.

2 Model

We show a model of group signature scheme with membership revocation [12]. Hereafter, MM (Membership Manager) stands for the group manager who has the authority to add a member into a group, and OM (Opening Manager) stands for a trusted third party who has the authority to disable the anonymity.

Definition 1. *A group signature scheme with membership revocation consists of the following procedures:*

Setup: *MM and OM generate the general public key and their secret keys.*
Join: *MM issues a joining user a* membership certificate *for a* membership secret *chosen by the user. In addition, MM authentically publishes a* public

membership information *that reflects the current members in the group such that the joining user belongs to the group.*

Membership revocation: *MM authentically publishes the public membership information that reflects the current members in the group such that the removed user does not belong to the group. Note that OM, unrelated members and even the removed member do not participate in this procedure.*

Sign: *Given a message, a group member with a membership secret and its membership certificate generates the signature for the message w.r.t. the public key and public membership information.*

Verify: *A verifier checks whether a signature for a message is made by a member in the group w.r.t. the public key and public membership information.*

Open: *Given a signature, OM with his secret specifies the identity of the signer.*

Definition 2. *A* secure *group signature scheme with membership revocation satisfies the following properties:*

Unforgeability: *Only a member in the group, which is indicated by the public membership information, can generate a valid signature.*

Coalition-resistance: *Colluding members including removed members cannot generate a valid membership certificate that MM did not generate, even if the members adaptively obtained valid certificates from MM.*

Anonymity: *Given a signature, it is infeasible that anyone, except the signer and OM, identifies the signer.*

Unlinkability: *Given two signatures, it is infeasible that anyone, except the signers and OM, determines whether the signatures ware made by the same signer.*

No framing: *Even if MM, OM, and members collude, they cannot sign on behalf of a non-involved member.*

Traceability: *OM is always able to open a valid signature and identify the signer.*

3 Preliminaries

3.1 Assumptions and Cryptographic Tools

Our group signature schemes utilize Camenisch and Lysyanskaya's ordinary signature scheme [8] whose security is based on the strong RSA assumption. Let $n = pq$ be an RSA modulus for safe primes p, q (i.e., $p = 2p' + 1, q = 2q' + 1$, and p, q, p', q' are prime), and let $QR(n)$ be the set of quadratic residues modulo n, that is, the cyclic subgroup of $\mathbb{Z}_n^*$ generated by an element of order $p'q'$. The strong RSA assumption on $QR(n)$ means that finding $(u \in QR(n), e \in \mathbb{Z}_{>1})$ s.t. $u^e = z \pmod{n}$ on inputs $(n, z \in QR(n))$ is infeasible. Furthermore, note that $QR(n)$ also satisfies the DDH assumption. Including the signature scheme, we use cryptographic tools on $QR(n)$ as follows. Hereafter, we use notations: Let $[a, a + d]$ be the integer interval of all integers int such that $a \leq int \leq a + d$,

for an integer a and a positive integer d. Let $[a, a+d)$ be the integer interval of all int such that $a \leq int < a+d$, and let $(a, a+d)$ be the interval for all int such that $a < int < a+d$. $S_1 \| S_2$ indicates the concatenation of S_1 and S_2 as bit strings.

Camenisch-Lysyanskaya (CL) Signature Scheme for Blocks of Messages.

Key generation: Let $\ell_n, \ell_m, \ell_s, \ell_e, \ell$ be security parameters s.t. $\ell_s \geq \ell_n + \ell_m + \ell$, $\ell_e \geq \ell_m + 2$ and ℓ is sufficiently large (e.g., 160). The secret key consists of safe primes p, q, and the public key consists of $n := pq$ of length ℓ_n and $a_1, \ldots, a_L, b, c \in_R QR(n)$, where L is the number of blocks.

Signing: Given messages $m_1, \ldots, m_L \in [0, 2^{\ell_m})$, choose $s \in_R [0, 2^{\ell_s})$ and a random prime e from $(2^{\ell_e - 1}, 2^{\ell_e})$. Compute A s.t.

$$A := (a_1^{m_1} \cdots a_L^{m_L} b^s c)^{1/e}.$$

The signature is (s, e, A).

Verification: Given messages $m_1, \ldots, m_L \in [0, 2^{\ell_m})$ and the signature (s, e, A), check $A^e = a_1^{m_1} \cdots a_L^{m_L} b^s c$ and $e \in (2^{\ell_e - 1}, 2^{\ell_e})$.

Commitment Scheme. A commitment scheme on $QR(n)$ under the strong RSA assumption is proposed by Damgård and Fujisaki [10]. The following is a slightly modified version due to Camenisch and Lysyanskaya [8].

Key generation: The public key consists of a secure RSA modulus n of length ℓ_n, h from $QR(n)$, and g from the group generated by h.

Commitment: For the public key, input x of length ℓ_x, and randomness $r \in_R \mathcal{Z}_n$, the commitment C is computed as $C := g^x h^r$.

3.2 Signatures of Knowledge

As main building blocks, we use signatures converted by Fiat-Shamir heuristic [11] from honest-verifier zero-knowledge proofs of knowledge, which are called signatures of knowledge. We abbreviate them as SPKs. The SPKs are denoted as $SPK\{(\alpha, \beta, \ldots) : R(\alpha, \beta, \ldots)\}(m)$, which means the signature for message m by a signer with the secret knowledge $\alpha, \beta, \ldots$ satisfying the relation $R(\alpha, \beta, \ldots)$.

The proofs used in our scheme show the relations among secret representations of elements in $QR(n)$ with unknown order. The SPK of a representation is proposed in [10]. We furthermore use the SPK of representations with equal parts [9], SPK of a representation with parts in intervals [5,9], and SPK of a representation with a non-negative part [5].

***SPK* of representation:** An SPK proving the knowledge of a representation of $C \in QR(n)$ to the bases $g_1, g_2, \ldots, g_t \in QR(n)$ on message m is denoted as

$$SPK\{(\alpha_1, \ldots, \alpha_t) : C = g_1^{\alpha_1} \cdots g_t^{\alpha_t}\}(m).$$

SPK **of representations with equal parts:** An SPK proving the knowledge of representations of $C, C' \in QR(n)$ to the bases $g_1, \ldots, g_t \in QR(n)$ on message m, where the representations include equal values as parts, is denoted as

$$SPK\{(\alpha_1, \ldots, \alpha_u) : C = g_{i_1}^{\alpha_{j_1}} \cdots g_{i_v}^{\alpha_{j_v}} \wedge C' = g_{i'_1}^{\alpha_{j'_1}} \cdots g_{i'_{v'}}^{\alpha_{j'_{v'}}}\}(m),$$

where indices $i_1, \ldots i_v, i'_1, \ldots i'_{v'} \in \{1, \ldots, t\}$ refer to the bases $g_1, \ldots, g_t$, and indices $j_1, \ldots j_v, j'_1, \ldots j'_{v'} \in \{1, \ldots, u\}$ refer to the secrets $\alpha_1, \ldots, \alpha_u$.

SPK **of representation with parts in intervals:** An SPK proving the knowledge of a representation of $C \in QR(n)$ to the bases $g_1, \ldots, g_t \in QR(n)$ on message m, where the i-th part lies in an interval $[a, a + d]$, is denoted as

$$SPK\{(\alpha_1, \ldots, \alpha_t) : C = g_1^{\alpha_1} \cdots g_t^{\alpha_t} \wedge \alpha_i \in [a, a + d]\}(m).$$

For this SPK, two types are known. One is due to Boudot [5], where it is assured that the knowledge exactly lies in the interval. However, this SPK needs the computations of about 10 normal SPKs of a representation. Another type appears in [9] for example, where the integer the prover knows in fact lies in the narrower interval than the interval the proved knowledge lies in. However, its efficiency is comparable to that of the normal SPK. For $\alpha_i \in [a, a + d]$ in fact, this SPK proves the knowledge in $[a - 2^{\tilde{\ell}}d, a + 2^{\tilde{\ell}}d]$, where $\tilde{\ell}$ is a security parameter derived from the challenge size and from the security parameter controlling the statistical zero-knowledge-ness (in practice, $\tilde{\ell} \approx 160$).

In this paper, for simplicity, we describe our schemes using the former protocol [5], since a design using the latter efficient protocol must address the expansion of the intervals. Although this expansion can be easily addressed as in [8], it may disturb a clear grasp of the essence of our schemes. However, in the later efficiency consideration, we evaluate the efficiency of our schemes using the efficient SPK of [9].

SPK **of representation with non-negative part:** An SPK proving the knowledge of a representation of $C \in QR(n)$ to the bases $g_1, \ldots, g_t \in QR(n)$ on message m, where the i-th part is not negative integer, is denoted as

$$SPK\{(\alpha_1, \ldots, \alpha_t) : C = g_1^{\alpha_1} \cdots g_t^{\alpha_t} \wedge \alpha_i \geq 0\}(m).$$

As for this, since we need to prove that the knowledge is exactly 0 and over, we adopt the SPK due to Boudot [5].

The interactive versions of these SPKs are also used. The interactive ones are denoted by substituting PK for SPK, such as $PK\{\alpha : y = g^\alpha\}$.

The knowledge of CL signature and messages can be proved by a combined SPK [8]. Let (s, e, A) be a CL signature on messages $m_1, \ldots, m_L$. Then, this SPK on a message m is denoted as

$$SPK\{(A, e, s, m_1, \ldots, m_L) : A^e = a_1^{m_1} \ldots a_L^{m_L} b^s c\}(m).$$

This SPK is computed by combining commitments and the above SPKs, and the signer needs 5 multi-exponentiations and the verifier needs 3 multi-exponentiations.

Moreover, for secret values proved by an SPK and public values, the polynomial equations and inequations among the values can be proved by a combined SPK, which is used in [12] for example. This type of SPK is simply denoted using the equation, such as $SPK\{(\alpha, \beta, \gamma, \delta) : \alpha\beta + \gamma = \delta\}(m)$. The inequation is similar.

4 Basic Scheme

4.1 Idea

The following intuitive discussion omits the open mechanism, which is the same as the previous scheme [12]. The proposed scheme is an extension of the previous scheme that adopts CL signatures as membership certificates. Let $Sign(m_1, m_2)$ (resp., $Sign(m_1, m_2, m_3)$) be the CL signature on messages m_1, m_2 (resp., m_1, m_2, m_3). Then, the previous membership certificate is $Sign(x, m)$ issued from MM to the member, where x is the member's secret and $m = 2^{i-1}$ for member ID i (i.e, only the i-th bit of m is 1). On the other hand, MM manages a public membership information $\tilde{m}$, where, for all $\tilde{\imath}$, the $\tilde{\imath}$-th bit of $\tilde{m}$ is 1 iff the member $\tilde{\imath}$ is valid (i.e., not revoked). In this setting, the group signature of member i on message M is the following SPK.

$$SPK\{(x, m, v, m_U, m_L) : v = Sign(x, m) \wedge \tilde{m} = m_U(2m) + m + m_L \\ \wedge (0 \leq m_L \leq m - 1)\}(M).$$

A revoked member cannot prove the above relation. For the details, refer to [12]. Note that the length of m is the group size, and thus the cost of the SPK depends on the group size, when m is larger than the security parameter ℓ_n.

In the proposed scheme for larger groups, the group is divided into sub-groups with less than ℓ_n members. We consider that a group of at most N members is partitioned into K sub-groups with ℓ_m members ($N \leq K\ell_m$). The sub-groups are indexed by ID j ($1 \leq j \leq K$). Then, each sub-group has public membership information $\tilde{m}_j$, where the i-th bit of $\tilde{m}_j$ indicates that member i in the sub-group j is valid (resp., invalid) if the bit is 1 (resp., 0). The membership certificate of member i in the sub-group j is modified into $Sign(x, m, j)$, where $m = 2^{i-1}$.

In addition, we introduce certificates of $\tilde{m}_j$ for all sub-groups, since $\tilde{m}_j$ should be hided in the group signature to conceal sub-group j. The certificate, called sub-group certificate, is $Sign(\tilde{m}_j, j, t)$, where t is a public time parameter. At every join (resp., revocation) of member i in the sub-group j, MM increases t and publishes a sub-group certificate $Sign(\tilde{m}_j, j, t)$, where the i-th bit of $\tilde{m}_j$ becomes 1 (resp., 0). Moreover, for the current time t, MM re-computes $Sign(\tilde{m}_{\tilde{\jmath}}, \tilde{\jmath}, t)$ for other all sub-groups $\tilde{\jmath}$ and publishes them.

The group signature is as follows.

$$SPK\{(x, m, v, m_U, m_L, \tilde{m}_j, j, \tilde{v}) : v = Sign(x, m, j) \wedge \tilde{m}_j = m_U(2m) + m + m_L \\ \wedge (0 \leq m_L \leq m - 1) \wedge \tilde{v} = Sign(\tilde{m}_j, j, t)\}(M),$$

for the current time t. A revoked member i cannot prepare his sub-group certificate $Sign(\tilde{m}_j, j, t)$, where the i-th bit of $\tilde{m}_j$ is 1. Thus, the revocation is achieved. Since $\tilde{m}_j, j$ are also concealed by the SPK, the signature remains anonymous and unlinkable. On the other hand, the computation cost is small, since the lengths of $\tilde{m}_j$ and m are less than ℓ_n.

4.2 Proposed Protocols

Setup. Let ℓ_n be a security parameter. Then, MM sets up CL signature scheme. Namely, MM computes two $(\ell_n/2)$-bit safe primes p, q and $n := pq$, and chooses $a_1, a_2, a_3, b,\ c \in_R QR(n)$. Furthermore, he sets up the commitment scheme on $QR(n)$ to generate g and h. He publishes $(n, a_1, a_2, a_3, b, c, g, h)$ as the public key, and keeps (p, q) as the secret key. For CL signature scheme, security parameters $\ell_m, \ell_e, \ell_s, \ell$ are set s.t. $\ell_s \geq \ell_n + \ell_m + \ell$ and $\ell_e \geq \ell_m + 2$. To simplify the description, we introduce interval notations as follows: Define $\mathcal{S} = [0, 2^{\ell_s}), \mathcal{E} = (2^{\ell_e - 1}, 2^{\ell_e}), \mathcal{M} = [0, 2^{\ell_m})$.

The group of at most N members is partitioned into K sub-groups with ℓ_m members. The sub-groups are indexed by ID j $(1 \leq j \leq K)$. Then, each sub-group has public membership information $\tilde{m}_j$, where the i-th bit of $\tilde{m}_j$ indicates that the i-th member in the sub-group j is valid (resp., invalid) if the bit is 1 (resp., 0). At first, MM clears $\tilde{m}_j$ as 0 (i.e., no member), and also clears a time parameter t as 0. Moreover, on the public directory, in addition to $t, \tilde{m}_j$, MM publishes the certificates of $\tilde{m}_j$, $Sign(\tilde{m}_j, j, t)$, which are computed as follows: MM chooses $\tilde{e}_j \in_R \mathcal{E}$ and $\tilde{s}_j \in_R \mathcal{S}$, and computes $\tilde{A}_j := (a_1^{\tilde{m}_j} a_2^j a_3^t b^{\tilde{s}_j} c)^{1/\tilde{e}_j}$. $Sign(\tilde{m}_j, j, t)$ is $(\tilde{e}_j, \tilde{s}_j, \tilde{A}_j)$ s.t. $\tilde{A}_j^{\tilde{e}_j} = (a_1^{\tilde{m}_j} a_2^j a_3^t b^{\tilde{s}_j} c)$.

OM sets up the ElGamal cryptosystem on $QR(n)$, i.e., OM chooses a secret key $x_{OM} \in_R \{0,1\}^{\ell_n}$ and publishes the public key $y = g^{x_{OM}}$.

Join. A user U joins the group as member i in sub-group j as follows.

1. U obtains a membership certificate $Sign(x, m, j)$ on his secret and his membership information m, as follows:
   (a) U chooses $x \in_R \mathcal{M}$, and sends $C := a_1^x$ to MM. Moreover, U conducts $PK\{\alpha : C = a_1^\alpha \wedge \alpha \in \mathcal{M}\}$ with MM.
   (b) MM chooses $e \in_R \mathcal{E}$, $s \in_R \mathcal{S}$, and computes $A := (C a_2^m a_3^j b^s c)^{1/e}$ for $m = 2^{i-1}$. Then, MM returns (e, s, A) to U as the membership certificate.

   $Sign(x, m, j)$ is (e, s, A) s.t. $A^e = (a_1^x a_2^m a_3^j b^s c)$.
2. For the group j, MM evolves $\tilde{m}_j := \tilde{m}_j + 2^{i-1}$. For other $\tilde{j}$, MM lets $\tilde{m}_{\tilde{j}}$ untouched. In addition, MM evolves $t := t + 1$. Then, for all $\tilde{j}$, MM evolves the sub-group certificate $Sign(\tilde{m}_{\tilde{j}}, \tilde{j}, t)$ on $\tilde{m}_{\tilde{j}}$, $\tilde{j}$ and t.
3. On the public directory, MM publishes the current time t, the sub-group membership information $\tilde{m}_{\tilde{j}}$ and the certificates $Sign(\tilde{m}_{\tilde{j}}, \tilde{j}, t)$ for all $\tilde{j}$.

Revoke. Member i in sub-group j is revoked as follows. For the group j, MM evolves $\tilde{m}_j := \tilde{m}_j - 2^{i-1}$. Then, MM evolves and publishes the time t, the sub-group membership informations $\tilde{m}_j$ and the certificates $Sign(\tilde{m}_j, j, t)$ for all j, as well as the join protocol.

Group sign and verify. Member i in sub-group j signs a message M at the public current time t, as follows. At first, in addition to t, the member fetches all sub-group membership informations $\tilde{m}_{\tilde{j}}$ and the certificates $Sign(\tilde{m}_{\tilde{j}}, \tilde{j}, t)$ at the time t. Then, he computes $T_1 := g^{r_e}$ and $T_2 := y^{r_e} a_1^x$, where $r_e \in_R \{0,1\}^{\ell_n}$, and the following $SPK\ V$.

$$\begin{aligned} V := SPK\{(x, r_e, A, e, m, \tilde{j}, s, \tilde{m}_j, \tilde{m}_U, \tilde{m}_L, \tilde{A}_j, \tilde{e}_j, \tilde{s}_j) : \\ T_1 = g^{r_e} \wedge T_2 = y^{r_e} a_1^x \wedge A^e = a_1^x a_2^m a_3^j b^s c \\ \wedge\ \tilde{m}_j = \tilde{m}_U(2m) + m + \tilde{m}_L \wedge (0 \le \tilde{m}_L \le m-1) \\ \wedge\ \tilde{A}_j^{\tilde{e}_j} = a_1^{\tilde{m}_j} a_2^j a_3^t b^{\tilde{s}_j} c \wedge (e, \tilde{e}_j \in \mathcal{E}) \wedge (x, m, j, \tilde{m}_j \in \mathcal{M})\}(M). \end{aligned}$$

The group signature on M is (T_1, T_2, V).

The verification is to check the current time t and to verify the $SPK\ V$.

Open. By decrypting the ElGamal ciphertext (T_1, T_2) in the signature, OM produces $C = a_1^x$. This can be linked to the identity of the signer via the transcript of the join protocol.

4.3 Security

We first concentrate in the unforgeability and coalition-resistance. We can prove that, even if valid members collude, any attacking group excluding MM cannot forge any new certificates, under the strong RSA assumption. This proof is the same as [12]. On the other hand, $SPK\ V$ in any group signature ensures the ownership of certificates $Sign(x, m, j)$ and $Sign(\tilde{m}_j, j, t)$. Thus, only a member who is issued $Sign(x, m, j)$ can compute the group signature. Consider a revoked member i in sub-group j. Then, at time t after the revocation, there does not exist $Sign(\tilde{m}_j, j, t)$ for $\tilde{m}_j$ s.t. the i-th bit of $\tilde{m}_j$ is 1. As shown in [12], the revoked member cannot prove $\tilde{m}_j = m_U(2m) + m + m_L$ and $0 \le m_L \le m-1$ in the $SPK\ V$, when the i-th bit of $\tilde{m}_j$ is 0. We turn to attacks using invalid certificates. The attacker may use old $Sign(\tilde{m}_j, j, t)$ where his bit of $\tilde{m}_j$ is valid. However, t rejects the old certificate. The attacker may use $Sign(\tilde{m}_{\tilde{j}}, \tilde{j}, t)$ for $\tilde{j} \neq j$. However, the SPK forces the singer to prove that $Sign(\tilde{m}_j, j, t)$ corresponds to his membership certificate $Sign(x, m, j)$ via j. Thus, he cannot use $Sign(\tilde{m}_{\tilde{j}}, \tilde{j}, t)$.

As well as [12], the anonymity and unlinkability holds, since a group signature consists of an ElGamal ciphertext and SPK revealing only t. This means that the signature does not reveal even the information on the sub-group. The other requirements are derived from the previous scheme [12].

5 Extended Scheme

5.1 Idea

The problem of the basic scheme in the previous section is that MM re-computes all sub-group certificates at every join and revocation. Here, using a tree structure of sub-group certificates, we propose an extended scheme, where MM re-computes only parts of sub-group certificates.

For simplicity, we show the case of a k-ary tree with 2 levels, for a constant k. The extension to more levels is easy. We consider that a group of at most N members is partitioned into k^2 sub-groups with ℓ_m members ($N \leq k^2\ell_m$). The sub-groups are indexed by 2 IDs j_1, j_2 ($1 \leq j_1, j_2 \leq k$). Consider the following tree. Let $\mathcal{N}_0$ be the root, and let $\mathcal{N}_{j_1}$ be the j_1-th child of $\mathcal{N}_0$ for $1 \leq j_1 \leq k$. Similarly, let $\mathcal{N}_{j_1 j_2}$ be the j_2-th child of $\mathcal{N}_{j_1}$. $\mathcal{N}_{j_1 j_2}$ is assigned to sub-group $j_1 j_2$.

Then, the sub-group membership information $\tilde{m}_{j_1 j_2}$ of the sub-group $j_1 j_2$ indicates the validity of membership, as the basic scheme. The sub-group certificate is modified into two certificates $Sign(\tilde{m}_{j_1 j_2}, j_1 \| j_2, u_{j_1})$ and $Sign(u_{j_1}, t)$, where t is the same time parameter as the basic scheme (we consider t as the time of the root node.) and another parameter u_{j_1} consists of its parent node ID j_1 and the time t_{j_1} of $\mathcal{N}_{j_1}$. When a sub-group membership information $\tilde{m}_{j_1 j_2}$ is evolved, MM evolves the times of nodes on the path from the root to $\mathcal{N}_{j_1 j_2}$, i.e., times t and t_{j_1} (Other $t_{\tilde{j}_1}$ remains). Then, MM re-computes only the certificates w.r.t. t and t_{j_1}, namely $Sign(\tilde{m}_{j_1 \tilde{j}_2}, j_1 \| \tilde{j}_2, u_{j_1})$ for $1 \leq \tilde{j}_2 \leq k$ and $Sign(u_{\tilde{j}_1}, t)$ for $1 \leq \tilde{j}_1 \leq k$. Thus, MM re-computes $2k$ certificates only, which is more efficient than the basic scheme.

The group signature is modified into the following.

$$
\begin{aligned}
SPK\{&(x, m, v, m_U, m_L, \tilde{m}_{j_1 j_2}, j, u_{j_1}, \tilde{v}_{j_1 j_2}, \tilde{v}_{j_1}) : \\
&v = Sign(x, m, j) \wedge \tilde{m}_{j_1 j_2} = m_U(2m) + m + m_L \wedge (0 \leq m_L \leq m - 1) \\
&\wedge \tilde{v}_{j_1 j_2} = Sign(\tilde{m}_{j_1 j_2}, j, u_{j_1}) \wedge \tilde{v}_{j_1} = Sign(u_{j_1}, t)\}(M),
\end{aligned}
$$

for the current time t. In this SPK, j indicates $j_1 \| j_2$, which is concealed together with $\tilde{m}_{j_1 j_2}$ and u_{j_1} because of anonymity and unlinkability. Since the time t is connected with $\tilde{m}_{j_1 j_2}$ via u_{j_1} by the certificates, a revoked member cannot conduct this SPK. The details are shown later.

5.2 Proposed Protocols

Setup. The setups of CL signature scheme, commitment scheme, and ElGamal cryptosystem are the same as the basic scheme.

As mentioned ahead, assume that a group of at most N members is partitioned into k^2 sub-groups with ℓ_m members, and the sub-groups are indexed by j_1, j_2. Then, the sub-group membership information $\tilde{m}_{j_1 j_2}$ of the sub-group indicates the validity of membership, as the basic scheme. At first, MM clears all $\tilde{m}_{j_1 j_2}$ as 0. In addition to time parameter t, other time parameters t_{j_1} are all initialized as 0. Define $u_{j_1} := j_1 \| t_{j_1}$.

Then, MM publishes certificates $Sign(\tilde{m}_{j_1 j_2}, j_1 \| j_2, u_{j_1})$ for all $\tilde{m}_{j_1 j_2}$. Additionally, MM publishes certificates $Sign(u_{j_1}, t)$ for all j_1.

Join. A user U joins the group as member i in a sub-group $j_1 j_2$ as follows.

1. As well as the basic scheme, U obtains $Sign(x, m, j_1 \| j_2)$ from MM.
2. MM evolves $\tilde{m}_{j_1 j_2} := \tilde{m}_{j_1 j_2} + 2^{i-1}$, for the sub-group $j_1 j_2$. For other $\tilde{j}_1 \tilde{j}_2$, MM lets $\tilde{m}_{j_1 j_2}$ untouched. In addition, MM evolves $t := t + 1$, and $t_{j_1} := t_{j_1} + 1$. Then, for all $1 \leq \tilde{j}_2 \leq k$, MM evolves the certificate $Sign(\tilde{m}_{j_1 \tilde{j}_2}, j_1 \| \tilde{j}_2, u_{j_1})$. In addition, MM evolves $Sign(u_{\tilde{j}_1}, t)$ for all $1 \leq \tilde{j}_1 \leq k$.
3. On the public directory, MM publishes the current time $t, t_{\tilde{j}_1}$, the sub-group membership informations $\tilde{m}_{\tilde{j}_1 \tilde{j}_2}$ and the certificates $Sign(\tilde{m}_{\tilde{j}_1 \tilde{j}_2}, \tilde{j}_1 \| \tilde{j}_2, u_{\tilde{j}_1})$ and $Sign(u_{\tilde{j}_1}, t)$, for all $1 \leq \tilde{j}_1, \tilde{j}_2 \leq k$.

Revoke. Member i in sub-group $j_1 j_2$ is revoked as follows. For sub-group $j_1 j_2$, MM evolves $\tilde{m}_{j_1 j_2} := \tilde{m}_{j_1 j_2} - 2^{i-1}$. Then, MM evolves and publishes the times, the sub-group membership informations and the certificates, as the join protocol.

Group sign and verify. Member i in a sub-group $j_1 j_2$ signs a message M at the public current time t, as follows. At first, in addition to t, the member fetches all $t_{\tilde{j}_1}$, all sub-group membership informations $\tilde{m}_{\tilde{j}_1 \tilde{j}_2}$ and the certificates $Sign(\tilde{m}_{\tilde{j}_1 \tilde{j}_2}, \tilde{j}_1 \| \tilde{j}_2, u_{\tilde{j}_1})$ and $Sign(u_{\tilde{j}_1}, t)$. Then, the member computes $T_1 := g^{r_e}$ and $T_2 := y^{r_e} a_1^x$, where $r_e \in_R \{0,1\}^{\ell_n}$, and the following $SPK\ V$.

$$
\begin{aligned}
V := SPK\{(x, r_e, A, e, m, j, s, \tilde{m}_{j_1 j_2}, \tilde{m}_U, \tilde{m}_L, u_{j_1}, \tilde{A}_{j_1 j_2}, \tilde{e}_{j_1 j_2}, \tilde{s}_{j_1 j_2}, \tilde{A}_{j_1}, \tilde{e}_{j_1}, \tilde{s}_{j_1}) : \\
T_1 = g^{r_e} \wedge T_2 = y^{r_e} a_1^x \wedge A^e = a_1^x a_2^m a_3^j b^s c \\
\wedge\ \tilde{m}_{j_1 j_2} = \tilde{m}_U (2m) + m + \tilde{m}_L \wedge (0 \leq \tilde{m}_L \leq m - 1) \\
\wedge\ \tilde{A}_{j_1 j_2}^{\tilde{e}_{j_1 j_2}} = a_1^{\tilde{m}_{j_1 j_2}} a_2^j a_3^{u_{j_1}} b^{\tilde{s}_{j_1 j_2}} c \wedge \tilde{A}_{j_1}^{\tilde{e}_{j_1}} = a_1^{u_{j_1}} a_2^t b^{\tilde{s}_{j_1}} c \\
\wedge\ (e, \tilde{e}_{j_1 j_2}, \tilde{e}_{j_1} \in \mathcal{E}) \wedge (x, m, j, \tilde{m}_{j_1 j_2}, u_{j_1} \in \mathcal{M})\}(M).
\end{aligned}
$$

The group signature on M is (T_1, T_2, V).

The verification is to check the current time t and to verify the $SPK\ V$.

Open. This is the same as the basic scheme.

5.3 Security

We show only unforgeability and coalition-resistance, since others are the same as the basic scheme. The unforgeability and coalition-resistance of certificates are the same. Consider a revoked member i in sub-group $j_1 j_2$. Then, at time t after the revocation, there does not exist $Sign(u_{j_1}, t)$ s.t. t_{j_1} in u_{j_1} is the time before the revocation. Moreover, at time t_{j_1} after the revocation, for the corresponding u_{j_1}, there does not exist $Sign(\tilde{m}_{j_1 j_2}, j_1 \| j_2, u_{j_1})$ s.t. the i-th bit of $\tilde{m}_{j_1 j_2}$ is 1. Thus, at time t after the revocation, there are no $Sign(\tilde{m}_{j_1 j_2}, j_1 \| j_2, u_{j_1})$ and $Sign(u_{j_1}, t)$ s.t. the i-th bit of $\tilde{m}_{j_1 j_2}$ is 1. Therefore, using the valid certificates at time t after the revocation, the revoked one cannot prove the SPK. The attack using the old certificates, and attack using the certificates with different $\tilde{j}_1 \tilde{j}_2$ are both protected as the basic scheme. In addition, an attacking group may use $Sign(\tilde{m}_{j_1 j_2}, j_1 \| j_2, u_{j_1})$ s.t. i-th bit of $\tilde{m}_{j_1 j_2}$ is 1 and $Sign(u_{\tilde{j}_1}, t)$ for different node $\mathcal{N}_{\tilde{j}_1}$. However, since $\mathcal{N}_{\tilde{j}_1} \neq \mathcal{N}_{j_1}$ implies $u_{\tilde{j}_1} \neq u_{j_1}$, this is also impossible.

6 Efficiency

For large groups, we evaluate the efficiency on the signing/verification computation (including the secret key modification), MM's join/revocation computation, and the fetched public information size. Here, we compare our schemes to scheme [7] with efficient signing/verification and scheme [12] with short public information. We omit the comparison to other schemes due to the following reasons. The efficiency of scheme [13] is similar to [7]. The scheme [4] also has similar efficiency except that the secret key modification depends on R instead of N. Other schemes [6,2,4] require $O(R)$ computation of the signer and/or verifier. Thus, these are not suitable to large groups.

Our protocols in the previous sections are described using the exact but inefficient SPK for intervals [5]. Since the versions using efficient SPK [9] are easily obtained in the same way as [8,12], we evaluate the efficiency for the efficient versions here.

At first, we discuss the signing/verification computation costs. In the basic scheme, signature generation (resp., verification) requires 5 (resp., 3) multi-exponentiations in addition to 31 (resp, 18) multi-exponentiations in the previous scheme [12], and thus 36 (resp., 21) multi-exponentiations in total. In the extended scheme, signature generation requires 41 multi-exponentiations and verification requires 24 multi-exponentiations in total. If a tree with L levels is adopted, the signature generation cost increases to $31 + 5L$ multi-exponentiations, which has a trade-off with MM's cost to re-compute certificates. The L factor is decided after the MM's cost and the public information size are discussed. On the other hand, [7] requires 14 and 8 multi-exponentiations for signing and verification, respectively. However, note that [7] furthermore requires to modify member's secret key before the signature generation, which needs $O(N)$ multi-exponentiations in the worst case. Although [12] needs 31 and 18 multi-exponentiations only, each multi-exponentiation requires longer times than ours for large groups, since $\tilde{m}$ (and thus m, m_U, m_L) becomes long. Thus, the signing and verification costs of [12] are evaluated as $O(N/\ell_n)$. Our scheme does not have such a problem, since the size of $\tilde{m}_j$ or $\tilde{m}_{j_1 j_2}$ is less than ℓ_n. Therefore, only our schemes are suitable to large groups, if small L is used.

Next, we consider the cost of MM at each join or revocation. In the basic scheme, MM has to perform roughly $K = N/\ell_n$ multi-exponentiations. In the extended scheme, the cost is reduced to roughly $2k = 2\sqrt{N/\ell_n}$. In case of using a tree with L levels, the cost is roughly $L(N/\ell_n)^{1/L}$. For example, if $N = 1,000,000$ and $\ell_n = 1,024$, the basic scheme requires about 1,000 multi-exponentiations, and the extended one requires about 60 and 30 multi-exponentiations for levels 2 and 3, respectively. Although the costs in [7,12] are $O(1)$ and very low, the cost of our extended scheme is also reasonable, even if $L = 2, 3$.

Finally, we discuss the size of the public information fetched by the signer. [7] requires that each signer fetches all the certificates. Thus, roughly $\ell_n N$ bits are required. [12] reduces it to only N bits of $\tilde{m}$. Our schemes requires mainly $\tilde{m}_j$ and the certificates (time parameters are neglected). The total size of all $\tilde{m}_j$ is N bits. The size of a certificate is about $3\ell_n$ and the number of sub-

groups is about N/ℓ_n (In the extended scheme, we count only certificates on the lowest level of the tree, since upper certificates are drastically less). Thus, all the certificates has roughly $3N$ bits, and the total size of the public information is roughly $4N$ bits. Although this is slightly larger than [12], it is sufficiently smaller than [7]. In the example of $N = 1,000,000$ and $\ell_n = 1,024$, the sizes of [7], [12], and ours are roughly 100 MBytes, 100 KBytes, and 500 KBytes, respectively. Therefore, our scheme is sufficiently reasonable. However, for huger groups with more than 10,000,000 members, all the schemes suffer from huge public information beyond Mbytes. Thus, the limit of the group size is roughly 10,000,000. On the other hand, even if $N = 10,000,000$, MM's join/revocation cost in our extended scheme is only about 200 and 60 multi-exponentiations in case of $L = 2, 3$, respectively. Therefore, since the factor L can be set to the small values, we conclude that the signing/verification cost of our schemes is $O(1)$ within the limit of $N = 10,000,000$.

References

1. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik, "A practical and provably secure coalition-resistant group signature scheme," Proc. CRYPTO2000, LNCS 1880, pp.255–270, 2000.
2. G. Ateniese, D. Song, and G. Tsudik, "Quasi-efficient revocation of group signatures," Proc. FC2002, LNCS 2357, pp.183–197, 2003.
3. D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," Proc. CRYPTO2004, LNCS 3152, pp.41–55, 2004.
4. D. Boneh and H. Shacham, "Group signatures with verifier-local revocation," Proc. ACM-CCS'04, pp.168–177, 2004.
5. F. Boudot, "Efficient proofs that a committed number lies in an interval," Proc. EUROCRYPT2000, LNCS 1807, pp.431–444, 2000.
6. E. Bresson and J. Stern, "Group signature scheme with efficient revocation," Proc. PKC2001, LNCS 1992, pp.190–206, 2001.
7. J. Camenisch and A. Lysyanskaya, "Dynamic accumulators and application to efficient revocation of anonymous credentials," Proc. CRYPTO2002, LNCS 2442, pp.61–76, 2002.
8. J. Camenisch and A. Lysyanskaya, "A signature scheme with efficient protocols," Proc. SCN'02, LNCS 2576, pp.268–289, 2002.
9. J. Camenisch and M. Michels, "Separability and efficiency for generic group signature schemes," Proc. CRYPTO'99, LNCS 1666, pp.413–430, 1999.
10. I. Damgård and E. Fujisaki, "A statistically-hiding integer commitment scheme based on groups with hidden order," Proc. ASIACRYPT2002, LNCS 2501, pp.125–142, 2002.
11. A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," Proc. CRYPTO'86, LNCS 263, pp.186–194, 1987.
12. T. Nakanishi and Y. Sugiyama, "A group signature scheme with efficient membership revocation for reasonable groups," Proc. ACISP2004, LNCS 3108, pp.336–347, 2004.
13. G. Tsudik and S. Xu, "Accumulating composites and improved group signing," Proc. ASIACRYPT2003, LNCS 2894, pp.269–286, 2003.

An Efficient Group Signature Scheme from Bilinear Maps

Jun Furukawa[1,2] and Hideki Imai[2]

[1] NEC Corporation, 1753, Shimonumabe, Nakahara, Kawasaki 211-8666, Japan
j-furukawa@ay.jp.nec.com
[2] Institute of Industrial Science, The University of Tokyo, 4-6-1, Komaba, Meguro, Tokyo 153-8505, Japan
imai@iis.u-tokyo.ac.jp

Abstract. We propose a new group signature scheme which is secure if we assume the Decision Diffie-Hellman assumption, the q-Strong Diffie-Hellman assumption, and the existence of random oracles. The proposed scheme is the most efficient among the all previous group signature schemes in signature length and in computational complexity.

1 Introduction

A group signature scheme, first proposed by Chaum and van Heyst [13] and followed by [1,2,6,8,10,11,12,27], allows each member of a group to sign messages on behalf of the group without revealing his own identity. The scheme also realizes a special authority that can identify actual signers in case of dispute. Group signatures have many applications in which user anonymity is required such as in anonymous credential systems [2], identity escrow [21,20], voting and bidding [1], and electronic cash systems.

Although earlier group signature schemes required large computational cost and long signatures, recently proposed schemes, such as the one proposed by Ateniese et al. in [1], are very efficient. In particular, Boneh, Boyen, and Shacham [7], Nguyen and Safavi-Naini [27], and Camenisch and Lysyanskaya [10] proposed very efficient group signature schemes based on bilinear maps. Currently, the most efficient construction is the one proposed in [7]. The signature length of the scheme in [7] is 42% and 38% of those of [27] and [10] respectively. The computational cost for the scheme in [7] is also smaller than those of [27] and [10][3].

This paper proposes a novel group signature scheme based on bilinear maps. Our scheme is more efficient than any of the previous schemes. Moreover, our scheme requires fewer assumptions than the scheme in [7], which is the most efficient among the previous schemes.

[3] The heaviest computation in these schemes is computation of a bilinear map such as Tate pairing. As shown in Table 10 in [17], its computational cost is smaller than that of computation of full-exponent RSA.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 455–467, 2005.

Our approach to the construction of a group signature scheme is similar to that adopted by Boneh et al. in [7]. They used a set of three groups $\mathcal{G}_1, \mathcal{G}_2$, and $\mathcal{G}_T$ of the same prime order p such that there exists a bilinear map from $\mathcal{G}_1 \times \mathcal{G}_2$ to $\mathcal{G}_T$. Each group member has a pair comprising a membership certificate and a membership secret with which he signs on behalf of the group. The membership certificate and membership secret are elements of $\mathcal{G}_1$ and $\mathbb{Z}/p\mathbb{Z}$. For a special authority to identify actual signers from group signatures in their scheme, signers are required to attach an encryption of a part of the membership certificate which is an element of $\mathcal{G}_1$. Because of the existence of the bilinear map, their scheme is not able to simply use ElGamal encryption scheme for this purpose. Hence, they introduced a new encryption scheme called "linear encryption scheme" based on a new assumption called the Decision Linear Diffie-Hellman (DLDH) assumption. This encryption scheme is more complex than the ordinary ElGamal type encryption scheme.

The main difference between our approach and that in [7] is that we use a group $\mathcal{G}$ of the same order p in addition to the three groups $\mathcal{G}_1, \mathcal{G}_2$, and $\mathcal{G}_T$ such that the Decision Diffie-Hellman (DDH) problem on $\mathcal{G}$ is difficult to solve. For a special authority to identify actual signers from group signatures in our scheme, signers are required to attach an encryption of the exponentiation of the membership secret in $\mathcal{G}$. Because this exponentiation to be encrypted is in $\mathcal{G}$, we can apply a simple ElGamal type encryption scheme. This makes our scheme more efficient and requires fewer assumptions than the scheme in [7].

For the groups $\mathcal{G}_1, \mathcal{G}_2$, and $\mathcal{G}_T$ and their associated bilinear map, we can use, for example, the elliptic curve proposed by [26] (MNT curve) and Tate pairing. The choice of such a curve makes it possible to express elements in $\mathcal{G}_1$ by a short string. Although the number of such curves are found in [26] is small, more MNT curves are found in [30]. Therefore, since we can easily find an elliptic curve of the same given order p as $\mathcal{G}$ with practically high probability by using a complex multiplication method, finding a desired set of $(\mathcal{G}_1, \mathcal{G}_2, \mathcal{G})$ is practical.

As a result, our signature lengths are, respectively, 83%, 36%, and 32% of those of signatures in [7], [27], and [10] if we choose groups so that elements of $\mathcal{G}_1$, $\mathcal{G}_T$, and $\mathcal{G}$ can be expressed in 171, 1020, and 171 bit strings respectively. Although we cannot a present precise estimation of the computational cost since it depends on the choice of groups, our scheme requires less computational cost than any of the schemes in [7,27,10]. The security of our scheme depends on the DDH assumption, the Strong Diffie-Hellman (SDH) assumption, and the existence of random oracles. We do not present how to revoke group members. However, the revocation mechanisms described in [7] can be also applied to our system. In our scheme, group members are able to determine their secret key when they join the group, which enables them to join many groups using the same secret key. This property may reduce operational cost when there are many groups. The scheme in [27] does not have such a property. (The scheme in [10] does.)

Our paper is organized as follows. Section 2 describes the model and security requirements of the group signature scheme and notation and complexity

assumptions. Section 3 proposes our group signature scheme, and Section 4 discusses its security. Section 5 compares our scheme with the previous schemes.

2 Background

2.1 Model of Group Signature Scheme

Let $b \leftarrow \mathsf{AL}(a)$ denote an algorithm AL, where its input is a and its output is b. Let $\langle c, d\rangle \leftarrow \mathsf{IP}_{A,B}\langle a, b\rangle$ denote an interactive protocol IP between A and B, where private inputs to A and B are, respectively, a and b, and outputs of A and B are, respectively, c and d.

The model of the group signature scheme is defined as follows. In this model, we do not consider revocation for the sake of simplicity.

Definition 1. *Players in the group signature scheme are a membership manager MM, a tracing manager TM, a group member U and a verifier V. $k \in \mathbb{N}$ is a security parameter.*

A group signature scheme $\mathcal{GS}$ *consists of the following five algorithms and one interactive protocol.* (M-KeyGen, T-KeyGen, Join, Sign, Verify, Open),

- *A probabilistic key generation algorithm for MM that, given a security parameter 1^k, outputs a* membership public key *mpk and a* membership secret key *msk.*

$$(msk, mpk) \leftarrow \mathsf{M\text{-}KeyGen}(1^k)$$

- *A probabilistic key generation algorithm for TM that, given* mpk*, outputs a* tracing public key *tpk and a* tracing secret key *tsk.*

$$(tsk, tpk) \leftarrow \mathsf{T\text{-}KeyGen}(mpk)$$

- *An interactive member registration protocol for the MM and a user U. MM is given* mpk, msk*, the user's identity U*[4]*, and a list of all group members $\mathcal{L}$. U is given* mpk*. If the interaction was successful, U outputs a* membership certificate $cert_U$*, a* membership secret sk_U*, and an* identifier $ider_U$ *and MM adds a pair $(U, ider_U)$ to $\mathcal{L}$ and outputs this revised $\mathcal{L}$.*

$$\langle(\mathcal{L}), (cert_U, sk_U, ider_U)\rangle \leftarrow \mathsf{Join}_{MM,U}\langle(\mathcal{L}, U, mpk, msk), (mpk)\rangle$$

- *A probabilistic signature generation algorithm for a U that, given* mpk*, tpk, $cert_U, sk_U$, and a* message m*, outputs a group signature gs on the message m.*

$$gs \leftarrow \mathsf{Sign}(mpk, tpk, cert_U, sk_U, m)$$

[4] We use the same notation U for a user and the identity of this user U.

– *A deterministic signature verification algorithm for any V that, given* mpk, tpk, m, *and* gs, *returns either* acc *or* rej. *Here,* acc *and* rej *represent, respectively, an acceptance and a rejection of the signature.*

$$acc/rej \leftarrow \mathsf{Verify}(mpk, tpk, m, gs)$$

We say that a group signature gs on m is valid *if* $acc \leftarrow \mathsf{Verify}(mpk, tpk, m, gs)$.

– *A deterministic signer tracing algorithm for the TM that, given* mpk, tpk, tsk, m, *and* gs, *outputs $\perp$ if gs on m is not valid. Otherwise, it outputs $(U, proof)$, where proof assures the validity of the result U. If the algorithm cannot find the actual signer in $\mathcal{L}$, the algorithm outputs $\perp'$ instead of U.*

$$\perp/(U/\perp', proof) \leftarrow \mathsf{Open}(mpk, tpk, tsk, m, gs, \mathcal{L})$$

2.2 Security Requirements

Security requirements for group signature schemes that includes a dynamically changing membership and separation of group manager into membership manager and tracing manager are proposed in [4,16,18]. In [4], Bellare et al. called these requirements Traceability, Anonymity, and Non-frameability. Requirements in [16,18] are basically the same.

Roughly, Traceability guarantees that no one except the MM is able to successfully add a new member to the group. Anonymity guarantees that no one except the TM is able to successfully identify actual signers of signatures. Non-Frameability guarantees that no one except each member is able to successfully create a signature which will be linked to his own identity when opened by the TM.

We give short description of these requirements with minor modifications, which do not consider revocation for the sake of simplicity.

Definition 2. *(***Traceability***) Let $\mathcal{GS}$ be a group signature scheme, and let $\mathcal{A}$ be an algorithm. We consider the following experiment that returns* 0/1. *Here, we assume that* Join *protocols are executed only sequentially.*

Experiment $\mathbf{Exp}^{\mathrm{Tr}}_{\mathcal{GS},\mathcal{A}}(k)$
  $(mpk, msk) \leftarrow \mathsf{M\text{-}KeyGen}(1^k)$
  $(tpk, State) \leftarrow \mathcal{A}(mpk)$
  $Cont \leftarrow \mathsf{true}$
  While $Cont = \mathsf{true}$ do
    $\langle(\mathcal{L}), (State)\rangle \leftarrow \mathsf{Join}_{MM,A}\langle(\mathcal{L}, U, mpk, msk), (mpk, State)\rangle$
  EndWhile
  $(m, gs) \leftarrow \mathcal{A}(State)$
  If $rej \leftarrow \mathsf{Verify}(mpk, tpk, m, gs)$ then return 0
  If $(\perp', proof) \leftarrow \mathsf{Open}(mpk, tpk, tsk, m, gs, \mathcal{L})$ then return 1
  Return 0

A group signature scheme $\mathcal{GS}$ has traceability *property if for all probabilistic, polynomial-time machines $\mathcal{A}$,*

$$\Pr[\mathbf{Exp}^{\mathrm{Tr}}_{\mathcal{GS},\mathcal{A}}(k) = 1]$$

is negligible in k.

Definition 3. *(***Anonymity***) Let $\mathcal{GS}$ be a group signature scheme, let $b \in \{0,1\}$, and let $\mathcal{A}$ be an algorithm. We consider the following experiment that returns* $0/1$.

Experiment $\mathbf{Exp}^{\mathrm{An}}_{\mathcal{GS},\mathcal{A}}(k,b)$
$(mpk, State) \leftarrow \mathcal{A}(1^k)$
$(tpk, tsk) \leftarrow \mathsf{T\text{-}KeyGen}(mpk)$
$(State, (cert_0, sk_0), (cert_1, sk_1), m) \leftarrow \mathcal{A}^{\mathsf{Open}(mpk,tpk,tsk,\cdot,\cdot,\cdot)}(State, tpk)$
$gs \leftarrow \mathsf{Sign}(mpk, tpk, cert_b, sk_b, m)$
$(b' \in \{0,1\} \leftarrow \mathcal{A}^{\mathsf{Open}(mpk,tpk,tsk,\cdot,\cdot,\cdot)}(State, gs)$
If $\mathcal{A}$ did not query Open oracle with (m, gs) after gs is given, then return b'
Return 0

A group signature scheme $\mathcal{GS}$ has anonymity *property if for all probabilistic polynomial-time machines $\mathcal{A}$,*

$$\Pr[\mathbf{Exp}^{\mathrm{An}}_{\mathcal{GS},\mathcal{A}}(k,0) = 1] - \Pr[\mathbf{Exp}^{\mathrm{An}}_{\mathcal{GS},\mathcal{A}}(k,1) = 1]$$

is negligible in k.

Definition 4. *(***Non-Frameability***) Let $\mathcal{GS}$ be a group signature scheme, and let $\mathcal{A}$ be an algorithm. We consider the following experiment that returns* $0/1$.

Experiment $\mathbf{Exp}^{\mathrm{NF}}_{\mathcal{GS},\mathcal{A}}(k)$
$(mpk, tpk, State) \leftarrow \mathcal{A}(1^k)$
$\langle State, (cert_U, sk_U, ider_U)\rangle \leftarrow \mathsf{Join}_{A,U}\langle State, mpk\rangle$
If the tuple $(cert_U, sk_U, ider_U)$ is not valid then return 0
$(m, gs, \mathcal{L}) \leftarrow \mathcal{A}^{\mathsf{Sign}(mpk,tpk,cert_U,sk_U,\cdot)}(State)$
$\mathcal{L} \leftarrow \mathcal{L} \cup \{(U, ider_U)\}$
If $rej \leftarrow \mathsf{Verify}(mpk, tpk, m, gs)0$ then return 0
If $(U, proof) \leftarrow \mathsf{Open}(mpk, tpk, tsk, m, gs, \mathcal{L})$ and m was not queried by $\mathcal{A}$ to the signing oracle Sign then return 1
Else return 0

A group signature scheme $\mathcal{GS}$ has Non-frameability *property if for all probabilistic polynomial-time machines $\mathcal{A}$,*

$$\Pr[\mathbf{Exp}^{\mathrm{NF}}_{\mathcal{GS},\mathcal{A}}(k) = 1]$$

is negligible in k.

2.3 Notation and Complexity Assumption

Let $\mathcal{G}_{1k}, \mathcal{G}_{2k}$, and $\mathcal{G}_k$ be a cyclic group of length k prime order p. We omit index k if not confusing. Let G_1, G_2, and G be, respectively, generators of $\mathcal{G}_1$, $\mathcal{G}_2$, and $\mathcal{G}$. Let ψ be an isomorphism from $\mathcal{G}_2$ to $\mathcal{G}_1$, with $\psi(G_2) = G_1$. Let e be a bilinear map $e : \mathcal{G}_1 \times \mathcal{G}_2 \to \mathcal{G}_T$. Let $\mathcal{H}$ be a hash function that maps string to $\mathbb{Z}/p\mathbb{Z}$.

Definition 5. (Decision Diffie-Hellman assumption) *Let the Decision Diffie-Hellman problem in $\mathcal{G}_k$ be defined as follows: given 4-tuple $(G, [a]G, [b]G, [c]G) \in (\mathcal{G}_k)^4$ as input, output 1 if $c = ab$ and 0 otherwise. An algorithm $\mathcal{A}$ has advantage $\epsilon(k)$ in solving the Decision Diffie-Hellman problem in $\mathcal{G}_k$ if*

$$|\Pr[A(G, [a]G, [b]G, [ab]G) = 1] - \Pr[A(G, [a]G, [b]G, [c]G) = 1]| \geq \epsilon(k)$$

where the probability is taken over the random choice of generator G in $\mathcal{G}_k$, of $(a, b, c) \in (\mathbb{Z}/p\mathbb{Z})^3$, and of the random tape of $\mathcal{A}$.

We say that the Decision Diffie-Hellman assumption holds in $\{\mathcal{G}_k\}_{k\in\mathbb{N}}$ if no polynomial-time algorithm has advantage $\epsilon(k)$ non-negligible in k in solving the Decision Diffie-Hellman problem in $\mathcal{G}_k$.

Definition 6. (Strong Diffie-Hellman Assumption) *Let the q-Strong Diffie-Hellman Problem (q-SDH) in $(\mathcal{G}_{1k}, \mathcal{G}_{2k})$ be defined as follows: given a $(q+2)$-tuple $(G_1, G_2, [\gamma]G_2, [\gamma^2]G_2, \ldots, [\gamma^q]G_2) \in \mathcal{G}_{1k} \times (\mathcal{G}_{2k})^{q+1}$ as input, output a pair $([1/(x+\gamma)]G_1, x)$ where $x \in \mathbb{Z}/p\mathbb{Z}$. An algorithm $\mathcal{A}$ has advantage $\epsilon(k)$ in solving the q-SDH problem in $(\mathcal{G}_{1k}, \mathcal{G}_{2k})$ if*

$$\Pr[A(G_1, G_2, [\gamma]G_2, \ldots, [\gamma^q]G_2) = ([1/(x+\gamma)]G_1, x)] \geq \epsilon(k),$$

where the probability is taken over the random choice of generator G_2 in $\mathcal{G}_{2k}$ (with $G_1 = \psi(G_2)$), of $\gamma \in \mathbb{Z}/p\mathbb{Z}$, and of the random tape of $\mathcal{A}$.

We say that the Strong Diffie-Hellman (SDH) assumption holds in $\{(\mathcal{G}_{1k}, \mathcal{G}_{2k})\}_{k\in\mathbb{N}}$ if no polynomial-time algorithm has advantage $\epsilon(k)$ non-negligible in k in solving the q-SDH problem in $(\mathcal{G}_{1k}, \mathcal{G}_{2k})$ for q polynomial of k.

The SDH assumption is proposed and proved to hold in generic bilinear groups in [6]. This assumption is a variant of an assumption proposed by Mitsunari et al. in [25].

3 Proposed Group Signature Scheme

Now we will present our efficient group signature scheme.

M-KeyGen

Given 1^k, M-KeyGen chooses $\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_T$ such that its order p is of length k and then randomly chooses $w \in_R \mathbb{Z}/p\mathbb{Z}$ and $(H, K) \in_R (\mathcal{G}_1)^2$ and generates $Y = [w]G_2$. Then, M-KeyGen outputs

$$(msk, mpk) := (w, (p, \mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_T, e, \mathcal{G}, G_1, G_2, G, \psi, \mathcal{H}, Y, H, K))$$

Here, some of the symbols are interpreted as binary strings that describe those symbols. For example, $\mathcal{G}$ expresses the string of the document that specifies group $\mathcal{G}$.

T-KeyGen

Given mpk, T-KeyGen first randomly chooses $(s,t) \in_R (\mathbb{Z}/p\mathbb{Z})^2$. Next, T-KeyGen generates $(S,T) = ([s]G, [t]G)$. Finally, T-KeyGen outputs

$$(tsk, tpk) := ((s,t),(S,T)) .$$

$\mathsf{Join}_{MM,U}$

1. – MM is given group member list $\mathcal{L}$, an identity of a user U, mpk, and msk.
   – A user U is given mpk.
2. U randomly chooses $sk_U := x_U \in_R \mathbb{Z}/p\mathbb{Z}$ and $z'_U \in_R \mathbb{Z}/p\mathbb{Z}$ and generates

$$ider_U := Q_U = [x_U]G \ , \ H_U = [x_U]H + [z'_U]K$$

and sends $(Q_U, H_U)$to MM[5].
Then, U proves in zero-knowledge to MM the knowledge of x_U and z'_U as follows. Although the protocol given here is only honest verifier zero-knowledge, from this we can construct a black-box zero-knowledge protocol using the technique presented in [24]. We still assume that Join protocols are executed in a sequential manner (or concurrently but with an appropriate timing-constraint [14]).

(i) U randomly chooses $(x'_U, z') \in_R (\mathbb{Z}/p\mathbb{Z})^2$ and generates

$$Q'_U = [x'_U]G \ , \ H'_U = [x'_U]H + [z']K$$

and sends them to MM.

(ii) MM sends U randomly chosen $c_U \in_R \mathbb{Z}/p\mathbb{Z}$.

(iii) U generates

$$r_U = c_U x_U + x'_U \ , \ s_U = c_U z'_U + z'$$

and sends (r_U, s_U) to MM.

(iv) MM checks that the following equations hold:

$$[r_U]G = [c_U]Q_U + Q'_U \ , \ [r_U]H + [s_U]K = [c_U]H_U + H'_U$$

3. The MM randomly chooses $(y_U, z''_U) \in_R (\mathbb{Z}/p\mathbb{Z})^2$ and generates

$$A_U = [1/(w + y_U)](G_1 - H_U - [z''_U]K)$$

and sends (A_U, y_U, z''_U) to U. The MM adds an entry $(U, ider_U) = (U, Q_U)$ to its group member list $\mathcal{L}$.

[5] U needs to sign on Q_U to prove that U agreed to be a group member; we omit this process for the sake of simplicity.

4. U generates its membership certificate as

$$cert_U := (A_U, y_U, z_U) = (A_U, y_U, z'_U + z''_U).$$

U checks that the following equation holds:

$$e(A_U, Y + [y_U]G_2) \cdot e([x_U]H, G_2) \cdot e([z_U]K, G_2) = e(G_1, G_2).$$

5. – MM outputs the revised $\mathcal{L}$.
   – U outputs $(cert_U, sk_U, ider_U) = ((A_U, y_U, z_U), x_U, Q_U)$.

Remark 1. Publishing $(cert_U, ider_U)$ which MM is able to obtain does not compromise the security of the system.

Sign

1. Sign is given $mpk, tpk, cert_U, sk_U$, and m.
2. Sign randomly chooses $(r, q) \in_R (\mathbb{Z}/p\mathbb{Z})^2$ and generates

$$B = A_U + [q]K \ , \quad U = [x_U + r]G \ , \quad V = [r]S \ , \quad W = [r]T \tag{1}$$

Here, the following equation holds.

$$\begin{aligned} & e(G_1, G_2) \\ = \ & e(B, Y) \cdot e(H, G_2)^{x_U} \cdot e(B, G_2)^{y_U} \cdot e(K, G_2)^{z_U - q\, y_U} \cdot e(K, Y)^{-q} \end{aligned} \tag{2}$$

The data generated hereafter is a Fiat-Shamir transformation of a zero-knowledge proof of knowledge of x_U, y_U, z_U, and q, r that satisfies Eqs. (1) and (2). Since B is a perfect hiding commitment of A_U, the only knowledge that the receiver of the signature can obtain is (U, V, W) which is an ElGamal type double encryption of $[x_U]G$

(i) Sign randomly chooses $(t, u, v, f, o) \in_R (\mathbb{Z}/p\mathbb{Z})^5$ and generates

$$\begin{aligned} X' &= e(H, G_2)^t \cdot e(B, G_2)^u \cdot e(K, G_2)^v \cdot e(K, Y)^f \\ U' &= [t + o]G \ , \quad V' = [o]S \ , \quad W' = [o]T \end{aligned}$$

(ii) Sign generates

$$c = \mathcal{H}(p, G_1, G_2, G_T, G, \psi, Y, S, T, H, K, B, U, V, W, X', V', W', U', m)$$

(iii) Sign generates

$$\begin{aligned} x' &= cx_U + t \ , \quad y' = cy_U + u \ , \quad z' = c(z_U - qy_U) + v \\ q' &= -cq + f \ , \quad r' = cr + o \end{aligned}$$

3. Sign outputs

$$gs := (B, U, V, W, c, x', y', z', q', r')$$

as a signature on message m.

Verify

1. Verify is given mpk, tpk, m, and gs.
2. Verify generates

$$X' = e(H, G_2)^{x'} e(B, G_2)^{y'} e(K, G_2)^{z'} e(K, Y)^{q'} \left(\frac{e(G_1, G_2)}{e(B, Y)} \right)^{-c}$$
$$U' = [x' + r']G - [c]U \ , \quad V' = [r']S - [c]V \ , \quad W' = [r']T - [c]W.$$

3. Verify outputs acc if equation

$$c = \mathcal{H}(p, G_1, G_2, G_T, G, \psi, Y, S, T, H, K, B, U, V, W, X', V', W', U', m)$$

holds. Otherwise, it outputs rej.

Open

1. Open is given mpk, tpk, tsk, m, gs, and $\mathcal{L}$.
2. If $\mathsf{Verify}(mpk, tpk, m, gs) = rej$, it outputs $\perp$ and stops.
3. Open generates and outputs

$$Q = U - [1/s]V \qquad (= U - [1/t]W)$$

Then, Open generates and outputs a non-interactive proof of knowledge of either s or t that satisfies either of the above equations and Q as a $proof$.
4. Open searches Q_U that coincides with the Q in $\mathcal{L}$. If there is such a Q_U, it outputs the corresponding U. Otherwise, it outputs $\perp'$.

4 Security

Theorem 1. *The proposed scheme has Traceability property if the SDH assumption holds.*

Theorem 2. *The proposed scheme has Anonymity property if the DDH assumption holds.*

Theorem 3. *The proposed scheme has Non-Frameability property if we assume the discrete logarithm problem is difficult to solve.*

Proofs of the theorems are given in the full paper [15].

5 Comparison with Previous Schemes

We compare the signature length and computational complexity of the proposed scheme to those of the previous schemes [27,10] and those of a variant of the scheme in [7]. This variant protocol is given in the full paper [15].

The variant scheme of [7] differs from the original one in two points. The first point is that it provides a joining protocol, whose construction is already presented in Section 7 of [6]. The second point is that it uses a double encryption scheme [28] variant of the linear encryption scheme instead of the simple linear encryption scheme used in the original scheme. Since the *Open* oracle in group signature plays a role similar to that of the role of the decryption oracle in the IND-CCA2 game of public key cryptosystems, the encryption scheme used in group signature needs to be IND-CCA2 secure. However, the signed ElGamal encryption is IND-CCA2 secure only in the generic model [31], in the same way that the linear encryption scheme adopted in [7] is. Hence, the use of a double encryption variant is a legitimate solution to avoid dependence on the generic group model.

Although the above variant scheme is less efficient than the original scheme, comparing our scheme with this variant scheme is appropriate. This is because our scheme and the schemes in [27] and [10] all provide a Join protocol and their security is proved in a non-generic group model.

We compare the group signature lengths of our scheme and those of the previous schemes. We assume that $\mathcal{G}_1 \neq \mathcal{G}_2$ such that the representation of G_1 can be a 172 bit string when $|p| = 171$ by using the elliptic curve defined by [26]. The choice of such a curve makes it possible to express B by a short string. When such a curve is not available, the signature length of our scheme is much shorter than those of the other previous schemes. We also assume that the representations of G_T and G are 1020 bits and 172 bits. A group signature of the variant of the scheme in [7] is composed of seven $\mathbb{Z}/p\mathbb{Z}$ and five $\mathcal{G}_1$ elements. That of the scheme in [27] is composed of ten $\mathbb{Z}/p\mathbb{Z}$, six $\mathcal{G}_1$, and two $\mathcal{G}_T$ elements, and that of the scheme in [10] is composed of four $\mathbb{Z}/p\mathbb{Z}$, three $\mathcal{G}_1$, and four $\mathcal{G}_T$ elements. In contrast, that of the proposed scheme is composed of six $\mathbb{Z}/p\mathbb{Z}$, one $\mathcal{G}_1$, and three $\mathcal{G}$ elements, and thus its signature length is the shortest among the other previous schemes.

We also estimate the computational cost of our scheme and that of the previous schemes by the number of scalar multiplications/modular exponentiations in $\mathcal{G}, \mathcal{G}_1, \mathcal{G}_2$, and $\mathcal{G}_T$ and the number of pairing operations e required for Sign and Verify, since these are the most costly computations. Although we cannot present a precise estimation of the computational cost of each operation since it depends on the choice of the groups $\mathcal{G}, \mathcal{G}_1, \mathcal{G}_2$, and $\mathcal{G}_T$, these computations can be done quite efficiently if we choose Tate pairing for e and adopt the computation tools described in [23].

We also list the assumptions required in our scheme and the previous schemes [27,10], and the variant of the scheme in [7]. From Theorems 1, 2, and 3, our scheme requires the SDH assumption, the Decision Diffie-Hellman assumption, and the existence of random oracles. The scheme in [7] requires the SDH as-

sumption, the DLDH assumption, and the existence of random oracles. That in [27] requires the SDH assumption, the Decision Bilinear Diffie-Hellman (DBDH) assumption, and the existence of random oracles. That in [10] requires the Lysyanskaya-Rivest-Sahai-Wolf (LRSW) assumption, the Decision Diffie-Hellman assumption, and the existence of random oracles. The DLDH assumption is proposed in [7] which is proved to hold in generic bilinear groups. The LRSW assumption is proposed in [22] and is proved to hold in generic groups. The LRSW assumption is also proved to hold in generic bilinear groups in [10]. The SDH assumption and the LRSW assumption cannot be compares to each other.

These results of estimation and required assumptions are given in Table 1, where "# of SMul" , "# of MExp", "# of pairings", and "Sig. Len." are abbreviations of "the number of scalar multiplications" , "the number of modular exponentiations", "the number of pairings", and "signature length". Installing the revocation mechanism proposed in [7] has no effect on this estimation[6].

	A variant of [7] Sign/Verify	Scheme in [27] Sign/Verify	Scheme in [10] Sign/Verify	Our Scheme Sign/Verify
# of SMul in $\mathcal{G}$	-	-	-	7/6
# of SMul in $\mathcal{G}_2$	13/12	20/13	3/0	-
# of MExp in $\mathcal{G}_T$	4/5	6/2	14/16	4/5
# of pairings	0/2	0/3	0/3	0/2
Sig. Len. (bits)	2057	4782	5296	1711
Assumptions	SDH,DLDH	SDH,DBDH	LRSW,DDH	SDH,DDH

Table 1. Complexity & Assumptions

References

1. G. Ateniese, J. Camenisch, M. Joye, G. Tsudik: A Practical and Provable Secure Coalition-Resistant Group Signature Scheme. CRYPTO 2000, LNCS 1880, pp.255–270.
2. G. Ateniese, B. de Medeiros: Efficient Group Signatures without Trapdoors. ASIACRYPT 2003, LNCS 2894, pp.246–268.
3. Niko Bari, Birgit Pfitzmann: Collision-Free Accumulators and Fail-Stop Signature Schemes without Trees. EUROCRYPT 1997: 480-494.
4. Mihir Bellare, Haixia Shi, Chong Zhang: Foundations of Group Signatures: The Case of Dynamic Groups. CT-RSA 2005: 136-153
5. M. Bellare, D. Micciancio, B. Warinschi: Foundations of Group Signatures: Formal Definitions, Simplified Requirements, and a Construction Based on General Assumptions. EUROCRYPT 2003, LNCS 2656, pp. 614-629.

[6] Given $(G_1, G_2, A_U, y_U, x_U, z_U, y_{\bar{U}}, \bar{H}, \bar{K}, Y)$ such that $Y = [w]G_2, [w + y_U]A_U + [x_U]H + [z_U]K = G_1, [w + y_{\bar{U}}]\bar{G}_1 = G_1, [w + y_{\bar{U}}]\bar{H} = H, [w + y_{\bar{U}}]\bar{K} = K$ for some $w \in \mathbb{Z}/q\mathbb{Z}$, $\bar{A}_U$ that satisfies $[w + y_U]\bar{A}_U + [x_U]\bar{H} + [z_U]\bar{K} = \bar{G}_1$ can be computed as $\bar{A}_U = [1/(y_{\bar{U}} - y_U)](A_U - \bar{G}_1 - [x_U]\bar{H} - [z_U]\bar{K})$.

6. Dan Boneh, Xavier Boyen: Short Signatures Without Random Oracles. EUROCRYPT 2004: 56-73.
7. Dan Boneh, Xavier Boyen, Hovav Shacham: Short Group Signature. CRYPTO 2004, Lecture Notes in Computer Science 3152, pp. 41-55, 2004, Springer.
8. Jan Camenisch, Jens Groth: Group Signatures: Better Efficiency and New Theoretical Aspects. Security in Communication Networks - SCN 2004, LNCS series.
9. Jan Camenisch, Anna Lysyanskaya: A Signature Scheme with Efficient Protocols. SCN 2002: 268-289.
10. Jan Camenisch, Anna Lysyanskaya: Signature Schemes and Anonymous Credentials from Bilinear Maps. Crypto 2004, Springer Verlag, 2004.
11. J. Camenisch, M. Michels: A group signature scheme based on an RSA-variant. Technical Report RS-98-27, BRICS, University of Aarhus, November 1998. An earlier version appears in ASIACRYPT '98.
12. J. Camenisch, M. Stadler: Efficient Group Signature Schemes for Large Groups. CRYPTO '97, LNCS 1296, pp. 410–424.
13. D. Chaum, E. van Heyst: Group Signatures. EUROCRYPT '91, LNCS 547, pp. 257–265.
14. Cynthia Dwork, Moni Naor, Amit Sahai: Concurrent Zero-Knowledge. STOC 1998: 409-418.
15. Jun Furukawa, Hideki Imai: Efficient Group Signature Scheme from Bilinear Maps (with appendixes). Available from the first author via e-mail.
16. Jun Furukawa, Shoko Yonezawa: Group Signatures with Separate and Distributed Authorities. Fourth Conference on Security in Communication Networks '04 (SCN04), 2004.
17. Tetsuya Izu, Tsuyoshi Takagi: Efficient Computations of the Tate Pairingfor the Large MOV Degrees. ICISC 2002: 283-297.
18. Aggelos Kiayias, Moti Yung: Group Signatures: Provable Security, Efficient Constructions and Anonymity from Trapdoor-Holders. Cryptology ePrint Archieve, Report 2004/076.
19. A. Kiayias, Y. Tsiounis, M. Yung: Traceable Signatures. EUROCRYPT 2004, LNCS 3027, pp. 571–589.
20. Joe Kilian, Erez Petrank: Identity Escrow. CRYPTO 1998: 169-185.
21. Seungjoo Kim, Sung Jun Park, Dongho Won: Convertible Group Signatures. ASIACRYPT 1996: 311-321.
22. Anna Lysyanskaya, Ronald L. Rivest, Amit Sahai, Stefan Wolf: Pseudonym Systems. Selected Areas in Cryptography 1999: 184-199.
23. A. Menezes, C. van Oorschot, S. Vanstone: *Handbook of Applied Cryptography*, CRC Press, pp. 617-627, (1997).
24. Daniele Micciancio, Erez Petrank: Efficient and Concurrent Zero-Knowledge from any public coin HVZK protocol. Electronic Colloquium on Computational Complexity (ECCC)(045):(2002).
25. S. Mitsunari, R. Sakai, M. Kasahara: A new Traitor tracing. IEICE Trans. Fundamentals, E85-A(2), pp.481-484, Feb. 2002.
26. Atsuko Miyaji, Masaki Nakabayashi, Shunzou Takano: New explicit conditions of elliptic curve traces for FR-reduction. IEICE Trans. E85-A(2), pp. 481-484, 2002.
27. Lan Nguyen, Rei Safavi-Naini: Efficient and Provably Secure Trapdoor-free Group Signature Schemes from Bilinear Pairings. Asiacrypt 2004. pp. 372-386.
28. Moni Naor, Moti Yung: Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks. STOC 1990: 427-437.
29. D. Pointcheval, J. Stern: Security Arguments for Digital Signatures and Blind Signatures. J. Cryptology 13(3): 361-396 (2000).

30. Michael Scott, Paulo S.L.M Barreto: Generating more MNT elliptic curves. Cryptology ePrint Archieve, Report 2004/058.
31. Claus-Peter Schnorr, Markus Jakobsson: Security of Signed ElGamal Encryption. ASIACRYPT 2000: 73-89.

Group Signature Where Group Manager, Members and Open Authority Are Identity-Based

Victor K. Wei[1], Tsz Hon Yuen[1], and Fangguo Zhang[2]

[1] Department of Information Engineering,
The Chinese University of Hong Kong,
Shatin, Hong Kong
{kwwei, thyuen4}@ie.cuhk.edu.hk
[2] Department of Electronics and Communication Engineering,
Sun Yat-sen University,
Guangzhou, 510275, P.R.China
isdzhfg@zsu.edu.cn

Abstract. We present the first group signature scheme with provable security and signature size $O(\lambda)$ bits where the group manager, the group members, and the Open Authority (OA) are all identity-based. We use the security model of Bellare, Shi, and Zhang [3], except to add three identity managers for manager, members, and OA respectively, and we discard the Open Oracle ($\mathcal{OO}$). Our construction uses identity-based signatures summarized in Bellare, Namprempre, and Neven [2] for manager, Boneh and Franklin's IBE [7] for OA, and we extend Bellare et al.[3]'s group signature construction by verifiably encrypt an image of the member public key, instead of the public key itself. The last innovation is crucial in our efficiency; otherwise, Camenisch and Damgard[9]'s verifiable encryption would have to be used resulting in lower efficiency.

1 Introduction

Identity based cryptography, introduced by Shamir [25], allows the users' public key to be their identity. Usually a trusted third party computes the private key from the identity (any arbitrary string such as email address). Comparing with certificate from certificate authority (CA), the identity based public key can identify the user immediately. Besides, the problem of distribution of public keys is avoided in identity based cryptography.

Group signature, introduced by Chaum and van Heyst [13], allows any member of a group to sign on behalf of the group. However, the identity of the signer is kept secret. Anyone can verify that the signature is signed by a group member, but cannot tell which one. Therefore group signature provides anonymity for signers. Usually in group signature schemes, a group manager issues certificates to his group members. Then the group member uses his certificate and his own secret key to sign messages. Anyone can verify the signature by the group manager's public key. In some cases, an open authority has a secret key to revoke

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 468–480, 2005.

the anonymity of any signature in case of dispute. Mostly it can be done by an encryption to the open authority when signing the message. On the other hand, anonymity can be revoked when a signer double signs in some schemes. Group signature is a very useful tool in real world. It can be used in e-cash, e-voting or attestation [8] in trusted computing group.

Weil and Tate pairing has been widely used in identity based cryptography in recent years. Pairing is also used to construct short group signature [6] recently. However, none of the existing group signature scheme can be completely verified in an identity based manner, that is the group public key and the opener public key are arbitrary strings. The current "Identity based" group signature are mostly for identity based group member only ([22][19][26][11][14]). We think that identity based group member is not enough for group signature. It is because the signer's public key is always anonymous in group signature. Whether it is identity based or not has no effect to the verifier. We think that it is constructive to have a group signature with identity based group public key, which is the identity of the group manager in this case. At the same time, we also want to support identity based group members, as well as open authority. We call this new scheme to be a fully identity based group signature. In this paper, we will give a generic construction, and then a specific instantiation of such a identity based group signature.

Contributions. Our main contributions are:

- We introduce the formal study of group signature schemes with identity based group manager, identity based group members and identity based open authority.
- We present the first construction of the above scheme, complete with security models, and reductionist security proofs in the random oracle model. The size of the signature is $O(\lambda)$ bits.
- We extend Bellare, Shi, and Zhang [3]'s generic group signature construction by verifiably encrypt, to the Open Authority (OA), a one-way image of the signer public key instead of the signer public key itself. This technique is crucial to the topic in this paper.

The rest of the paper is **organized** as follows: Section 2 contains preliminaries. Section 3 contains the security model. Section 4 contains the constructions. Section 5, security theorems. Section 6, discussions and applications.

2 Preliminaries

Related results:
After the introduction of group signature by Chaum and van Heyst [13], there are numerous group signature schemes proposed, such as Ateniese et al [1], Dodis et al [15], Boneh et al [6]. The state-of-the-art is to have a group signature scheme with signature size independent of the group size. The security model of dynamic group signature is proposed in [3].

Identity based signature is suggested in 1984 by Shamir [25], but practical identity based encryption is not found until 2001 by Boneh and Franklin [7] using Weil pairing. Identity based group signature is firstly proposed by Park et al [22]. [19] showed that the anonymity of the scheme was not guaranteed. Tseng and Jan [26] presented a novel ID-based group scheme. However, it is universally forgeable [18] and not coalition-resistant [17]. Several identity based group signature schemes are proposed in [11], [14]. [11] requires a new pair of certificate for each signature. However all of them only have identity based key pairs for group members only. Group signature scheme with identity based group manager and identity based open authority remains as an open problem.

Pairings:
Following the notation of pairings in [7], let $\mathbb{G}_1, \mathbb{G}_2$ be (mutiplicative) cyclic groups of prime order p. Let g_1 be a generator of $\mathbb{G}_1$ and g_2 be a generator of $\mathbb{G}_2$. Let ψ is a computable isomorphism from $\mathbb{G}_1$ to $\mathbb{G}_2$, with $\psi(g_2) = g_1$.

Definition 1. *A map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is called a bilinear pairing if, for all $x \in \mathbb{G}_1, y \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p$, we have $\hat{e}(x^a, y^b) = \hat{e}(x, y)^{ab}$, and $\hat{e}(g_1, g_2) \neq 1$.*

Definition 2. *(co-CDH problem) The co-computational Diffie-Hellman problem in $(\mathbb{G}_1, \mathbb{G}_2)$ is as follows: given $P, P^\alpha \in \mathbb{G}_1$, $Q \in \mathbb{G}_2$, for unknown $\alpha \in \mathbb{Z}_p$, to compute Q^α.*

Definition 3. *(DDH problem) The decisional Diffie-Hellman problem in $\mathbb{G}_1$ is as follows: given $P, P^\alpha, P^\beta, R \in \mathbb{G}_1$ for unknown $\alpha, \beta \in \mathbb{Z}_p$, to decide if $R = P^{\alpha\beta}$.*

Definition 4. *(co-DBDH problem) The co-decisional Bilinear Diffie-Hellman problem in $(\mathbb{G}_1, \mathbb{G}_2)$ is as follows: given $P, P^\alpha, P^\beta \in \mathbb{G}_1$, $Q \in \mathbb{G}_2$, $R \in \mathbb{G}_T$ for unknown $\alpha, \beta \in \mathbb{Z}_p$, to decide if $R = \hat{e}(P, Q)^{\alpha\beta}$.*

Definition 5. *(k-SDH' problem) The k-Strong Diffie-Hellman' problem in $(\mathbb{G}_1, \mathbb{G}_2)$ is as follows: given $g_1, {g_1}^\gamma, ..., {g_1}^{\gamma^k} \in \mathbb{G}_1$ and $g_2, {g_2}^\gamma \in \mathbb{G}_2$ as input, outputs a pair $({g_1}^{1/\gamma+x}, x)$ where $x \in \mathbb{Z}_p^*$.*

Definition 6. *(k-CAA2 problem) The k-CAA2 problem in $(\mathbb{G}_1, \mathbb{G}_2)$ is as follows: given $v, u \in \mathbb{G}_1$, $g_2, {g_2}^\gamma \in \mathbb{G}_2$ and pairs (A_i, e_i, λ_i) with distinct and nonzero e_i's satisfying $A_i^{\gamma+e_i} v^{\lambda_i} = u$ for $1 \leq i \leq k$ as input, outputs a pair $(A_{k+1}, e_{k+1}, \lambda_{k+1})$ satisfying $A_{k+1}^{\gamma+e_{k+1}} v^{\lambda_{k+1}} = u$, with $e_{k+1} \neq e_i$ for all $1 \leq i \leq k$.*

The above k-SDH' problem and k-CAA2 problem are proven equivalent in [27] assume the value $log_u(v)$ is known. [27] also shows that the k-Strong Diffie-Hellman assumption in [20],[5],[28] is at least as strong as the k-SDH' problem.

Definition 7. *Let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a pairing. Given the following:*

1. $g_1, g_1^{\alpha}, g_1^{\beta_i}, g_1^{\gamma_i} \in \mathbb{G}_1$ *for* $1 \leq i \leq k$;
2. $g_2, g_2^{\delta_1}, g_2^{\delta_2} \in \mathbb{G}_2$, $R \in G_T$;
3. $\Pr\{\gamma_i = \alpha\beta_i, \text{ all } i, 1 \leq i \leq k\} = \Pr\{\gamma_i \neq \alpha\beta_i, \text{ all } i, 1 \leq i \leq k\} = 1/2$.
4. $\Pr\{\gamma_i = \alpha\beta_i, \text{ all } i, 1 \leq i \leq k \text{ AND } R = \hat{e}(g_1, g_2)^{\delta_1\delta_2}\} = \Pr\{\gamma_i \neq \alpha\beta_i, \text{ all } i, 1 \leq i \leq k \text{ AND } R \neq \hat{e}(g_1, g_2)^{\delta_1\delta_2}\} = 1/2$

The Lockstep DDH Problem *(resp.* Lockstep DDH+coDBDH Problem*) is to distinguish between the two nonzero probability events in (3) (resp. (4)) above with non-negligible probability over 1/2. The* Lockstep DDH Assumption *(resp.* Lockstep DDH+coDBDH Assumption*) is that no PPT algorithm can solve the Lockstep DDH Problem (resp. Lockstep DDH+coDBDH Problem).*

Lemma 1 *The Lockstep DDH Assumption in $\mathbb{G}_1$ holds if and only if the DDH Assumption in $\mathbb{G}_1$ holds. The Lockstep DDH+coDBDH Assumption holds in $(\mathbb{G}_1, \mathbb{G}_2)$ if and only if the DDH Assumption in $\mathbb{G}_1$ and the co-DBDH assumption in $(\mathbb{G}_2, \mathbb{G}_1)$ both hold.*

The proof of this lemma can be found in the full version of this paper.

3 Security Model

We present a security model for the identity based group signature. Here we adapt the models for dynamic group signature in [3], and add support for IBGS. Our scheme is applicable to multiple certificate authorities (CA, or group managers) and open authorities (OA).

Syntax: A *identity-based group signature (IBGS)* is a tuple (Init, OKg, GKg, UKg, Join, Iss, GSig, GVf, Open, Judge) where:

- Init: $1^{\lambda} \mapsto \mathsf{param}$. On input the security parameter 1^{λ}, generates system-wide public parameters param. The identity manager of CA (IM_A) has (sk,pk) pair (x_A, y_A) for CA (resp. IM_U has (x_U, y_U) for group members, IM_O has (x_O, y_O) for OA) and an efficiently samplable one-way NP-relation $\langle\mathcal{R}_A\rangle$, with trapdoor x_A (resp. $\langle\mathcal{R}_U\rangle$, with trapdoor x_U, $\langle\mathcal{R}_O\rangle$, with trapdoor x_O). An efficiently samplable family of one-way NP-relation $\mathcal{F} = \{\langle\mathcal{R}_{C,i}\rangle : i\}$ with trapdoor gsk_i, is defined for issuing certificate. param is (y_A, y_U, y_O, $\mathcal{R}_A$, $\mathcal{R}_U$, $\mathcal{R}_O$, $\mathcal{F}$).
- OKg:$(\mathsf{oa}, x_O) \mapsto (x_{\mathsf{oa}}, aux_{\mathsf{oa}})$. On input the OA identity oa, the IM_O uses his secret key x_O to compute the secret key x_{oa} of the OA, some auxiliary information aux_{oa} such that $((x_{\mathsf{oa}}, aux_{\mathsf{oa}}), \mathsf{oa}) \in \mathcal{R}_O$.
- GKg: $(\mathsf{ca}, x_A) \mapsto (x_{\mathsf{ca}}, aux_{\mathsf{ca}}, \langle\mathcal{R}_{C,\mathsf{ca}}\rangle)$. On input the CA identity ca, the IM_A samples $\mathcal{F}$ to get the relation $\langle\mathcal{R}_{C,\mathsf{ca}}\rangle$. The IM_A uses his secret key x_A to compute the group secret key of CA x_{ca}, some auxiliary information aux_{ca} such that $((x_{\mathsf{ca}}, aux_{\mathsf{ca}}), \mathsf{ca}) \in \mathcal{R}_A$.

- UKg: $(\mathsf{id}, x_U) \mapsto (x_{\mathsf{id}}, aux_{\mathsf{id}})$. On input the identity id of the member, the IM_U uses his secret key x_U to compute the secret key x_{id} of the member, some auxiliary information aux_{id} such that $((x_{\mathsf{id}}, aux_{\mathsf{id}}), \mathsf{id}) \in \mathcal{R}_U$.
- Join,Iss is a pair of interactive protocols between the user and the CA, with common inputs ca and id. Iss's additional inputs are x_{ca} and aux_{ca}. Join's additional inputs are x_{id} and aux_{id}. At the conclusion, Join obtains $cert_{\mathsf{id}}$ satisfying $((x_{\mathsf{id}}, aux_{\mathsf{id}}, cert_{\mathsf{id,ca}}), \mathsf{id}) \in \mathcal{R}_{C,\mathsf{ca}}$, and Iss stores $(\mathsf{id}, cert_{\mathsf{id,ca}})$ in a registration table reg.
- GSig: $(\mathsf{id}, x_{\mathsf{id}}, aux_{\mathsf{id}}, \mathsf{ca}, \mathsf{oa}, cert_{\mathsf{id,ca}}, M) \mapsto \sigma$. On input the keys, certificates and message, outputs a signature σ.
- GVf: $(\mathsf{ca}, \mathsf{oa}, M, \sigma) \mapsto 0$ or 1. On input the message and signature, outputs 1 for valid signature and 0 for invalid signature.
- Open: $(\mathsf{ca}, x_{\mathsf{oa}}, \mathsf{reg}, M, \sigma) \mapsto (i, \omega)$. The OA with key x_{oa} has read access to reg. On input a valid signature σ for message M for ca, output identity i for the corresponding signer, and ω is the proof of this claim. Output $i = \perp$ if no such member is found.
- Judge: $(\mathsf{ca}, \mathsf{id}, \mathsf{oa}, M, \sigma, \omega) \mapsto 0$ or 1. It checks if the proof ω is a valid proof that id is the real signer of σ for message M under ca, oa. Outputs 1 for valid and 0 for invalid.

Remarks: Here we use $(\mathsf{param}, \mathsf{ca})$ to denote *gpk* in [3]'s original syntax. We also split the GKg in [3] into Init, OKg and GKg. It is because we want to emphasize that group managers (CA) and open authorities (OA) are identity based.

Security notions: We have the security notions of Correctness, Anonymity, Traceability, Non-frameability from [3], with modification for identity based. We give a brief description here.

Correctness: Let $\sigma \leftarrow \mathsf{GSig}(\mathsf{id}, x_{\mathsf{id}}, aux_{\mathsf{id}}, \mathsf{ca}, \mathsf{oa}, cert_{\mathsf{id,ca}}, M)$ for arbitrary id, x_{id}, aux_{id}, ca, oa, $cert_{\mathsf{id,ca}}$, M. The IBGS has opening correctness if $(\mathsf{id}, \omega) \leftarrow \mathsf{Open}(\mathsf{ca}, x_{\mathsf{oa}}, \mathsf{reg}, M, \sigma)$ and $\mathsf{Judge}(\mathsf{ca}, \mathsf{id}, \mathsf{oa}, M, \sigma, \omega) = 1$ with overwhelming probability. It has verification correctness if $\mathsf{GVf}(\mathsf{ca}, \mathsf{oa}, M, \sigma) = 1$ with probability 1. The IBGS is correct if it has verification and opening correctness.

We have the following oracles for the adversary to query:

- The *Random Oracle* $\mathcal{RO}$: simulate the random oracle normally.
- The *Key Extraction Oracle-CA* $\mathcal{KEO}_c$: $\mathsf{ca} \rightarrow x_{\mathsf{ca}}$. Upon input CA ca, outputs his secret key x_{ca}.
- The *Key Extraction Oracle-OA* $\mathcal{KEO}_o$: $\mathsf{oa} \rightarrow x_{\mathsf{oa}}$. Upon input OA oa, outputs his secret key x_{oa}.
- The *Key Extraction Oracle-User* $\mathcal{KEO}_u$: $\mathsf{id} \rightarrow x_{\mathsf{id}}$. Upon input user id, outputs his secret key x_{id}.
- The *Join Oracle* $\mathcal{JO}$: $(\mathsf{id}, \mathsf{ca}) \rightarrow cert_{\mathsf{ca}}$. Upon input id of group ca, outputs the $cert_{\mathsf{ca}}$ corresponding to an honest Iss-executing CA.
- The *Issue Oracle* $\mathcal{IO}$: $(\mathsf{id}, \mathsf{ca}) \rightarrow cert_{\mathsf{ca}}$. Upon input id of group ca, outputs the $cert_{\mathsf{ca}}$ corresponding to an honest Join-executing user.
- The *Corruption Oracle* $\mathcal{CO}$: $(\mathsf{id}, \mathsf{ca}) \rightarrow (x_{\mathsf{id}}, aux_{\mathsf{id}}, cert_{\mathsf{ca}})$. Upon input user id of group ca, outputs the secret keys $(x_{\mathsf{id}}, aux_{\mathsf{id}}, cert_{\mathsf{ca}})$.

- The *Signing Oracle* $\mathcal{SO}$: $(\mathsf{id}, \mathsf{ca}, \mathsf{oa}, M) \rightarrow \sigma$. Upon input user id, group ca, oa and a message M, outputs a valid signature.
- The *Open Oracle* $\mathcal{OO}$: $(\mathsf{oa}, \mathsf{ca}, M, \sigma) \rightarrow (\mathsf{id}, \omega)$. Upon input a valid signature σ for message M under ca, oa, outputs the signer id and the proof ω.
  Remark: $\mathcal{KEO}_O$ is a stronger oracle than $\mathcal{OO}$ in the sense that $\mathcal{KEO}_O$ directly gives the secret key for OA, while $\mathcal{OO}$ only opens a particular signature.

Anonymity: We have the following **Experiment Anon** for anonymity:

1. Simulator $\mathcal{S}$ invokes Init. $\mathcal{S}$ invokes UKg, Join, Iss together q_u times to generate a set of honest users, denoted HU, with secret keys and certificates.
2. $\mathcal{A}$ queries $\mathcal{RO}, \mathcal{CO}, \mathcal{OO}, \mathcal{IO}, \mathcal{KEO}_c, \mathcal{KEO}_u, \mathcal{KEO}_o$ in arbitrary interleaf.
3. $\mathcal{A}$ selects two users $\mathsf{id}_0, \mathsf{id}_1 \in$ HU, $\mathsf{ca}_g, \mathsf{oa}_g$ a message M and gives them to $\mathcal{S}$. Then $\mathcal{S}$ randomly chooses $b \in \{0, 1\}$ and returns the gauntlet ciphertext $\sigma \leftarrow \mathcal{SO}(\mathsf{id}_b, \mathsf{ca}_g, \mathsf{oa}_g, M)$. oa_g should not be input to $\mathcal{OO}, \mathcal{KEO}_o$ before.
4. $\mathcal{A}$ queries $\mathcal{RO}, \mathcal{CO}, \mathcal{OO}, \mathcal{IO}, \mathcal{KEO}_c, \mathcal{KEO}_u, \mathcal{KEO}_o$ in arbitrary interleaf. oa_g should not be input to $\mathcal{OO}, \mathcal{KEO}_o$.
5. $\mathcal{A}$ delivers an estimate $\hat{b} \in \{0, 1\}$ of b.

$\mathcal{A}$ also has write access to registration table reg in the experiment. $\mathcal{A}$ wins the Experiment Anon if $\hat{b} = b$, and oa_g has never been queried to $\mathcal{KEO}_o$. $\mathcal{A}$'s advantage is its probability of winning Experiment Anon minus half.
Remark: By not allowing to query the gauntlet oa_g, our model is closer to that of [6] which does not support any $\mathcal{OO}$, than to that of [3] which supports $\mathcal{OO}$.

Definition 8. *The IBGS is anonymous if no PPT adversary has a non-negligible advantage in Experiment Anon.*

Traceability: We have the following **Experiment Trace** for traceability:

1. $\mathcal{S}$ invokes Init. $\mathcal{S}$ invokes UKg, Join, Iss together q_u times to generate a set of honest users, denoted HU, with secret keys and certificates.
2. $\mathcal{A}$ queries $\mathcal{RO}, \mathcal{CO}, \mathcal{JO}, \mathcal{KEO}_c, \mathcal{KEO}_u, \mathcal{KEO}_o$ in arbitrary interleaf.
3. $\mathcal{A}$ delivers signature σ for messages M for group ca and open authority oa. ca should not be input to $\mathcal{KEO}_c$.

$\mathcal{A}$ also has read access to reg. $\mathcal{A}$ wins the Experiment Trace if $\mathsf{GVf}(\mathsf{ca}, \mathsf{oa}, M, \sigma) = 1$, either $i = \perp$ or $\mathsf{Judge}(\mathsf{ca}, i, \mathsf{oa}, m, \sigma, \omega) = 0$, where $(i, \omega) \leftarrow \mathsf{Open}(\mathsf{ca}, x_{\mathsf{oa}}, \mathsf{reg}, M, \sigma)$, ca has never been queried to $\mathcal{KEO}_c$, and (i, ca) has never been queried to $\mathcal{CO}$, $\mathcal{A}$'s advantage is its probability of winning.

Definition 9. *The IBGS is traceable if no PPT adversary has a non-negligible advantage in Experiment Trace.*

Non-Frameability: We have the following **Experiment NF** for non-frameability:

1. $\mathcal{S}$ invokes Init. $\mathcal{S}$ invokes UKg, Join, Iss together q_u times to generate a set of honest users, denoted HU, with secret keys and certificates.

2. $\mathcal{A}$ queries $\mathcal{RO}, \mathcal{CO}, \mathcal{SO}, \mathcal{IO}, \mathcal{KEO}_c, \mathcal{KEO}_u, \mathcal{KEO}_o$ in arbitrary interleaf.
3. $\mathcal{A}$ delivers (σ, M, i, ω), where ω is the proof of user i signed the signature σ for messages M with group ca and open authority oa.

$\mathcal{A}$ also has write access to reg. $\mathcal{A}$ wins the Experiment NF if $\mathsf{GVf}(\mathsf{ca}, \mathsf{oa}, M, \sigma) = 1$, $\mathsf{Judge}(\mathsf{ca}, i, \mathsf{oa}, M, \sigma, \omega) = 1$, i has never been queried to $\mathcal{CO}$ and σ is not the output from $\mathcal{SO}$ for $M, i, \mathsf{ca}, \mathsf{oa}$. $\mathcal{A}$'s advantage is its probability of winning.

Definition 10. *The IBGS is non-frameable if no PPT adversary has a non-negligible advantage in Experiment NF.*

Definition 11. *An IBGS scheme is secure if it is correct, anonymous, traceable and non-frameable.*

4 Constructions

In this paper, we present a generic construction for *identity-based group signature (IBGS)* which is applicable to different kinds of relations between the identity based CA, users and open authority. After the generic construction, we give an efficient implementation which is provably secure in the random oracle model.

4.1 Generic Construction

A generic *IBGS* is a tuple (Init, OKg, GKg, UKg, Join, Iss, GSig, GVf, Open, Judge):

- Init, GKg, OKg, UKg, Open, Judge follows the syntax.
- Join,Iss is a pair of interactive protocols with common inputs ca and id. Iss's additional inputs are x_{ca} and aux_{ca}. Join's additional inputs are x_{id} and aux_{id}. Join runs a proof of knowledge protocol to proof that he knows x_{id} and aux_{id} to Iss. At the conclusion, Join obtains $cert_{\mathsf{id}}$ satisfying $((x_{\mathsf{id}}, aux_{\mathsf{id}}, cert_{\mathsf{id,ca}}), \mathsf{id}) \in \mathcal{R}_{C,\mathsf{ca}}$, and Iss stores $(\mathsf{id}, cert_{\mathsf{id,ca}})$ in a registration table reg. Join may also obtain aux_{ca} as part of $cert_{\mathsf{id,ca}}$.
- GSig: $(\mathsf{id}, \mathsf{ca}, \mathsf{oa}, x_{\mathsf{id}}, aux_{\mathsf{id}}, cert_{\mathsf{id}}, M) \mapsto \sigma$. A user id who has $cert_{\mathsf{id}}$ runs:

$$SPK\{(\mathsf{id}, x_{\mathsf{id}}, aux_{\mathsf{id}}, cert_{\mathsf{id,ca}}, r) : (x_{\mathsf{id}}, aux_{\mathsf{id}}, \mathsf{id}) \in \mathcal{R}_U$$
$$\wedge\ (\mathsf{id}, aux_{\mathsf{id}}, cert_{\mathsf{id,ca}}) \in \mathcal{R}_{C,\mathsf{ca}}\ \wedge\ \mathsf{ctxt} = Enc(\mathsf{id}, \mathsf{oa}, r)\}(M)$$

  The signature σ is obtained from the above SPK, following [10]'s notion.
- GVf: $(\sigma, M) \mapsto 0$ or 1. On input the signature σ, a verifier verifies σ according to the above SPK. The verifier outputs 1 for valid signature and 0 otherwise.

4.2 An Instantiation: IBGS-SDH

We instantiate the generic construction above in the SDH group.
Init: On input the security parameter 1^λ, generates a pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ where the above three (mutiplicative) cyclic groups are of order p. The IM_A (resp. IM_O, IM_U) secret key is $x_A \in \mathbb{Z}_p^*$ (resp. x_O, x_U) and public keys are $g_A, y_A = g_A^{x_A} \in \mathbb{G}_2$ (resp. $g_O, y_O = g_O^{x_O}$, and $g_U, y_U = g_U^{x_U}$). Let u be a generator in $\mathbb{G}_1$. Define cryptographic hash functions $\mathcal{H}_A : \{0,1\}^* \to \mathbb{Z}_p^*$, $\mathcal{H}_U : \{0,1\}^* \to \mathbb{G}_1$, $\mathcal{H}_O : \{0,1\}^* \to \mathbb{G}_1$, $\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_p^*$.

For CA, define $\mathcal{R}_A = \{((x_{\mathsf{ca}}, R), \mathsf{ca}) : g_A^{x_{\mathsf{ca}}} = R y_A^{\mathcal{H}_A(R||\mathsf{ca})}\}$. For OA, define $\mathcal{R}_O = \{(x_{\mathsf{oa}}, \mathsf{oa}) : x_{\mathsf{oa}} = \mathcal{H}_O(\mathsf{oa})^{x_O}\}$. For user, define $\mathcal{R}_U = \{(x, i) : x = \mathcal{H}_U(i)^{x_U}\}$. For certificate, define $\mathcal{F} = \{\langle \mathcal{R}_{C,i} \rangle : i\}$ with trapdoor x_i. $\mathcal{R}_{C,\mathsf{ca}} = \{(\mathsf{id}, (A, e)) : A^{e+x_{\mathsf{ca}}} \mathcal{H}_U(\mathsf{id}) = u\}$.

Let $g_0, g_1, g_2, g_3, g_4, u$ are generators in $\mathbb{G}_1$. Then:
$\mathsf{param} = (\hat{e}, g_A, y_A, g_O, y_O, g_U, y_U, g_0, ..., g_4, u, \mathcal{H}_A, \mathcal{H}_U, \mathcal{H}_O, \mathcal{H}, \mathcal{R}_A, \mathcal{R}_U, \mathcal{R}_O, \mathcal{F})$.

OKg: On input OA identity oa, the identity manager IM_O uses x_O to compute OA secret key $x_{\mathsf{oa}} = \mathcal{H}_O(\mathsf{oa})^{x_O}$.

GKg: On input CA identity ca, the identity manager IM_A defines $\mathcal{R}_{C,\mathsf{ca}} = \{(\mathsf{id}, (A, e)) : A^{e+x_{\mathsf{ca}}} \mathcal{H}_U(\mathsf{id}) = u\}$ and computes as follows:

1. Randomly generate $r \in \mathbb{Z}_p^*$.
2. Compute $aux_{\mathsf{ca}} = g_A^r$, $x_{\mathsf{ca}} = r + \mathcal{H}_A(aux_{\mathsf{ca}}||\mathsf{ca}) x_A \bmod p$.

This is taken from BNN-IBI [2]. Finally CA gets $(x_{\mathsf{ca}}, aux_{\mathsf{ca}})$.

UKg: On input user identity id, the identity manager IM_U uses x_U to compute user secret key $x_{\mathsf{id}} = \mathcal{H}_U(\mathsf{id})^{x_U}$.

Join,Iss: Common inputs are id, ca. Join's additional input is x_{id} and Iss's additional inputs are $x_{\mathsf{ca}}, aux_{\mathsf{ca}}$. Join firstly runs a proof of knowledge of x_{id} for id. Then Iss uses x_{ca}, aux_{ca} to computes $cert_{\mathsf{id},\mathsf{ca}} = (A, e)$ satisfying $(\mathsf{id}, cert_{\mathsf{id},\mathsf{ca}}) \in \mathcal{R}_{C,\mathsf{ca}}$. Iss randomly selects $e \in \mathbb{Z}_p^*$, and computes $A = (u/\mathcal{H}_U(\mathsf{id}))^{1/(e+x_{\mathsf{ca}})}$. Iss sends $(A, e, aux_{\mathsf{ca}})$ to Join. The validity of aux_{ca} can be checked by BNN's IBI [2]. Note u is a fairly generated public parameter, Join accepts the certificate if and only if $\hat{e}(u, g_A) = \hat{e}(A, g_A)^e \hat{e}(A, S) \hat{e}(\mathcal{H}_U(\mathsf{id}), g_A)$, where $S = g_A^{x_{\mathsf{ca}}} = aux_{\mathsf{ca}} y_A^{\mathcal{H}_A(aux_{\mathsf{ca}}||\mathsf{ca})}$. Finally Join obtains $cert_{\mathsf{id},\mathsf{ca}}, aux_{\mathsf{ca}}$. Iss computes $W = \hat{e}(\mathcal{H}_U(\mathsf{id}), g_A)$, and puts (id, A, e, W) in reg.

GSig: A member id from group ca with secret key x and certificate (A, e) computes a signature σ for message M and oa by

$$SPK\{(\mathsf{id}, x, (A, e), d) : x = \mathcal{H}_U(\mathsf{id})^{x_U} \ \wedge \ A^{e+x_{\mathsf{ca}}} \mathcal{H}_U(\mathsf{id}) = u \\ \wedge \ \mathsf{ctxt} = \hat{e}(\mathcal{H}_U(\mathsf{id}), g_A) \hat{e}(\mathcal{H}_O(\mathsf{oa}), y_O)^d \ \wedge \ U = g_O^d\}(M) \qquad (1)$$

which is equivalent to

$$\begin{aligned} SPK\{(\mathsf{id}, x, (A, e), d) : \hat{e}(x, g_U) = \hat{e}(\mathcal{H}_U(\mathsf{id}), y_U) \\ \wedge\ \hat{e}(u, g_A) = \hat{e}(A, g_A)^e \hat{e}(A, S)\hat{e}(\mathcal{H}_U(\mathsf{id}), g_A) \qquad (2) \\ \wedge\ \mathsf{ctxt} = \hat{e}(\mathcal{H}_U(\mathsf{id}), g_A)\hat{e}(\mathcal{H}_O(\mathsf{oa}), y_O)^d\ \wedge\ U = g_O^d \\ \wedge\ S = aux_{\mathsf{ca}} y_A^{\mathcal{H}_A(aux_{\mathsf{ca}}||\mathsf{ca})}\}(M) \qquad (3) \end{aligned}$$

The further instantiation is as follows. Randomly selects $s_1, d \in \mathbb{Z}_p^*$. Computes $s_2 = es_1$. The masked images are:

$$t_0 = g_0^{s_1}\ \wedge\ t_1 = xg_1^{s_1}\ \wedge\ t_2 = \mathcal{H}_U(\mathsf{id})g_2^{s_1}\ \wedge\ t_3 = Ag_3^{s_1}\ \wedge\ t_5 = t_3^e g_4^{s_1} \qquad (4)$$

And we have: $\mathsf{ctxt} = \hat{e}(\mathcal{H}_U(\mathsf{id}), g_A)\hat{e}(\mathcal{H}_O(\mathsf{oa}), y_O)^d\ \wedge\ U = g_O^d$.

Randomly selects $r_1, r_2, r_3, r_4 \in \mathbb{Z}_p$, $R_1, R_2, R_3 \in \mathbb{G}_1$. Computes:

$$\begin{gathered} \tau_0 = g_0^{r_1}\ \wedge\ \tau_1 = R_1 g_1^{r_1}\ \wedge\ \tau_2 = R_2 g_2^{r_1}\ \wedge\ \tau_3 = R_3 g_3^{r_1} \\ \wedge\ \tau_4 = [\hat{e}(g_1, g_U)^{-1}\hat{e}(g_2, y_U)]^{r_1}\ \wedge\ \tau_5 = t_3^{r_3} g_4^{r_1} \\ \wedge\ \tau_6 = \hat{e}(g_3, g_A)^{r_2}[\hat{e}(g_3, S)\hat{e}(g_2 g_4, g_A)]^{r_1}\ \wedge\ \tau_7 = g_A^{r_4} \\ \wedge\ \tau_8 = \hat{e}(\mathcal{H}_O(\mathsf{oa}), y_O)^{r_4}\hat{e}(g_2, g_A)^{-r_1} \end{gathered}$$

The challenge is:

$$c = \mathcal{H}((t_0, \cdots, t_3, t_5)||(\tau_0, \cdots, \tau_8)||aux_{\mathsf{ca}}||\mathsf{ctxt}||U||M) \qquad (5)$$

The responses are:

$$\begin{gathered} z_0 = r_1 - cs_1\ \wedge\ Z_1 = R_1 x^{-c}\ \wedge\ Z_2 = R_2 \mathcal{H}_U(i)^{-c} \\ \wedge\ Z_3 = R_3 A^{-c}\ \wedge\ z_4 = r_3 - ce\ \wedge\ z_5 = r_2 - cs_2\ \wedge\ z_6 = r_4 - cd \end{gathered}$$

The signature σ is: $(t_0, \cdots, t_3, t_5)||c||(z_0, \cdots, z_6)||aux_{\mathsf{ca}}||\mathsf{ctxt}||U||M$.

GVf: Given a signature σ, it computes:

$$\begin{gathered} t_4 = \hat{e}(t_1, g_U)^{-1}\hat{e}(t_2, y_U)\ \wedge\ t_6 = \hat{e}(u, g_A)^{-1}\hat{e}(t_2 t_5, g_A)\hat{e}(t_3, S) \\ \wedge\ t_8 = \mathsf{ctxt} \cdot \hat{e}(t_2, g_A)^{-1}\ \wedge\ \tau_0 = g_0^{z_0} t_0^c\ \wedge\ \tau_1 = Z_1 g_1^{z_0} t_1^c \\ \wedge\ \tau_2 = Z_2 g_2^{z_0} t_2^c\ \wedge\ \tau_3 = Z_3 g_3^{z_0} t_3^c\ \wedge\ \tau_4 = [\hat{e}(g_1, g_U)^{-1}\hat{e}(g_2, y_U)]^{z_0} t_4^c \qquad (6) \\ \wedge\ \tau_5 = t_3^{z_4} g_4^{z_0} t_5^c\ \wedge\ \tau_6 = \hat{e}(g_3, g_A)^{z_5}[\hat{e}(g_3, S)\hat{e}(g_2 g_4, g_A)]^{z_0} t_6^c\ \wedge\ \tau_7 = g_A^{z_6} U^c \\ \wedge\ \tau_8 = \hat{e}(\mathcal{H}_O(\mathsf{oa}), y_O)^{z_6}\hat{e}(g_2, g_A)^{-z_0} t_8^c\ \wedge\ S = aux_{\mathsf{ca}} y_A^{\mathcal{H}_A(aux_{\mathsf{ca}}||\mathsf{ca})} \end{gathered}$$

Then it computes challenge $\hat{c}$ according to Eq. (5), and compares it to the challenge c received in the signature. If they are equal, output 1 for *valid signature*. In all other cases, output 0.

Open: The open authority uses his secret key x_{oa} to open the encryption in the signature σ. Denote $Q_{\mathsf{oa}} = \mathcal{H}_O(\mathsf{oa})$. He computes:

$$m = \hat{e}(\mathcal{H}_U(\mathsf{id}), g_A) = \mathsf{ctxt}/\hat{e}(x_{\mathsf{oa}}, U)$$

The open authority compares W with the registration table reg. If no such entry is find, output $\perp$. If it is found to be user id, the open authority computes a proof of knowledge of x_{oa} such that $\hat{e}(x_{\mathsf{oa}}, U) = \mathsf{ctxt}/m$:

1. Randomly picks $s'_0 \in \mathbb{Z}_p$. Computes:
   $t'_0 = x_{\mathsf{oa}} h^{s'_0} \ \wedge \ t'_1 = \hat{e}(h,U)^{s'_0} \ \wedge \ t'_2 = \hat{e}(h,g_O)^{s'_0}$.
2. Randomly picks $r'_0, r'_1 \in \mathbb{Z}_p$. Computes:
   $\tau'_0 = Q_{\mathsf{oa}}^{r'_1} h^{r'_0} \ \wedge \ \tau'_1 = \hat{e}(h,U)^{r'_0} \ \wedge \ \tau'_2 = \hat{e}(h,g_O)^{r'_0}$.
3. Computes $c' = \mathcal{H}((t'_0,t'_1,t'_2)||(\tau'_0,\tau'_1,\tau'_2)||\mathsf{ctxt}||U||m)$.
4. Computes $z'_0 = r'_0 - c's'_0, Z'_1 = Q_{\mathsf{oa}}^{r'_1} x_{\mathsf{oa}}^{c'}$.

Outputs the proof $\omega = (t'_0||c'||(z'_0, Z'_1))$ to judge.

Judge: On input $\mathsf{id}, \mathsf{ca}, \mathsf{oa}$, the signature σ and the proof ω, it computes:

$$m = \hat{e}(\mathcal{H}_U(\mathsf{id}), g_A) \ \wedge \ m' = \mathsf{ctxt}/m \ \wedge$$
$$t'_1 = \hat{e}(t'_0, U)/m' \ \wedge \ t'_2 = \hat{e}(t'_0, g_O)\hat{e}(Q_{\mathsf{oa}}, y_O) \ \wedge \tag{7}$$
$$\tau'_0 = Z'_1 {t'_0}^{c'} h^{z'_0} \ \wedge \ \tau'_1 = \hat{e}(h,U)^{z'_0} {t'_1}^{c'} \ \wedge \ \tau'_2 = \hat{e}(h,g_O)^{z'_0} {t'_2}^{c'} \tag{8}$$

Then compares if $c' = \mathcal{H}((t'_0,t'_1,t'_2)||(\tau'_0,\tau'_1,\tau'_2)||\mathsf{ctxt}||U||m)$. If it is true, output 1. Otherwise, output 0.

5 Security Theorems

We now give the security theorems for the above instantiation. It follows the definition in section 3. The proofs can be found in the full version of this paper.

Theorem 2. *The IBGS-SDH scheme is correct.*

Theorem 3. *The IBGS-SDH is anonymous in the random oracle model if and only if the DDH Assumption in $\mathbb{G}_1$ and the co-DBDH Assumption in $(\mathbb{G}_2, \mathbb{G}_1)$ both hold.*

Theorem 4. *The IBGS-SDH is traceable in the random oracle model if and only if the k-CAA2 assumption holds.*

Theorem 5. *The IBGS-SDH is non-frameable in the random oracle model if and only if the co-CDH assumption holds.*

Theorem 6. *The IBGS-SDH is secure if and only if the DDH Assumption in $\mathbb{G}_1$, the co-DBDH Assumption in $(\mathbb{G}_2, \mathbb{G}_1)$, the k-SDH' Assumption, and the co-CDH Assumption all hold in the random oracle model.*

6 Discussions

6.1 Other Instantiation

For the above generic construction, we use a discrete logarithm type of identity based key pairs for CA and pairing type of identity based key pairs for OA and group members to give an instantiation. From [2], we have three identity based identification for discrete logarithm type: Beth [4], Okamoto [21], BNN [2]. They are suitable for constructing the key pairs for both CA and group members. We have different identity based identification for pairing type ([24], [16], [12]). They are suitable for constructing the key pairs for group members. For OA, the key pairs can be obtained from secure identity based encryption which allows efficient verification. Therefore we can form different identity based group signature using different combination of the above key pairs.

For other kinds of certificates in group signature schemes, CA in Ateniese et al [1] has private key (p', q') from the strong RSA assumption. However no existing identity based identification has this form of user key pairs. For Dodis et al [15], there is no CA and the group public key is some accumulated value. Both are not suitable for having identity based group manager.

If one wants the encryption scheme for the open authority to be CCA-2, then we can modify our scheme as follows. We perform the SPK without encryption, and then perform a verifiable encryption scheme from Camenisch and Damgård [9] with Fiat-Shamir heuristic. The encryption scheme used is FullIdent from Boneh and Franklin [7], which is CCA-2. However, the signature size of this scheme will depend on the group size.

6.2 Short Ring Signature

We can formulate our group signature scheme without open authority. We refer this kind of signature scheme as ring signature, as the anonymity of the signature scheme is non-revokeable. It extends the idea of ring signature in [23].

Without the open authority, our signature scheme has signature size independent of the group size. To turn the identity based group signature to a short identity based ring signature scheme, we only have to remove the encryption from GSig. The OA, Open, Judge are also removed. Then short identity based ring signature is constructed.

7 Conclusion

In this paper, we present a fully identity based group signature scheme, with identity based group manager, identity based group members and identity based open authority. We give a generic construction and also an instantiation, which the signature size is independent of the group size. We prove the security of the instantiation in the random oracle model. We also showed that a short identity based ring signature can be formed similarly.

References

1. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *Proc. CRYPTO 2000*, pages 255–270. Springer-Verlag, 2000. Lecture Notes in Computer Science No. 1880.
2. M. Bellare, C. Namprempre, and G. Neven. Security proofs for identity-based identification and signature schemes. In *Proc. EUROCRYPT 2004*, pages 268–286. Springer-Verlag, 2004. Lecture Notes in Computer Science No. 3027.
3. M. Bellare, H. Shi, and C. Zhang. Foundations of group signature: The case of dynamic groups. To appear in CT-RSA 2005, 2005.
4. T. Beth. Efficient zero-knowledged identification scheme for smart cards. In *Proc. EUROCRYPT 88*, pages 77–86. Springer-Verlag, 1988. Lecture Notes in Computer Science No. 330.
5. D. Boneh and X. Boyen. Short signatures without random oracles. In *Proc. EUROCRYPT 2004*, pages 56–73. Springer-Verlag, 2004. Lecture Notes in Computer Science No. 3027.
6. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Proc. CRYPTO 2004*, pages 41–55. Springer-Verlag, 2004. Lecture Notes in Computer Science No. 3152.
7. D. Boneh and M. Franklin. Identity-based encryption from the weil paring. In *Proc. CRYPTO 2001*, pages 213–229. Springer-Verlag, 2001. Lecture Notes in Computer Science No. 2139.
8. E. Brickell, J. Camenisch, and L. Chen. Direct anonymous attestation. Cryptology ePrint Archive, Report 2004/205, 2004. http://eprint.iacr.org/.
9. J. Camenisch and I. Damgård. Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes. In *Proc. ASIACRYPT 2000*, pages 331–345. Springer-Verlag, 2000. Lecture Notes in Computer Science No. 1976.
10. J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. In *Proc. CRYPTO 97*, pages 410–424. Springer-Verlag, 1997. Lecture Notes in Computer Science No. 1294.
11. C. Castelluccia. How to convert any ID-based signature schemes into a group signature scheme. Cryptology ePrint Archive, Report 2002/116, 2002. http://eprint.iacr.org/.
12. J.C. Cha and J.H. Cheon. An identity-based signature from gap diffie-hellman groups. In *Practice and Theory in Public Key Cryptography – PKC'2003*, pages 18–30. Springer-Verlag, 2003. Lecture Notes in Computer Science No. 2567.
13. D. Chaum and E. van Heyst. Group signatures. In *Proc. EUROCRYPT 91*, pages 257–265. Springer-Verlag, 1991. Lecture Notes in Computer Science No. 547.
14. X. Chen, F. Zhang, and K. Kim. A new ID-based group signature scheme from bilinear pairings. Cryptology ePrint Archive, Report 2002/184, 2002. http://eprint.iacr.org/.
15. Y. Dodis, A. Kiayias, A. Nicolosi, and V. Shoup. Anonymous identification in ad hoc groups. In *Proc. EUROCRYPT 2004*, pages 609–626. Springer-Verlag, 2004. Lecture Notes in Computer Science No. 3027.
16. F. Hess. Efficient identity based signature schemes based on pairing. In *Selected Area in Cryptography, SAC2002*, pages 310–324. Springer-Verlag, 2003.
17. M. Joye. On the difficulty coalition-resistance in group signature schemes (ii). *Technique Report*, LCIS-99-6B, 1999.

18. M. Joye, S. Kim, and N. Lee. Cryptanalysis of two group signature schemes. In *Information Security 1999*, pages 271–275. Springer-Verlag, 1999. Lecture Notes in Computer Science No. 1729.
19. W. Mao and C.H. Lim. Cryptanalysis in prime order subgroup of Z_n. In *Proc. ASIACRYPT 98*, pages 214–226. Springer-Verlag, 1998. Lecture Notes in Computer Science No. 1514.
20. S. Mitsunari, R. Sakai, and M. Kasahara. A new traitor tracing. IEICE Trans. Fundamentals, E85-A(2):481-4, 2002.
21. T. Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In *Proc. CRYPTO 92*, pages 31–53. Springer-Verlag, 1992. Lecture Notes in Computer Science No. 740.
22. S. Park, S. Kim, and D. Won. ID-based group signature. In *Electronics Letters, 1997, 33(15)*, pages 1616–1617, 1997.
23. R. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *Proc. ASIACRYPT 2001*, pages 552–565. Springer-Verlag, 2001. Lecture Notes in Computer Science No. 2248.
24. R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairing. In *Proc. of SCIS 2000*, 2000.
25. A. Shamir. Identity-based cryptosystems and signature schemes. In *Proc. CRYPTO 84*, pages 47–53. Springer-Verlag, 1984. Lecture Notes in Computer Science No. 196.
26. Y. Tseng and J. Jan. A novel ID-based group signature. In *International computer symposium, workshop on cryptology and information security*, pages 159–164, 1998.
27. Victor K. Wei. Tight reductions among strong diffie-hellman assumptions. Cryptology ePrint Archive, Report 2005/057, 2005. http://eprint.iacr.org/.
28. F. Zhang, R. Safavi-Naini, and W. Susilo. An efficient signature scheme from bilinear pairings and its applications. In *Proc. PKC'2004*, pages 277–290. Springer-Verlag, 2004. Lecture Notes in Computer Science No. 2947.

Analysis of the HIP Base Exchange Protocol

Tuomas Aura[1], Aarthi Nagarajan[2], and Andrei Gurtov[3]

[1] Microsoft Research, Cambridge, United Kingdom
tuomaura@microsoft.com
[2] Technische Universität Hamburg-Harburg, Germany
aarthi.nagarajan@tu-harburg.de
[3] Helsinki Institute for Information Technology, Finland
gurtov@cs.helsinki.fi

Abstract The Host Identity Protocol (HIP) is an Internet security and multi-addressing mechanism specified by the IETF. HIP introduces a new layer between the transport and network layers of the TCP/IP stack that maps host identifiers to network locations, thus separating the two conflicting roles that IP addresses have in the current Internet. This paper analyzes the security and functionality of the HIP base exchange, which is a classic key exchange protocol with some novel features for authentication and DoS protection. The base exchange is the most stable part of the HIP specification with multiple existing implementations. We point out several security issues in the current protocol and propose changes that are compatible with the goals of HIP.

1 Introduction

The Host Identity Protocol (HIP) is a multi-addressing and mobility solution for the IPv4 and IPv6 Internet. HIP is also a security protocol that defines host identifiers for naming the endpoints and performs authentication and creation of IPsec security associations between them. A new protocol layer is added into the TCP/IP stack between the network and transport layers. The new layer maps the host identifiers to network addresses and vice versa. This achieves the main architectural goal of HIP: the separation of identifiers from locations. In the traditional TCP/IP architecture, IP addresses serve both roles, which creates problems for mobility and multi-homing.

The *host identity* (HI) in HIP is a public key. This kind of identifier is self-certifying in the sense that it can be used to verify signatures without access to certificates or a public-key infrastructure. The host identity is usually represented by the *host identity tag* (HIT), which is a 128-bit hash of the HI. IPv4 and IPv6 addresses in HIP are purely locations. The protocol is composed of three major parts. The endpoints first establish session keys with the HIP *base exchange* [10], after which all packets are protected using IPsec ESP [9]. Finally, there is a readdressing mechanism to support IP address changes with mobility and multi-homing.

In this paper, focus on the HIP base exchange as a cryptographic key-exchange protocol. We analyze its security with emphasis on denial-of-service (DoS) issues. Several protocol details were found to be vulnerable to DoS attacks or accidental deadlocking. Additionally, we point out a minor issue with key freshness. To fix these problems, we propose feasible solutions that are in line with the goals of the HIP

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 481-493, 2005.

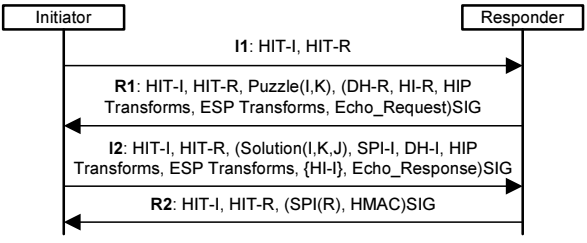


Figure 1 HIP base exchange messages

protocol. (Note that this paper does not fully explain the thinking behind the HIP protocol [11] and we do not try argue in favor of or against adopting HIP as a part of the Internet protocol stack.) We start by introducing the HIP base exchange in Section 2 and discuss the identified problems and solutions in the following sections.

2 HIP Base Exchange

The main building block of the HIP protocols is the HIP base exchange [10]. It is used to establish a pair of IPsec security associations (SA) between two hosts. The base exchange is built around a classic authenticated Diffie-Hellman key exchange but there are some unusual features related to DoS-protection. No certificates are required for the authentication because the HITs are self-certifying. The protocol can be compared with key exchange protocols like IKE [7] and IKEv2 [4] and evaluated against most of the same security requirements [13].

In this section, we outline the base exchange messages I1, R1, I2 and R2 shown in Figure 1. The Initiator first sends an empty message I1 to the Responder. It triggers the next message R1 from the Responder. All HIP packets contain the initiator and responder identity tags (HIT-I and HIT-R) in the header.

Even before the Responder receives the I1 message, it precomputes a partial R1 message. The precomputed R1 includes the HIT-R, the Responder's Diffie-Hellman key, the Responder host identity HI-R (i.e., a public key), the proposed cryptographic algorithms for the rest of the base exchange (HIP transforms), the proposed IPsec algorithms (ESP transforms), and an Echo_Request field. The Echo_Request contains data that the Initiator returns unmodified in the following message I2. It is important that the responder sends R1 without creating any protocol state. The Echo_Request can be used to store some data in a stateless way. The responder signs the message. The HIT-I and the Puzzle field are left empty at this point. These two fields are populated after receiving an I1 and they are not protected by the signature.

The Puzzle parameter in R1 contains a cryptographic puzzle [3,4], which the Initiator is required to solve before sending the following packet I2. The idea is that the Initiator is forced to perform a moderately expensive brute-force computation before the Responder commits its computational resources to the protocol or creates a protocol state. The puzzle has three components: the puzzle nonce I, the difficulty level K, and the solution J. It is easiest to explain how a puzzle solution is verified: First, concatenate I, the host identity tags HIT-I and HIT-R, and the solution J. Then,

compute the SHA-1 hash of the concatenation. Finally, check that the K low-order bits of the hash are all zeros.

$$\text{Ltrunc}(\text{SHA-1}(\text{I} \mid \text{HIT-I} \mid \text{HIT-R} \mid \text{J}), \text{K}) == 0$$

The Initiator must do a brute-force search for the value of J, which takes $O(2^K)$ trials. The Responder, on the other hand, can verify the solution by computing a single hash.

On receiving R1, the initiator checks that it has sent a corresponding I1 and verifies the signature using the public key HI-R. If the signature is ok, it solves the puzzle and creates the message I2. I2 includes the puzzle and its solution, the Initiator's Diffie-Hellman key, the HIP and ESP transforms proposed by the Initiator, a security parameter index (SPI) for the Responder-to-Initiator IPsec SA, the Initiator public key (HI-I) encrypted using the new session key, and the Echo_Response. A signature covers the entire message. Key material for the session keys is computed as a SHA-1 hash of the Diffie-Hellman shared secret Kij:

$$\text{KEYMAT\_k} = \text{SHA-1}(\text{Kij}, \mid \text{sort}(\text{HIT-I} \mid \text{HIT-R}) \mid \text{k}) \quad \text{for k} = 1, 2,...$$

On receiving I2, the Responder verifies the puzzle solution. If it is correct, the Responder computes the session keys, decrypts HI-I, and verifies the signature on I2. The Responder then sends R2, which contains the SPI for the Initiator-to-Responder IPsec SA, an HMAC computed using the session key, and a signature.

For the Initiator, the exchange is concluded by the receipt of R2 and the verification of the HMAC and the signature. The HMAC confirms the establishment of the session key. For the Responder, the key confirmation is provided by the first inbound IPsec packet that is protected with the new security association.

3 Replays of R1

In this section, we consider replays of the R1 message. As explained earlier, R1 is partially signed. There is, however, nothing in R1 to prove its freshness.

Before explaining why we think the freshness of R1 should be checked, we'll consider arguments against such protection. First, it is infeasible to include a nonce in the signed part of R1 because that would prevent the Responder precomputing the signature. Thus, nonce-based replay protection appears not to work. Second, timestamps have well-known problems with clock synchronization. Third, there are features in the protocol that mitigate the consequences of R1 replays. The signature on the last message, R2, covers the session key (indirectly by covering the HMAC). Thus, the Initiator detects the replay of an old Diffie-Hellman key in R1 when it receives R2. For these reasons, the freshness of R1 many not appear very important.

There is, however, another type of attack based on replaying R1: the attacker spoofs R1 and tricks the Initiator into solving the wrong puzzle. The attacker can send a trickle of replayed R1 messages to the Initiator with random I values. If the frequency of the spoofed R1 messages is higher than the roundtrip between the Initiator and Responder, the first R1 to arrive at the Initiator after it has sent I1 is always a replay. This prevents the Initiator from ever solving an authentic puzzle.

Problem 1: Attacker can replay the signed parts of R1 and trick the Initiator into solving the wrong puzzle. This results in denial-of-service for the Initiator because the solution is rejected by the Responder.

The seriousness of the above DoS attack is increased by the ease of obtaining the replay material. No sniffing is necessary; the attacker can obtain the partially signed R1 by sending an I1 message to the same Responder.

There is a simple mechanism for preventing the attack: include a nonce in I1 and in the unsigned part of R1. The nonce prevents the attacker from replaying R1 unless it can sniff the corresponding I1. The cost of adding the nonce is low and it reduces significantly the threat of R1 replays.

Proposed solution 1: Add a nonce of the Initiator to I1 and to the unsigned part of R1 to prevent replays of R1.

The lack of this kind of "cookies" in the current HIP specification may be the influence of [3] where the first protocol message was potentially a broadcast message and, thus, could not contain a per-responder nonce. In HIP, the first message I1 is always unicast and therefore a nonce can be added.

4 Reuse of Diffie-Hellman Keys

Diffie-Hellman public keys (g^x and g^y) are often reused in order to amortize the cost of public-key generation over multiple key exchanges. The trade-off is the loss of forward secrecy: old session keys can be recovered as long as the key owner stores the private exponent. Another consequence of the key reuse is that the Diffie-Hellman shared secret (Kij = g^{xy}) is not guaranteed to be fresh. The usual solution is to compute the session key as a hash of Kij and nonces from both participants.

In the HIP base exchange, Diffie-Hellman keys are sometimes reused. First, the Responder uses the same key in all R1 messages over a time period (Delta). Second, the same host acting simultaneously as the Initiator and as the Responder uses the same key in both roles. (While the reuse is not mandated by the specification, it is probably necessary in practice to avoid further complicating the protocol state machine. See Section 7.) Nevertheless, the HIP base exchange does not include nonces in the session-key computation.

The lack of nonces may, in fact, lead to a vulnerability. If the same two hosts perform the base exchange twice within the time Delta (i.e., the time during which Diffie-Hellman keys are reused), they end up with the same session keys. In practice, this depends on the timing in the implementations at each end-point. Such dependence on the implementation detail is, of course, not acceptable in a security protocol.

Problem 2: Reuse of Diffie-Hellman keys may result in reuse of session keys.

The puzzle I and the solution J are, in effect, nonces of the Responder and Initiator. Thus, they can be used for freshness in the session-key generation:

$$\text{KEYMAT\_k} = \text{SHA-1}(\text{Kij} \mid \text{sort}(\text{HIT-I} \mid \text{HIT-R}) \mid \text{I} \mid \text{J} \mid \text{k})$$

Proposed solution 2: The nonces I and J should be hashed into the key material.

5 Puzzle Implementation

The client puzzles force the Initiator to perform a moderately expensive computation before the Responder commits its computational resources or creates a protocol state. The Initiator, in effect, pays for the resources of the Responder by solving the puzzle, which is why this kind of mechanism is sometimes called hash cash. During DoS attacks or otherwise heavy load, the Responder increases the price. The HIP puzzle mechanism originates from [3] but some changes have been made in order to address specific security concerns. In this section, we analyze the HIP puzzles and suggest changes that are compatible with the current implementations.

The following requirements for the puzzles can be derived from [10] or [3]:

1. The Responder must not verify the signature on I2 or do other expensive computation before it has verified the puzzle solution. It must verify at most one signature for each puzzle solved by the Initiators (or attackers).
2. The Responder must not create a per-Initiator or per-session state before verifying the puzzle solution. It must only store a small amount of information for each puzzle solved by the Initiators (or attackers).
3. The cost of creating and verifying a puzzle must be small, preferably requiring only one computation of a one-way hash function by the Responder.
4. The attacker must not be able to pre-compute solutions for a burst attack. That is, the solutions must remain valid only for a short time period.
5. The Responder must not reject correct solutions sent by an honest Initiator because the attacker has previously solved the same puzzle.
6. In order to verify the Initiator IP address, the Responder must recognize I in I2 as the same nonce that it previously sent to the source address of I2.
7. The Responder must accept at most one correct puzzle solution for each I1/R1 exchange that takes place. (We will later argue that Requirement 7 is unnecessary even though it is implied in [10].)

A naive puzzle implementation will send a random number I in every R1 and store the random number until it either receives a solution or a time Delta has passed. The naive puzzles fail Requirement 2 because a small state is created for each received I1. An attacker can exploit this by flooding the Responder with I1 messages.[1]

The puzzle implementation proposed in [10] (Appendix D) tries to address all of the above requirements. The basic idea is that the Responder has a fixed-size table of pre-generated random I values (I_k for k=0...n-1), called *cookies*, and it selects one of them for each R1 message by computing the index to the cookie table as a function of HIT-I and HIT-R and the Initiator and Responder IP addresses (IP-I and IP-R). The function is not a strong cryptographic hash but an inexpensive combination of XOR operations. Upon receiving a solution in I2, the Responder recomputes the index k to the cookie table and checks that the I in I2 matches the value in the table. The Responder then verifies the puzzle solution by computing a SHA-1 hash. If the solution is correct, it marks the nonce I_k as used. Only the first solution to each puzzle

[1] The naive algorithm is presented in Section 4.1.1 of [10]. The rest of the specification talks about the cookie-table mechanisms.

is accepted. A background process replaces the used nonces I_k with new ones within the time period Delta.

Unfortunately, the pseudo-random function described in the specification is linear and it is easy for the attacker to create a collision with the honest Initiator. The function is simply an XOR of the Responder's secret 1-byte key r and the bytes of the Initiator and Responder HITs and IP addresses:

index = XOR of r and all bytes of HIT-I, HIT-R, IP-I and IP-R

The attacker can cause an index collision with an honest Initiator by selecting the Initiator HIT in the spoofed I2 message as follows:

HIT-A[m] = HIT-I[m] for m=0,...,14
HIT-A[15] = XOR of HIT-I[15] and all bytes of IP-A and IP-I

The attacker obtains an R1 by sending an I1 from its own IP address (IP-A), solves the puzzle for HIT-A and IP-A, and then sends an I2 from IP-A using HIT-A as the initiator HIT. The Responder accepts this solution and rejects the one sent later by the honest initiator because the puzzle has already been used. The signature on the attacker's I2 is invalid because HIT-A is not a hash of any public key, but the Responder must mark the puzzle used even when the signature is invalid.

Problem 3: The pseudo-random function for selecting the puzzle is linear. Thus, the attacker can cause collisions and consume puzzles of honest initiators by solving them.

The obvious solution is to compute the index with a second-preimage-resistant hash function. The trade-off is that this adds two hash computations to the total cost of creating and verifying a puzzle. We suggest using a standard function like SHA-1 because it would be non-trivial to design a lower-cost non-linear function that nevertheless has sufficient strength to match puzzle difficulties up to $O(2^{64})$.

Proposed solution 3: The pseudo-random function used to select the value of I should be a strong hash function, such as SHA-1.

There is another problem with the cookie table. Even if we use SHA-1 to index the table, the attacker can still solve a lot of puzzles so that a significant portion of the puzzles in the table remain used at any point of time. The attacker can do this from its own IP address by picking random HITs and by solving the puzzles for them. For example, in order to consume n/2 nonces, the attacker has to solve approximately 0.69*n puzzles. The suggested table size is n=256, which means that the attacker needs to solve about 177 puzzles during the time Delta to cause the exchange to fail for 50% of honest Initiators. Even with a larger table, the birthday paradox ensures that the attacker can block out a small number of legitimate connection attempts.

Problem 4: With any realistic cookie-table size, the attacker can cause some index collisions and, thus, authentication failures.

The purpose of the cookie table is to implement Requirements 6 and 7 above, i.e., to bind the Initiator IP address to the puzzles and to prevent the reuse of puzzles. If an attacker wants to flood the Responder with correct puzzle solutions, it has to repeat the I1/R1 exchange and it must use its own IP address.

We suggest a way of generating puzzles that does not require the Responder to store a table of nonces: compute the puzzle I as a hash of the Initiator HIT and IP address, and a periodically changing secret key K_{Res} known only to the Responder.

$$I = \text{SHA-1}(\text{IP-I} \mid \text{IP-R} \mid \text{HIT-I} \mid \text{HIT-R} \mid K_{Res})$$

The Responder does not need to store any information after sending this puzzle in R1. When it receives I2, it can recompute I from the information in that message.

Proposed solution 4: In order to bind the puzzle to the Initiator IP address, compute the puzzle I as a SHA-1 hash of the address. This gives the same level of security as the cookie-table with the same computational cost and less memory.

It should be noted that the cookie-table mechanism implements Requirement 7 but our alternative solution does not. That is, we do not prevent the Initiator from solving the same puzzle multiple times during the time Delta. It is not clear what would be achieved by forcing the Initiator to perform the I1/R1 exchange more frequently.

Next, we turn our attention to another feature in the puzzle mechanism that does not achieve its intended purpose. It is suggested in [10] (Section 4.1.1 and in Appendix D) that if the Responder receives multiple false solutions from the same IP address and HIT, it should block further I2 messages from this source for a period of time. The problem is that this contradicts Requirement 2, i.e., not creating any per-Initiator state at the Responder until a correct puzzle solution is verified. The attacker can exhaust the blocking mechanism by flooding the Responder with false puzzle solutions from spoofed IP addresses.

Problem 5: If the Responder blocks I2 packets from HITs or IP addresses after receiving false cookie solutions, the blocking mechanism is vulnerable to a flooding attack.

It is not easy to define any robust criteria for filtering incoming puzzle solutions without verifying them. Any such filtering mechanism can probably be circumvented or, worse, exploited in DoS attacks. It is, therefore, better to design the system without the blocking.

Proposed solution 5: The responder should not create any state after receiving false puzzle solutions.

Finally, we suggest a complete puzzle mechanism that solves the above problems while maintaining compatibility with the current HIP specification.

- The Responder has a secret key K_{Res} that it generates periodically, once in every Delta. The Responder remembers the two last values of K_{Res}.
- The Responder uses one pre-signed message R1. The signature is recomputed periodically when the Diffie-Hellman key is replaced, which happens slightly less often than the generation of a new K_{Res}.
- The Responder computes the puzzle nonce I as the SHA-1 hash of the newest K_{Res} and the Initiator and Responder HITs and IP addresses. It computes the value of I on the fly for each R1 and forgets the value after sending R1.

- A 1-bit key counter is incremented every time a new key K_{Res} is generated. The value of this counter is sent in R1 and I2 with the puzzle.[2]
- On receiving I2, the Responder recomputes I from K_{Res} and the HITs and IP addresses, which it takes from the I2 message. The correct K_{Res} is identified by the 1-bit counter.
- The Responder then compares the computed value of I with the one in I2. If the values match, it verifies the puzzle solution J by computing the SHA-1 hash.
- If the puzzle solution is correct, the Responder stores the puzzle I and the correct solution J. (Alternatively, the Responder can store the Initiator HIT and IP address and the correct solution J.) Separate storage is maintained for the latest and second latest value of K_{Res}. When the older K_{Res} is deleted, the corresponding storage is purged as well.

6 Encryption of Initiator HI in I2

The Initiator host identity HI-I in the I2 message is encrypted with the new session key. In this section, we argue that the encryption is unnecessary and bad for security.

In some key exchange protocols, such as IKE [4], the endpoint identifiers, or the certificates containing the identifiers, are encrypted to enhance user privacy. The host identifiers in HIP could be similarly encrypted to prevent an eavesdropper from identifying the hosts. There are, however, several reasons why the encryption does not make sense. First, the privacy issue is mitigated by the fact that the host identifiers are public keys and not user or machine names. Second, the HITs that appear in every message header are hashes of the host identifier and, thus, uniquely identify the hosts. Encrypting the HI achieves little as long as the HIT is sent unprotected. Third, the Responder identity in R1 is sent in plaintext. An attacker that impersonates the Responder can easily discover the Initiator's identity by reversing the roles and sending an I1 message to the Initiator. It will receive the peer's HI in R1. (The HIP specification suggests that a privacy-conscious HIP host may refuse to act as the Responder but that will lead to communications failure if both endpoints follow the same policy.) For these reasons, the encryption is ineffective as a privacy mechanism.

There could be other reasons for the encryption. First, it could be a freshness check: the encryption is a function of the session key, which is a function of the Responder's Diffie-Hellman key, which changes for every time Delta. Thus, valid encryption links message I2 to a fresh value generated by the Responder. But this freshness check is superfluous because the puzzle nonce I already provides freshness for the I2 message. Second, the encryption could serve as key confirmation: by encrypting with the session key, the Initiator proves to the Responder that it knows the session key. But this is clearly not the intention because there is a separate mechanism for key confirmation. (The protocol state machine requires the Responder to wait in the R2-SENT state for a valid ESP packet from the Initiator before moving

[2] The counter bit can be sent in the Opaque field of the puzzle data structure or in an Echo_Request field, both of which are returned unmodified in I2.

to the ESTABLISHED state.) Hence, neither freshness nor key confirmation is a valid motivation for encrypting the HI.

In addition to being unnecessary, the encryption of the Responder HI prevents NAT and firewall support [1,5,6] for HIP. The catch is that when the HI is encrypted, middle boxes in the network cannot verify the signature on I2 and, thus, cannot safely create a state for the HIP association. On the other hand, if the HI is not encrypted, a stateful middle box like a NAT can process I2 and create a protocol state for the Initiator. A firewall can also verify the puzzle and signature on I2, thus making it possible to push the I1/R1 exchange into the firewall and to filter false puzzle solutions at the firewall. The encryption of HI-I prevents such middle-box implementations. (See [11,14,15] for details.)

Problem 6: The encryption of the Initiator HI in the I2 message does not provide any privacy protection and prevents HIP support in firewalls and NATs.

The solution is obvious:

Proposed solution 6: Do not encrypt the Initiator HI in I2.

7 State Machine Issues

The base-exchange protocol state machine is shown in Figure 2. The Initiator moves from the UNASSOCIATED state via I1-SENT, on receiving R1, to I2-SENT and, finally, on receiving R2, to ESTABLISHED. The Responder remains in UNASSOCIATED until it receives and verifies I2, after which it moves to R2-SENT and, finally, on receiving a valid ESP packet, to ESTABLISHED. These basic state transitions are sensible and have been thoroughly tested. On the other hand, the recovery from packet loss and handshake collisions, where the hosts act simultaneously in the Initiator and Responder roles, is ad-hoc and not fully specified. In this section, we discuss problems with these less frequent state transitions.

The first issue is that there is a timeout transition for the Responder from the R2-SENT to ESTABLISHED. The reason for waiting for the ESP packet is key confirmation: only after receiving the ESP packet the Responder knows that the Initiator received a fresh Diffie-Hellman key (rather than a replay) in R1 and has computed the valid session key. The timeout transition breaks this mechanism.

Problem 7a: The timeout transition to the ESTABLISHED state breaks key confirmation.

An alternative would be a timeout transition to UNASSOCIATED instead of the ESTABLISHED state. This could, however, cause an infinite loop (livelock) in which the hosts never reach the ESTABLISHED state, even if no messages are lost or spoofed. Consider the following sequence of events: both hosts initiate the protocol by sending I1 and move to the I1-SENT state; both hosts receive I1 and respond with R1; both hosts receive R1, respond with I2 and move to I2-SENT; both hosts receive I2, send R2 and move to the R2-SENT state; both hosts receive R2 and drop it; after a timeout, both hosts move back to UNASSOCIATED and start the same process form the beginning.

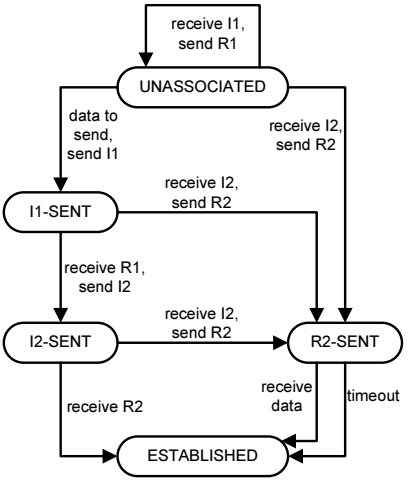


Figure 2 Partial protocol state machine.

Problem 7b: If the timeout transition is changed to lead to UNASSOCIATED instead of ESTABLISHED, a livelock may occur where neither party ever reaches ESTABLISHED.

We can resolve this problem by creating an asymmetry between the participants. A convenient place to do that is when both hosts are in the I2-SENT state and both receive an I2 packet. One of the hosts should continue in the Initiator role and the other should assume the Responder role. (In the current state-machine specification, both move to R2-SENT, i.e., select the Responder role.) One way to assign the roles to the hosts is to compare their HITs as if they were integers. We have (arbitrarily) decided to make the host with lower HIT the Initiator.

Proposed solution 7:
(a) The timeout transition from the R2-SENT state should lead to the UNASSOCIATED state.
(b) In I1-SENT, if a host receives I1 and the local HIT is lower than the peer HIT, the host should drop the received I1 and remain in I1-SENT. Otherwise, the host should process the received I1, send R1, and remain in I1-SENT.
(c) In I2-SENT, if a host receives I2 and the local HIT is lower than the peer HIT, the host should drop the received I2 and remain in I2-SENT. Otherwise, the host should process the received I2, send R2, and go to R2-SENT.

This way, both hosts cannot end up in the R2-SENT state at the same time and liveness of the protocol is guaranteed unless there is repeated message loss. The difference between solutions 7(b) and 7(c) is that 7(b) saves two messages but 7(c) is more robust in the presence of middle boxes. We suggest implementing both.

Above, we have only considered the protocol states as listed in the HIP specification. In reality, the state also comprises the SPI values, HIP and ESP transforms, and cryptographic keys. In the following, we will consider how these values are treated in the state transitions, in particular, when two handshakes collide.

The Diffie-Hellman keys and the HIP and ESP transforms are selected together and can be considered parts of the same data structure. We will only discuss the keys but the same considerations apply to the transforms as well. If the nonces I and J are used in the session-key generation (see Section 3), we also need to consider the nonces a part of the protocol state.

It is not specified when the Diffie-Hellman keys should be replaced. This is particularly a problem if the handshakes collide and new Diffie-Hellman keys are generated during the exchange. In that case, the hosts may end up with inconsistent views of the keys. Specifically, if a host receives I2 in the I2-SENT state, which *peer* Diffie-Hellman key should it use for computing KEYMAT, the one from the just-arrived I2 or the one received earlier in R1? And which *local* Diffie-Hellman key should it use, the one it sent in I2 or the one it sent earlier in R1? The consistency problem is even more acute for the nonces I and J because they are always fresh. It turns out that if the hosts behave symmetrically, as a straight-forward implementation would do, any choice of keys would be wrong.

Problem 8: It is not specified which Diffie-Hellman keys (and nonces) a host should use to compute the session key if it receives I2 in the I2-SENT state. If the hosts behave symmetrically, they may end up with different session keys.

We assume that the proposed solutions 7(a) and (c) are implemented by all hosts but 7(b) only by some. The following rules can then be used to select consistent keys and nonces.

Proposed solutions 8: In the I2-SENT state, if a host receives I2 and the local HIT is lower than the peer HIT, the host should use the *peer* Diffie-Hellman key and nonce I from the R1 packet it received earlier. It should also use the *local* Diffie-Hellman key and nonce J from the I2 packet it sent earlier. Otherwise, it should use the *peer* Diffie-Hellman key and nonce J from the just-received I2, and the *local* Diffie-Hellman key and nonce I from the R1 packet it sent earlier.

A similar confusion occurs with the SPI values in the unmodified protocol. It is not specified which SPI values should be used to create the IPsec SAs if handshakes collide. It is possible that the hosts end up with different pairs of SPIs. The ambiguity is resolved by either one of the protocol changes 7(b) and 7(c).

We formalized the protocol state machine, including the Diffie-Hellman keys and SPIs, and used the Zing model checker to verify deadlock-freeness and consistency of the keys, nonces and SPIs after our proposed changes to the protocol. Further work is required to verify the absence of livelocks.

8 Conclusion

In this paper, we analyzed the security of the HIP base exchange protocol. The base exchange is a fairly basic authenticated Diffie-Hellman key exchange that is further simplified by the fact that the host identities in HIP are self-certifying. We did not find major attacks against the authentication and key-exchange apart from a small issue with the freshness of the Diffie-Hellman keys.

The more interesting security properties in the HIP base exchange relate to denial-of-service: the protocol uses client puzzles with several novel details to protect against resource-exhaustion DoS attacks. We showed that these details require modification to provide the intended protection. In particular, an attacker was able trick the honest Initiator into solving the wrong puzzles, and it was able to consume the puzzles of the honest Initiator by solving them first. Moreover, the protocol state machine requires changes to prevent deadlocks and livelocks.

Our analysis of the abstract protocol complements in an important way the detailed protocol design, specification, implementation, and testing that happens during the IETF standardization process. It should be remembered that HIP is as much a multi-addressing and mobility protocol as a security protocol. We suggested solutions to all the discovered problems and believe that the proposed protocol changes fit well into the HIP framework without compromising its original goals.

References

1. Bernard Aboba and William Dixon. IPsec-network address translation (NAT) compatibility requirements. RFC 3715, IETF, March 2004.
2. Tony Andrews, Shaz Qadeer, Sriram K. Rajamani, Jakob Rehof, and Yichen Xie. Zing: Exploiting program structure for model checking concurrent software. In Proc. CONCUR 2004, volume 3170 of LNCS. Springer, August 2004.
3. Tuomas Aura, Pekka Nikander, and Jussipekka Leiwo. DOS-resistant authentication with client puzzles. In Proc. Security Protocols Workshop 2000, volume 2133 of LNCS, pages 170-181, Cambridge, UK, April 2000. Springer.
4. Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In Advances in Cryptology - Proc. CRYPTO '92, volume 740 of LNCS, pages 139-147, Santa Barbara, CA USA, August 1992. Springer.
5. Kjeld Borch Egevang and Paul Francis. The IP network address translator (NAT). RFC 1631, IETF, May 1994.
6. Ned Freed. Behavior of and requirements for Internet firewalls. RFC 2979, IETF, October 2000.
7. Dan Harkins and Dave Carrel. The Internet key exchange (IKE). RFC 2409, IETF Network Working Group, November 1998.
8. Charlie Kaufman (Ed.). Internet key exchange (IKEv2) protocol. Internet-Draft draft-ietf-ipsec-ikev2-17, IETF IPsec WG, September 2004. Work in progress.
9. Stephen Kent and Randall Atkinson. IP encapsulating security payload (ESP). RFC 2406, IETF, November 1998.
10. Robert Moskowitz, Pekka Nikander, Petri Jokela, and Thomas R. Henderson. Host identity protocol. Internet Draft draft-ietf-hip-base-01, IETF HIP WG, October 2004. (http://www.watersprings.org/pub/id/draft-ietf-hip-base-01.txt)
11. Aarthi Nagarajan. Security issues of locator-identifier split and middlebox traversal for future Internet architectures. Master's thesis, Technische Universität Hamburg-Harburg, Germany, November 2004.
12. Pekka Nikander, Jukka Ylitalo, and Jorma Wall. Integrating security, mobility, and multi-homing in a HIP way. In Proc. NDSS '03, pages 87-99, San Diego, CA USA, February 2003.
13. Radia Perlman and Charlie Kaufman. Key exchange in IPSec: Analysis of IKE. IEEE Internet Computing, 4(6):50-56, November/December 2000.

14. Hannes Tschofenig, Aarthi Nagarajan, Vesa Torvinen, Jukka Ylitalo, and Murugaraj Shanmugam. NAT and firewall traversal for HIP. Internet-Draft draft-tschofenig-hiprg-hip-natfw-traversal-00, October 2004. Work in progress.
15. Hannes Tschofenig, Aarthi Nagarajan, Murugaraj Shanmugam, Jukka Ylitalo and Andrei Gurtov. Traversing Middle Boxes with Host Identity Protocol. Proc. ACISP '05, July 2005, Brisbane, Australia.

ID-based Authenticated Key Agreement for Low-Power Mobile Devices*

Kyu Young Choi[1], Jung Yeon Hwang[1], Dong Hoon Lee[2], and In Seog Seo[3]

[1,2] Center for Information Security Technologies(CIST),
Korea University, Seoul, Korea
[3] National Security Research Institute(NSRI), Daejeon, Korea
[1]{young,videmot}@cist.korea.ac.kr
[2]{donghlee}@korea.ac.kr, [3]{isseo}@etri.re.kr

Abstract. In this paper we present an efficient ID-based authenticated key agreement (AKA) protocol by using bilinear maps, especially well suited to unbalanced computing environments : an ID-based AKA protocol for *Server* and *Client*. Particularly, considering low-power clients' devices, we remove expensive operations such as bilinear maps from a client side. To achieve our goal we combine two notions, key agreement and ID-based authenticryption in which only designated verifier (or *Sever*) can verify the validity of a given transcript. We prove the security of our ID-based AKA protocols in the *random oracle* model.

1 Intoduction

Background. Key agreement protocols are among the most basic and widely used cryptographic protocols. A key agreement protocol allows two or more participants to share a key which may later be used to achieve some cryptographic goals. In addition, an authentication mechanism provides an assurance of key-sharing with intended users. A protocol achieving these two goals is called an authenticated key agreement (AKA) protocol.

Among various authentication flavors, asymmetric techniques such as certificate based system and ID-based system are commonly used to provide authentication. In a typical PKI (public key infrastructure) deployed system, a user should obtain a certificate of a long-lived public key from the certifying authority and this certificate is given to participants to authenticate the user. Meanwhile in an ID-based system, participants just have to know the public identity of the user such as e-mail address. Thus, compared to certificate-based PKI systems, ID-based authenticated systems simplify key management procedures.

This simplicity in key management may be exploited for constructing a secure channel in unbalanced computing environments such as wireless networks where some of communicating parties are using low-power devices and hence PKIs

* This research was supported by the MIC(Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment)

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 494–505, 2005.

can not be utilized. However, to apply ID-based authenticated systems to these resource-restricted applications, only small amount of computation should be done on low-power mobile devices. Unfortunately, most ID-based authenticated systems in the literature can not be directly applied to strictly resource-restricted applications because of their expensive computation on low-power devices.

Contribution. In this paper we present an efficient ID-based authenticated key agreement protocol for low-power mobile devices by using bilinear maps: an ID-based AKA protocol for two participants *Server* and *Client*.

To take advantage of an ID-based key management system in unbalanced computing environments, simple application of previously known ID-based AKA protocols with bilinear maps is not suitable. Although ID-based systems based on bilinear maps can be fully functional, computation of bilinear maps such as Weil/Tate pairing and Map-To-Point operation converting a hash value of ID to a point on an elliptic curve, are relatively expensive for low-power devices. Despite of some attempts [3,4,10] for reduction in the complexity of pairing, a pairing operation is still 5-10 times compared to a scalar multiplication. In most ID-based AKA protocols suggested for normal network circumstance, such a costly computation is required in a client side, which is an obstacle to be overcome for ID-based AKA protocols to be applied to low-power mobile devices.

In our AKA construction using bilinear maps, we firstly remove both complicate operations of bilinear maps and Map-To-Point from a client side. In addition, using off-line precomputation, the client computes only two scalar multiplications and one addition during on-line phase. To achieve our goal we combine two notions, key agreement and ID-based authenticryption in which only designated verifier (*Sever*) can verify the validity of a given transcript.

We prove the security of the proposed ID-based AKA protocol under the intractability of k-CAA (collusion attack algorithm with k traitors) and k-mBIDH (modified Bilinear Inverse DH with k values) problems in the *random oracle* model. The protocol achieves *half forward secrecy* in the sense that exposure of client's long-lived secret key does not compromise the security of previous session keys, while exposure of server's long-lived secret key does compromise the security of previous session keys. An additional feature of the proposed protocol is a key agreement between members of distinct domains. For example, mobile clients can establish a secure channel with the server of other domain. Also the proposed AKA protocol can be expanded to a centralized ID-based group AKA protocol, i.e., one *Sever* and multiple *Clients*.

Related Work. Since the original two-party Diffie-Hellman key agreement protocol appeared in [9], authenticated key agreement problems have been extensively researched. In particular, Bellare and Rogaway adapted so-called *provable security* to a key exchange and firstly provided formal framework [5,6]. Based on that model, many subsequent works have identified concrete AKA problems for various situations. Huang et al. proposed a certificate-based key agreement between sensor node and security manager in the sensor network by using ECC cryptosystems [11]. Bresson et al. proposed a certificate-based authenticated

group key agreement protocol [1] for low-power mobile devices. Their protocol has been found to be vulnerable to the parallel session attack [18]. Recently, Katz and Yung [14], and Hwang et al. [12] proposed modular methods to establish contributory key agreement (KA) protocols. The combination of these two methods provides an efficient mechanism to extend two-party KA protocols to 3-round group AKA protocols. Very recently, Kim et al. [13] proposed an efficient contributory group AKA protocol for low-power mobile devices. These AKA protocols are all based on PKIs.

Based on the notion of ID-based cryptosystem introduced by Shamir [22], several ID-based AKA protocols using bilinear maps have been proposed. But the results in [19,21] only present informal analysis for the security of the proposed protocols and some of these protocols are later found to be flawed [19]. Even secure ID-based protocols [16,8] in the literature are not suitable to unbalanced computing environments because of requirement of costly pairing operations on low-power mobile devices.

Organization. The remainder of this paper is organized as follows. We define our security model in Section 2. We review bilinear maps and cryptographic assumptions needed in Section 3. We present our ID-based AKA protocol and security analysis in Section 4.

2 The Model for ID-based AKA Protocol

The model described in this section extends one of Bresson et al. [7] which follows the approach of Bellare et al. [5,6].

2.1 Security Model

We assume that mobile client $\mathcal{U}$ and server $\mathcal{V}$ have unique identities ID_U and ID_V from $\{0,1\}^\ell$, respectively. In the model we allow client $\mathcal{U}$ to execute a protocol repeatedly with server $\mathcal{V}$. *Instances* of $\mathcal{U}$ (resp. $\mathcal{V}$) model distinct executions of the protocol. We denote instance s of $\mathcal{U}$ (resp. $\mathcal{V}$), called an oracle, by Π_U^s (resp. Π_V^s) for an integer $s \in \mathbb{N}$. The public parameters params and identities $\mathcal{ID} = \{ID_U, ID_V\}$ are known to client and server (and also to an adversary).

Adversarial model. To define the notion of security, we define the capabilities of an adversary. We allow a probabilistic polynomial time (PPT) adversary $\mathcal{A}$ to potentially control all communications in the network via access to a set of oracles as defined below. We consider a game in which the adversary asks queries to oracles, and the oracles answer back to the adversary. Oracle queries model attacks which an adversary may use in the real system. We consider the following types of queries for ID-based AKA protocol. Let $i \in \{U, V\}$.

- $\mathsf{Extract}(ID)$: This query allows $\mathcal{A}$ to get the long-term secret key of ID for $ID \notin \mathcal{ID}$.

- Execute(U, V): This query models passive attacks, where $\mathcal{A}$ eavesdrops an execution of the protocol. $\mathcal{A}$ gets back the complete transcripts of an honest execution between U and V.
- Send(Π_i^s, M): This query is used to send a message M to instance Π_i^s. When Π_i^s receives M, it responds according to the ID-based AKA protocol. $\mathcal{A}$ may use this query to perform active attacks by modifying and inserting the messages of the protocol. Impersonation attacks and man-in-the-middle attacks are also possible using this query.
- Reveal(Π_i^s): This query models *known key* attacks (or Denning-Sacco attacks) in the real system. $\mathcal{A}$ is given the session key for instance Π_i^s.
- Corrupt(ID_i): This query models exposure of the long-term secret key held by ID_i. The proposed ID-based AKA protocol satisfies half forward secrecy and hence $\mathcal{A}$ is allowed to ask Corrupt(Π_U^s), but not to ask Corrupt(Π_V^s). However $\mathcal{A}$ cannot control the behavior of U directly (of course, once $\mathcal{A}$ has asked a query Corrupt(Π_U^s), $\mathcal{A}$ may impersonate U in subsequent Send queries).
- Test(Π_i^s): This query is used to define the advantage of $\mathcal{A}$. When $\mathcal{A}$ asks this query to an instance Π_i^s for $i \in \{U, V\}$, a random bit b is chosen; if $b = 1$ then the session key is returned. Otherwise a random string is returned. $\mathcal{A}$ is allowed to make a single Test query, at any time during the game.

In the model we consider two types of adversaries according to their attack types. The attack types are simulated by the queries issued by an adversary. A *passive adversary* is allowed to issue Execute, Reveal, Corrupt, and Test queries, while an *active adversary* is additionally allowed to issue Send and Extract queries. Even though Execute query can be simulated using Send queries repeatedly, we use Execute query for more concrete analysis.

2.2 Security Notions

Freshness. An oracle Π_i^s is said *fresh* (or holds a *fresh* key K) if:
- Π_i^s has accepted a session key $K \neq$ NULL and neither Π_i^s nor its partner has been asked for a Reveal query,
- No Corrupt query has been asked before a query of the form Send(Π_i^s,*) or Send(Π_j^t,*), where Π_j^t is Π_i^s's partner.

Definitions of Security. We define the security of the protocol by following game between the adversary $\mathcal{A}$ and an infinite set of oracles Π_i^s for $ID_i \in \mathcal{ID}$ and $s \in \mathbb{N}$.

1. A long-term key is assigned to each user through the initialization phase related to the security parameter.
2. Run adversary $\mathcal{A}$ who may ask some queries and get back the answers from the corresponding oracles.

3. At some stage during the execution a Test query is asked by the adversary to a *fresh* oracle. The adversary may continue to ask other queries, eventually outputs its guess b' for the bit b involved in the Test query and terminates.

In this game, the advantage of the adversary $\mathcal{A}$ is measured by the ability distinguishing a session key from a random value, i.e., its ability guessing b. We define Succ to be the event that $\mathcal{A}$ correctly guesses the bit b used by the Test oracle in answering the query. The advantage of an adversary $\mathcal{A}$ in attacking protocol P is defined as $\mathsf{Adv}_{\mathcal{A},P}(k) = \left|2 \cdot Pr[\mathsf{Succ}] - 1\right|$.

We say that a protocol P is a *secure ID-based authenticated key agreement* scheme if the following two properties are satisfied:

- Correctness: in the presence of an adversary, a partner oracle accepts the same key.
- Indistinguishability: for every PPT adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A},P}(k)$ is negligible.

Authentication. In this paper, we focus on AKA with implicit authentication; a key agreement protocol is said to provide implicit key authentication if a participant is assured that no other participants except its intended partner can possibly learn the value of a particular secret key. Note that the property of implicit key authentication does not necessarily mean that the partner has actually obtained the key.

3 The Bilinear Maps and Some Problems

In this section, we review bilinear maps and some assumptions related to our protocol. Let $\mathbb{G}_1$ be a cyclic additive group of prime order q and $\mathbb{G}_2$ be a cyclic multiplicative group of same order q. We assume that the discrete logarithm problems (DLP) in both $\mathbb{G}_1$ and $\mathbb{G}_2$ are intractable. We call $e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$ an *admissible bilinear* map if it satisfies the following properties:

1. Bilinearity: $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q^*$.
2. Non-degenerancy: There exists $P \in \mathbb{G}_1$ such that $e(P, P) \neq 1$.
3. Computability: There exists an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in \mathbb{G}_1$.

The modified Weil and Tate pairings in elliptic curve are examples of the admissible bilinear maps.

Computational Diffie-Hellman (CDH) problem: The CDH problem is to compute abP when given P, aP and bP for some $a, b \in \mathbb{Z}_q^*$.

Inverse Computational Diffie-Hellman (ICDH) problem: The ICDH problem is to compute $a^{-1}P$ when given P and aP for some $a \in \mathbb{Z}_q^*$.

Modified Inverse Computational Diffie-Hellman (mICDH) problem: The mICDH problem is to compute $(a + b)^{-1}P$ when given b, P, aP and $(a + b)P$ for some $a, b \in \mathbb{Z}_q^*$.

Bilinear Diffie-Hellman (BDH) problem: The BDH problem is to compute $e(P,P)^{abc}$ when given P, aP, bP and cP for some $a, b, c \in \mathbb{Z}_q^*$.

Bilinear Inverse Diffie-Hellman (BIDH) problem: The BIDH problem is to compute $e(P,P)^{\frac{1}{a}c}$ when given P, aP and cP for some $a, c \in \mathbb{Z}_q^*$.

Modified Bilinear Inverse Diffie-Hellman (mBIDH) problem: The mBIDH problem is to compute $e(P,P)^{\frac{1}{a+b}c}$ when given b, P, aP and cP for some $a, b, c \in \mathbb{Z}_q^*$.

We can easily show that the CDH, ICDH and mICDH problems are polynomial time equivalent.

Theorem 1. *The BDH, BIDH and mBIDH problems are polynomial time equivalent.*

Proof. The BDH and BIDH problems are polynomial time equivalent [23]. Hence we only show that the BIDH and mBIDH problems are polynomial time equivalent.

- [BIDH ⇒ mBIDH] Given inputs b, P, aP and cP to mBIDH, set inputs P, $a'P$ and cP to BIDH by computing $a'P = aP + bP$. BIDH outputs $e(P,P)^{\frac{1}{a'}c} = e(P,P)^{\frac{1}{a+b}c}$, which is the output of mBIDH.
- [mBIDH ⇒ BIDH] Given inputs P, aP and cP to BIDH, set inputs b, P, $a'P$ and cP to mBIDH by computing $a'P = aP - bP$. mBIDH outputs $e(P,P)^{\frac{1}{a'+b}c} = e(P,P)^{\frac{1}{a}c}$, which is the output of BIDH.

□

We assume that the CDH, BDH and BIDH problems are intractable. That is, there is no polynomial time algorithm solving the CDH, BDH and BIDH problems with non-negligible probability. As noted in [2], gap Diffie-Hellman (GDH) parameter generators satisfying the GDH assumption are believed to be constructed from the Weil and Tate pairings associated with super-singular elliptic curves or abelian varieties.

To prove the security of the proposed protocol, we review the k-CAA (Collusion Attack Algorithm with k traitor) problem and define k-mBIDH (modified BIDH with k values) problem. Actually, the k-mBIDH problem is a bilinear variant of the k-CAA problem.

Definition 1. *The k-CAA [17] problem is to compute* $\frac{1}{s+h}P$ *for some* $h \in \mathbb{Z}_q^*$ *when given*

$$P, sP, h_1, h_2 ..., h_k \in \mathbb{Z}_q^*, \frac{1}{s+h_1}P, \frac{1}{s+h_2}P, ..., \frac{1}{s+h_k}P.$$

Definition 2. *The k-mBIDH is to compute* $e(P,P)^{\frac{1}{s+h}t}$ *when given*

$$P, sP, tP, h, h_1, h_2 ..., h_k \in \mathbb{Z}_q^*, \frac{1}{s+h_1}P, \frac{1}{s+h_2}P, ..., \frac{1}{s+h_k}P.$$

More formally, the advantage of $\mathcal{A}$ *is defined to be:*

$$\left| \Pr\left[\mathcal{A}\left(\begin{array}{c} P, sP, tP, h, h_1, ..., h_k, \\ \frac{1}{s+h_1}P, \frac{1}{s+h_2}P, ..., \frac{1}{s+h_k}P \end{array} \right) = e(P,P)^{\frac{1}{s+h}t} \;\middle|\; \begin{array}{c} s, t, h, h_1 ..., h_k \leftarrow \mathbb{Z}_q^*; \\ P \leftarrow \mathbb{G}_1;\ h \neq h_i \end{array} \right] \right|.$$

4 Proposed Protocol and Security Analysis

In this section, we propose an ID-based authenticated key agreement protocol between mobile *Client* $\mathcal{U}$ and *Server* $\mathcal{V}$ using authenticryption scheme [15] and prove its security in the random oracle model. We denote this protocol by ID-AKA.

4.1 ID-based Authenticated Key Agreement Protocol

In the following description, groups $\mathbb{G}_1$, $\mathbb{G}_2$ and a bilinear map e are generated by a GDH generator in Section 3. We assume that the mICDH problem in $\mathbb{G}_1$ is intractable. The ID-AKA protocol is using the trusted key generation center (KGC), which generates a secret key for a given public identity. In our protocol, the security of secret keys is based on the intractability of the mICDH problem. Fig. 1 describes the proposed ID-AKA protocol.

Setup. KGC chooses a random $s \in \mathbb{Z}_q^*$ and a generator P of $\mathbb{G}_1$, and computes $P_{pub} = sP$. It also chooses four cryptographic hash functions $H : \{0,1\}^* \longrightarrow \mathbb{Z}_q^*$, $H_1 : \mathbb{G}_2 \longrightarrow \mathbb{Z}_q^*$, $H_2 : \{0,1\}^* \longrightarrow \{0,1\}^t$ and $H_3 : \{0,1\}^* \longrightarrow \{0,1\}^k$ where t is a secret parameter and k is the bit length of a session key. Then KGC keeps s secret as the *master secret key* and publishes system parameters $\mathsf{params} = \{e, \mathbb{G}_1, \mathbb{G}_2, q, P, P_{pub}, H, H_1, H_2, H_3\}$.

Extract. When client $\mathcal{U}$ with identity ID_U wishes to obtain a secret key, KGC computes $q_u = H(ID_U)$ and $g = e(P, P)$. KGC then computes secret key $S_U = \frac{1}{s+q_u}P$ and returns (g, S_U) to $\mathcal{U}$ via secure channel. Similarly, given identity ID_V of server $\mathcal{V}$, KGC computes secret key $S_V = \frac{1}{s+q_v}P$, where $q_v = H(ID_V)$. KGC then returns S_V to $\mathcal{V}$ via secure channel.

When $\mathcal{U}$ and $\mathcal{V}$ want to establish a session key, they execute the following protocol as shown in Figure 1:

1. The client $\mathcal{U}$ computes $q_v = H(ID_V)$ and picks a random number a in Z_q^*. $\mathcal{U}$ computes $t_u = g^a$ and $Q_V = P_{pub} + q_v P$. Also, $\mathcal{U}$ computes $h = H_1(t_u)$, $X = aQ_V$ and $Y = (a+h)S_U$. $\mathcal{U}$ then sends $\langle ID_U, (X, Y)\rangle$ to $\mathcal{V}$.
2. $\mathcal{V}$ computes $q_u = H(ID_U)$ and $Q_U = P_{pub} + q_u P$. $\mathcal{V}$ then computes $t_u = e(X, S_V)$ and $c = e(Y, Q_U)$ by using received messages and its secret key S_V. Also, $\mathcal{V}$ computes $h = H_1(t_u)$ and checks if $c = t_u g^h$. If it is not true, $\mathcal{V}$ outputs FAIL and aborts. Otherwise, $\mathcal{V}$ picks a random number t_v in Z_q^*, and computes authentication message $z = H_2(t_u||t_v||\mathcal{T}||\mathcal{ID})$ where $\mathcal{T} = X||Y$. Finally, $\mathcal{V}$ sends z and t_v to $\mathcal{U}$, and computes the session key $sk = H_3(t_u||t_v||z||\mathcal{T}||\mathcal{ID})$.
3. $\mathcal{U}$ computes $z' = H_2(t_u||t_v||\mathcal{T}||\mathcal{ID})$ by using t_u and t_v, and checks if $z' = z$. If it is not true, $\mathcal{U}$ outputs FAIL and aborts. Otherwise, $\mathcal{U}$ computes the session key $sk = H_3(t_u||t_v||z||\mathcal{T}||\mathcal{ID})$.

Client $\mathcal{U}$		Server $\mathcal{V}$
$P, P_{pub}, [ID_U, S_U]$		$P, P_{pub}, [ID_V, S_V]$
$g = e(P, P)$		
$a \stackrel{R}{\leftarrow} Z_q^*, q_v \leftarrow H(ID_V)$		
$t_u = g^a,\ Q_V = P_{pub} + q_v P$		
$h \leftarrow H_1(t_u),\ X = aQ_V$		
$Y = (a+h)S_U$	$\xrightarrow{ID_U,\ (X,Y)}$	
		$q_u \leftarrow H(ID_U)$
		$Q_U = P_{pub} + q_u P$
		$t_u = e(X, S_V)$
		$c = e(Y, Q_U),\ h \leftarrow H_1(t_u)$
		$c \stackrel{?}{=} t_u g^h$
		$t_v \stackrel{R}{\leftarrow} Z_q^*$
	$\xleftarrow{z,\ t_v}$	$z = H_2(t_u \|\| t_v \|\| \mathcal{T} \|\| \mathcal{ID})$
$z' = H_2(t_u \|\| t_v \|\| \mathcal{T} \|\| \mathcal{ID})$		
$z' \stackrel{?}{=} z$		$sk = H_3(t_u \|\| t_v \|\| z \|\| \mathcal{T} \|\| \mathcal{ID})$
$sk = H_3(t_u \|\| t_v \|\| z \|\| \mathcal{T} \|\| \mathcal{ID})$		

Fig. 1. ID-AKA protocol

In the above protocol, $\mathcal{U}$ can pre-compute t_u and Y, and hence needs one hash operation, one point addition and two scalar multiplications on-line. Furthermore, hash function H is not Map-To-Point operation which requires many computations.

The ID-AKA protocol provides half forward secrecy. That is, if the secret key of the server is compromised, then all the session keys are revealed using transcripts. However, the exposure of client's secret key is not helpful to get the information about previous session keys. We note that in general, it is practical to assume that low-power devices held by clients are vulnerable to attacks, while a server is more secure.

4.2 Security Analysis

For the following security analysis we define Forger as a PPT forger which forges client's transcripts in the ID-AKA by using ID_U and ID_V. We first show that given ID_U and ID_U, forging client's transcripts is intractable.

Lemma 1. *Assume that the hash functions H and H_1 are random oracles. Suppose there exists a Forger $\mathcal{A}$ for given ID_U and ID_V with running time t_0 and advantage ε_0. Suppose $\mathcal{A}$ makes at most q_H, q_{H_1}, q_S and q_E queries to the H, H_1, Send and Extract, respectively. If $\varepsilon_0 \geq 10q_{H_1}^2(q_S+1)(q_S+q_H)/q$, then there exists an attacker $\mathcal{B}$ that solves the k-CAA problem within expected time $t_1 \leq 120686 q_{H_1} t_0/\varepsilon_0$.*

Proof. $\mathcal{B}$ is given an instance (P, sP, q_1, ..., q_k, $\frac{1}{q_1+s}P$,..., $\frac{1}{q_k+s}P$) of the k-CAA problem , where $k \geq q_H, q_S$. Its goal is to computes $\frac{1}{q_0+s}P$ for some q_0.

Then $\mathcal{B}$ runs $\mathcal{A}$ as a subroutine and simulates its attack environment. First, $\mathcal{B}$ generates GDH parameters $\langle e, \mathbb{G}_1, \mathbb{G}_2 \rangle$ and sets public system parameters $\langle e, \mathbb{G}_1, \mathbb{G}_2, P, P_{pub}, g \rangle$ by computing $P_{pub} = sP, g = e(P, P)$. $\mathcal{B}$ gives public parameters to $\mathcal{A}$.

Without loss of generality, we assume that for any ID, $\mathcal{A}$ queries H, H_1, Send and Extract at most once, and Send and Extract queries are preceded by an H-hash query. To avoid collision and consistently respond to these queries, $\mathcal{B}$ maintains lists L_H and L_{H_1}. The lists are initially empty. $\mathcal{B}$ interacts with $\mathcal{A}$ as follows:

H-query. When $\mathcal{A}$ makes an H-query ID_i, if $ID_i = ID_U$, then $\mathcal{B}$ returns q_0. Otherwise, $\mathcal{B}$ adds $\langle ID_i, q_i \rangle$ to L_H and returns q_i.

H_1-query. When $\mathcal{A}$ makes an H_1-query m, then $\mathcal{B}$ returns a random number h and adds (m, h) to L_{H_1}.

Send-query. When $\mathcal{A}$ makes a $\mathsf{Send}(\Pi_i)$ query, if $ID_i = ID_U$, then $\mathcal{B}$ chooses random numbers a, h and computes $X = a(sP + q_0 P)$, $Y = hP$. $\mathcal{B}$ returns $ID_U, (X, Y)$. Otherwise, $\mathcal{B}$ finds $\langle ID_i, q_i \rangle$ in L_H. $\mathcal{B}$ also chooses random numbers a, h, and computes $X = a(sP + q_i P)$, $Y = (a+h)\frac{1}{q_i+s}P$. $\mathcal{B}$ returns $ID_U, (X, Y)$. If $\mathcal{A}$ makes a $\mathsf{Send}(ID_U)$ query, then $\mathcal{B}$ can not construct a valid transcript (X, Y) and returns X and a random value Y. The simulation works correctly since $\mathcal{A}$ can not distinguish whether any transcript (X, Y) is valid or not unless he knows a long-term secret key S_V of the server.

Extract-query. When $\mathcal{A}$ asks an Extract query on $ID_i \notin \{ID_V, ID_U\}$, $\mathcal{B}$ finds $\langle ID_i, q_i \rangle$ in L_H. Then $\mathcal{B}$ returns $\frac{1}{q_i+s}P$ to $\mathcal{A}$.

Eventually, $\mathcal{A}$ outputs a new valid message tuple $\langle ID_U, (X, Y) \rangle$, without accessing any oracle expect H_1. By replaying $\mathcal{B}$ with the same tape but different choices of H_1, as done in the *forking lemma* [20], $\mathcal{A}$ outputs two valid message tuples $\langle ID_U, (X = aQ_V, Y = (a+h)\frac{1}{s+q_0}P) \rangle$ and $\langle ID_U, (X = aQ_V, Y' = (a+h')\frac{1}{s+q_0}P) \rangle$ such that $h \neq h'$. $\mathcal{B}$ can compute $(Y - Y')/(h - h') = \frac{1}{s+q_0}P$ and outputs it.

The probability that $\mathcal{B}$ correctly guesses h and h' is $1/q_{H_1}^2$. Also, the total running time t_1 of $\mathcal{B}$ is equal to the running time of the *forking lemma* which is bounded by $120686 q_{H_1} t_0 / \varepsilon_0$, as desired.

□

Theorem 2. *Assume that the hash functions are random oracles. Suppose there exists an ID-AKA adversary $\mathcal{A}$ with running time t and given ID_U and ID_V. Then the ID-AKA is a secure AKA protocol providing half forward secrecy under the hardness of the k-mBIDH and k-CAA problems. Concretely,*

$$\mathrm{Adv}^{\mathrm{AKA-hfs}}_{\mathsf{ID\text{-}AKA},\mathcal{A}}(t) \leq \frac{1}{2}\, q_S q_{H_1} \mathrm{Adv}^{\mathrm{k-mBIDH}}_{\mathbb{G}_1,\mathbb{G}_2,\mathrm{e}}(t) + \mathrm{Adv}^{\mathrm{Forge}}(t)$$

where $\mathrm{Adv}^{\mathrm{Forge}}(t)$ is the maximum advantage of any Forger running in time t, and an adversary $\mathcal{A}$ makes q_S send queries, q_C corrupt queries and q_{H_i} hash queries to H_i.

Proof. Let $\mathcal{A}$ be an active adversary that gets advantage in attacking ID-AKA. The adversary $\mathcal{A}$ can get the advantage by following cases:

Case 1. Forging authentication transcripts, namely impersonating a client.
Case 2. Breaking the protocol without altering transcripts.

First, to compute the advantage of $\mathcal{A}$ from Case 1, we use $\mathcal{A}$ to construct a Forger $\mathcal{F}$ that generates a valid message pair $\langle ID, (X, Y)\rangle$ as follows: $\mathcal{F}$ honestly generates all other public and secret keys for the system. $\mathcal{F}$ simulates the oracle queries of $\mathcal{A}$ in the natural way. Let Forge denote the event that $\mathcal{A}$ generates a new and valid message pair $\langle ID, (X, Y)\rangle$. Then the success probability of $\mathcal{F}$ satisfies $\Pr_{\mathcal{A}}[\mathsf{Forge}] \leq \mathrm{Adv}_{\mathcal{F}}^{\mathrm{Forge}}(t) \leq \mathrm{Adv}^{\mathrm{Forge}}(t)$. By Lemma 1, $\Pr_{\mathcal{A}}[\mathsf{Forge}]$ is negligible.

Next, we compute the upper bound of the advantage from Case 2. To get any information of a session key sk in the random oracle model, adversary $\mathcal{A}$ has to ask $\langle t_u||t_v||z||\mathcal{T}||\mathcal{ID}\rangle$ to the hash oracle H_3. Therefore, $\mathcal{A}$ has to compute the secret value t_u. Then we can construct an attacker $\mathcal{B}$ that breaks the k-mBIDH problem using $\mathcal{A}$ with non-negligible probability. $\mathcal{B}$ is given an instance of the k-mBIDH problem $(e, \mathbb{G}_1, \mathbb{G}_2, P, sP, tP, q_0, ..., q_k, \frac{1}{q_1+s}P, ..., \frac{1}{q_k+s}P)$, where $k \geq q_{H_0}, q_S$. Its goal is to compute $e(P, P)^{\frac{1}{q_0+s}t}$. $\mathcal{B}$ runs $\mathcal{A}$ as a subroutine and simulates its attack environment. $\mathcal{B}$ sets the public system parameters $\langle e, \mathbb{G}_1, \mathbb{G}_2, P, P_{pub}, g\rangle$ by letting $P_{pub} = sP, g = e(P, P)$. $\mathcal{B}$ gives public parameters to $\mathcal{A}$. To make use of the advantage of $\mathcal{A}$, $\mathcal{B}$ guesses α such that $\mathcal{A}$ asks its Test query in the α-th session.

Without loss of generality, we assume that $\mathcal{A}$ does not ask queries on a same message more than once, and Send and Corrupt (or Extract) queries are preceded by an H-hash query. To avoid collision and consistently respond to these queries, $\mathcal{B}$ maintains lists L_H and L_{H_1}. The lists are initially empty. $\mathcal{B}$ simulates the oracle queries of $\mathcal{A}$ as follows:

H-query. When $\mathcal{A}$ makes an H-query $ID_i = ID_V$, then $\mathcal{B}$ returns q_0. Otherwise, $\mathcal{B}$ adds $\langle ID_i, q_i\rangle$ to L_H and returns q_i.

$\mathbf{H_1}$-query. If $\mathcal{A}$ makes an H_1-query m such that a record (m, h) appears in L_{H_1}, then $\mathcal{B}$ returns h. Otherwise, $\mathcal{B}$ returns a random number h and adds (m, h) to L_{H_1}.

$\mathbf{H_2}$-query. When $\mathcal{A}$ makes an H_2-query $\langle t_u||t_v||X||Y||ID_U||ID_V\rangle$, $\mathcal{B}$ finds the tuple of the form $\langle (t_v, X, Y, ID_U, ID_V), z\rangle$ in L_{H_2}. Then $\mathcal{B}$ returns z to $\mathcal{A}$.

$\mathbf{H_3}$-query. When $\mathcal{A}$ makes an H_3-query m, then $\mathcal{B}$ returns a random number.

Send-query. For convenience we classifies Send queries into two types: client-to-server and server-to-client types, denoted by Send_S and Send_C, respectively.

- When $\mathcal{A}$ makes a $\mathsf{Send}_S(\Pi_U^s, Start)$ query, if the query is asked in the α-th session, $\mathcal{B}$ chooses a random number r and computes $X = tP, Y = rP$ and returns ID_U and (X, Y). Otherwise, $\mathcal{B}$ finds $\langle ID_U, q_u\rangle$ in L_H. $\mathcal{B}$ also chooses random numbers a, h, and computes $Q_V = sP + q_0P$, $X = aQ_V$, $Y = (a + h)\frac{1}{q_u+s}P$, $t_u = e(P, P)^a$. Then $\mathcal{B}$ adds (t_u, h) to L_{H_1} and returns ID_U and (X, Y).

– When $\mathcal{A}$ makes a $\mathsf{Send}_C(\Pi_V^j, (ID_U, X, Y))$ query, $\mathcal{B}$ chooses random numbers z, t_v, and returns z, t_v and adds $\langle (t_v, X, Y, ID_U, ID_V), z\rangle$ to L_{H_2}.

Execute-query. When $\mathcal{A}$ asks an $\mathsf{Execute}(U, V)$ query, then $\mathcal{B}$ returns the transcript $\langle (ID_U, X, Y),\ (z, t_v)\rangle$ using the above simulation of Send queries.

Extract-query. When $\mathcal{A}$ asks an $\mathsf{Extract}$ query on $ID_i \notin \{ID_V, ID_U\}$, $\mathcal{B}$ finds $\langle ID_i, q_i\rangle$ in L_H. Then $\mathcal{B}$ returns $\frac{1}{q_i+s}P$ to $\mathcal{A}$.

Corruct-query. When $\mathcal{A}$ asks a $\mathsf{Corrupt}$ query on ID_U, $\mathcal{B}$ finds $\langle ID_U, q_u\rangle$ in L_H. Then $\mathcal{B}$ returns $\frac{1}{q_u+s}P$ to $\mathcal{A}$.

Reveal-query. When $\mathcal{A}$ makes a Reveal query, then $\mathcal{B}$ returns a random number.

Test-query. When $\mathcal{A}$ makes a Test query, if the query is not asked in the α-th session, $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ flips a coin b; if $b = 1$, $\mathcal{B}$ returns a session key, else $\mathcal{B}$ returns a random number.

The success probability of $\mathcal{B}$ depends on the event that $\mathcal{B}$ correctly guesses α and $\mathcal{A}$ asks a secret value $t_u = e(P, P)^{\frac{1}{q_0+s}t}$ to H_1 hash oracle. In the above simulation, the probability that $\mathcal{B}$ correctly guess α is $1/q_S$. If the advantage of the correct guess of $\mathcal{A}$ is ε, then $\mathcal{A}$ issues a query for $H_1(t_u)$ with advantage 2ε. Thus, if $\mathcal{B}$ correctly guesses α, the secret value t_u appears in the list L_{H_1} with probability at least 2ε. Therefore, $\mathcal{B}$ solves the k-mBIDH problem with probability at least $2\varepsilon/q_S q_{H_1}$ as required. Finally, we have the result that the advantage of $\mathcal{A}$ conditioned on the event **Case 2** is bounded by $\frac{1}{2}q_S q_{H_1}\mathrm{Adv}^{\mathrm{k-mBIDH}}_{\mathbb{G}_1,\mathbb{G}_2,\mathrm{e}}(t)$. Hence we have

$$\mathrm{Adv}^{\mathrm{AKA-hfs}}_{\mathit{ID\text{-}AKA},\mathcal{A}}(t) \leq \tfrac{1}{2}q_S q_{H_1}\mathrm{Adv}^{\mathrm{k-mBIDH}}_{\mathbb{G}_1,\mathbb{G}_2,\mathrm{e}}(t) + \mathrm{Adv}^{\mathrm{Forge}}_{\sigma}(t).$$

□

References

1. E. Bresson, O. Chevassut, A. Essiari and D. Pointcheval, *Mutual Athentication and Group Key Agreement for Low-Power Mobile Devices*, In the 5th IEEE International Conference on Mobile and Wireless Communications Networks, 2003.
2. D. Boneh and M. Franklin, *Identity-based encryption from the Weil pairing*, Proc. of Crypto '01, LNCS 2139, pp.213-229, Springer-Verlag, 2001.
3. P. S. L. M. Barreto, H. Y. Kim, B. Lynn and M. Scott, *Efficient algorithms for pairing-based cryptosystems.*, Proc. of Crypto '02, LNCS 2442, pp. 354-368, Springer-Verlag, 2002.
4. P. S. L. M. Barreto, B. Lynn and M. Scott, *Efficient implementation of pairing-based cryptosystems.*, Journal of Cryptology, pp. 321-334, 2004.
5. M. Bellare, P. Rogaway, *Entity authentication and key distribution*, Proc. of Crypto '93, pp.232-249.
6. M. Bellare, P. Rogaway, *Provably-Secure Session Key Distribution : The Three Party Case*, Proc. of STOC '95, pp. 57-66.
7. E. Bresson, O. Chevassut and D. Pointcheval, *Provably Authenticated Group Diffie-Hellman Key Exchange-The Dynamic Case*, Proc. of Asiacrypt '02, LNCS 2248, pp.290-309, Springer-Verlag, 2001.

8. K. Y. Choi, J. Y. Hwang, D. H. Lee, *Efficient ID-based Group Key Agreement with Bilinear Maps*, Proc. of PKC '04, LNCS 2947, pp.130-144, Springer-Verlag, 2004.
9. W. Diffie and M. Hellman, *New Directions in Cryptography*, IEEE Transactions on Information Theory 22(6), pp.644-654, 1976.
10. S. D. Galbraith, K. Harrison and D. Soldera, *Implementing the Tate pairing*, Proc. of ANTS'02, LNCS 2369, pp.324-337, Springer-Verlag, 2002.
11. Q. Huang, J. Cukier, H. Kobayashi. B. Liu and J. Zhang, *Fast Authenticated Key Establishment Protocols for Self-Organizing Sensor Networks*, In Proc. of WSNA'03, Copyright 2003 ACM.
12. J. Y. Hwang, S. M. Lee and D. H. Lee, *Scalable key exchange transformation : from two-party to group*, Electronics Letters, Vol. 40, No. 12, Jun. 2004.
13. H. J. Kim, S. M. Lee and D. H. Lee, *Constant-Round Authenticated Gourp Key Exchange for Dynamic Groups*, Proc. of Asiacrypt 2004, LNCS 3329, pp.245-259, Springer-Verlag, 2004.
14. J. Katz and M. Yung, *Scalable Protocols for Authenticated Group Key Exchange*, Proc. of Crypto 2003, LNCS 2729, pp.110-125, Springer-Verlag, 2003.
15. N. McCullagh and P. S. L. M. Barreto, *Effcient and Foward-Secure Identity-Based Signcryption*, Cryptology ePrint Archive, Report 2004/117. http://eprint.iacr.org/2004/117/.
16. N. McCullagh and P. S. L. M. Barreto, *A New Two-Party Identity-Based Authenticated Key Agreement*, Proc. of CT-RSA'05, LNCS 3376, pp.262-274 , Springer-Verlag, 2005.
17. S. Mitsunari, R. Sakai and M. Kasahara, *A new traitor tracing*, Proc. of IEICE Trans. Vol. E85-A, No.2, pp.481-484, 2002.
18. J. Nam, S. Kim, D. Won, *Attacks on Bresson-Chevassut-Essiari-Pointcheval's Group Key Agreement Scheme for Low-Power Mobile Devices*, Proc. of IEEE Communications Letters, 2005.
19. D. Nalla, K. C. Reddy, *ID-based tripartite Authenticated Key Agreement Protocols from pairings*, Cryptology ePrint Archive, Report 2003/004. http://eprint.iacr.org/2003/004/.
20. D. Pointcheval and J. Stern, *Security arguments for digital signatures and blind signatures*, J. of Cryptology, Vol. 13, pp.361-396, 2000.
21. N.P.Smart, *An Identity based authenticated Key Agreement protocol based on the Weil pairing*, Electronics Letters, vol. 38 (13): 630–632, June 2002.
22. A. Shamir, *Identity Based Cryptosystems and Signature Schemes*, Proc. of Crypto 1984, LNCS 0196, Springer-Verlag, 1984.
23. F. Zhang, R. Safavi-Naini and W. Susilo, *An Efficient Signature Scheme from Bilinear Pairings and Its Applications*, , Proc. of PKC '04, LNCS 2947, pp.277-290, Springer-Verlag, 2004.

On the Security of Two Key-Updating Signature Schemes*

Xingyang Guo[1,2], Quan Zhang[1], and Chaojing Tang[1]

[1] School of Electronic Science and Engineering, National University of Defense Technology, P.R. China 410073
{xyguo,quangzhang}@nudt.edu.cn cjtang@263.net
[2] School of Telecommunication Engineering, Air force Engineering University, P.R. China 710077
xingyang_guo@hotmail.com

Abstract. In ICICS 2004, Gonzalez-Deleito, Markowitch and Dall'Olio proposed an efficient strong key-insulated signature scheme. They claimed that it is $(N-1, N)$-key-insulated, i.e., the compromise of the secret keys for even $N-1$ time periods does not expose the secret keys for the remaining time period. But in this paper, we demonstrate an attack and show that an adversary armed with the signing keys for any two time periods can derive the signing key for any of the remaining time periods with high probability. In a second attack, the adversary may be able to forge signatures for many remaining time periods without computing the corresponding signing keys. A variant forward-secure signature scheme was also presented in ICICS 2004 and claimed more robust than traditional forward-secure signature schemes. But we find that the scheme has two similar weaknesses. We give the way how to repair the two schemes in this paper.

1 Introduction

Many cryptographic techniques today, whether only available in the literature or actually used in practice, are believed to be quite secure. Several, in fact, can be proven secure under very reasonable assumptions. In a vast majority of solutions, however, security guarantees last only as long as secrets remain unrevealed. If a secret is revealed, security is often compromised not only for subsequent uses of the secret, but also for prior ones. For example, if a secret signing key becomes known to an adversary, one cannot trust any signature produced with that key and the signer is forced to revoke its public key. Unfortunately, this does not always suffice as even valid signatures having been produced before the revealment become invalid, unless a time-stamping authority has attested that they were produced before the corresponding public key was revoked.

* This work is supported by the National Natural Science Foundation of China under Grant No. 60472032.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 506–517, 2005.

Getting rid of the revocation and time-stamping mechanisms in order to simplify key management is an active research topic. In recent years, some key-updating approaches are presented to limit the damages arising when secret keys are exposed.

An approach to this problem is the forward-secure cryptosystem. In the forward-secure model[2,3], the lifetime of secret keys is divided into discrete time periods. At the beginning of each period, users compute a new secret key by applying a public one-way function to the secret key used during the previous time period, while public keys remain unchanged. An adversary compromising the secret signing key at a given time period will be unable to produce signatures for previous periods, but will still be able to sign messages during the current and future time periods. Unlike classical schemes, the validity of previously produced signatures is therefore assured, but public keys have to be revoked. Several recent investigations in forward-secure signature scheme are given in [1,7,9].

The notion of key-insulated cryptosystems, which was introduced by Dodis et al.[4], generalises the concept of forward-secure cryptography. In this model, lifetime of secret keys is also divided into discrete periods and, as in previous models, signatures are supposed to be generated by relatively insecure devices. However, the secret associated with a public key is here shared between the user and a physically secure device. At the beginning of each time period the user obtains from the device a partial secret key for the current time period. By combining this partial secret key with the secret key for the previous period, the user derives the secret key for the current time period. Exposure of the secret key at a given period will not enable an adversary to derive secret keys for the remaining time periods. More precisely, in a (t, N)-key-insulated scheme the compromise of the secret keys for up to t time periods does not expose the secret key for any of the remaining $N - t$ time periods. Therefore, public keys do not need to be revoked unless t periods have been exposed. Strong key-insulated schemes[5] guarantee that the physically secure device (or an attacker compromising the partial secrets held by this device) is unable to derive the secret key for any time period. This is an extremely important property if the physically secure device serves several different users.

Itkis and Reyzin[8] introduced the notion of intrusion-resilient signatures, which strengthens the one of key-insulation by allowing an arbitrary number of non-simultaneous compromises of both the user and the device, while preserving security of prior and future time periods.

In ICICS'04, Gonzalez-Deleito, Markowitch and Dall'Olio[6] proposed a new strong $(N-1, N)$-key-insulated signature scheme (GMD scheme, from now on). The scheme is claimed more efficient than previous proposals and that has the property that becomes forward-secure when all the existing secrets at a given time period are compromised. They also presented a variant forward-secure signature scheme claimed to be more robust than traditional forward-secure signature schemes. In this paper, we demonstrate two attacks on each of the two schemes, respectively. We try to repair the two schemes.

The remaining of the paper is organized as follows: section 2 briefly describes some definitions of key-updating signature schemes. Section 3 reviews and analyzes the GMD key-insulated signature scheme, section 4 reviews and analyzes the GMD forward-secure signature scheme, section 5 gives the suggestion to repair the two schemes, section 6 repairs the forward-secure signature scheme as an example on the suggestion, section 7 concludes.

2 Definitions of Key-Updating Signature Schemes

The following definitions of key-insulated signature schemes are based on the definitions given by Gonzalez-Deleito et al.[6].

A *key-insulated signature scheme* is a 5-tuple of polynomial time algorithms (KGen, UpdD, UpdU, Sig, Ver) such that:

- KGen, the key generation algorithm, is a probabilistic algorithm taking as input one or several security parameters sp and (possibly) the total number of periods N, and returning a public key PK, a master secret key MSK and a user's initial secret key USK_0.

- UpdD, the physically secure device key-update algorithm, is a (possibly) probabilistic algorithm which takes as input the index i of the next time period, the master secret key MSK and (possibly) the total number of periods N, and returns a partial secret key PSK_i for the i-th time period.

- UpdU, the user key-update algorithm, is a deterministic algorithm which takes as input the index i of the next time period, the user's secret key USK_{i-1} for the current time period and the partial secret key PSK_i. It returns the user's secret key USK_i and the secret signing key SK_i for the next time period. The secret keys USK_i and SK_i are stored in the relatively insecure device.

- Sig, the signing algorithm, is a probabilistic algorithm which takes as input the index i of the current time period, a message M and the signing key SK_i for the time period i; it returns a pair $< i, s >$ composed of the time period i and a signature s.

- Ver, the verification algorithm, is a deterministic algorithm which takes as input a message M, a candidate signature $< i, s >$ on M, the public key PK and (possibly) the total number of periods N; it returns **true** if $< i, s >$ is a valid signature on M for period i, and **false** otherwise.

The life cycle of keys in a key-insulated scheme can be described as follows. A user begins by running the KGen algorithm, obtaining a public key PK, as well as the corresponding master secret key MSK and user's initial secret key USK_0. The public key PK is certified through a certification authority (CA) and made

publicly available, while MSK is stored on the physically secure device and USK_0 is stored by the user himself. For each time period i, $1 \leq i \leq N$, the user is now able to obtain a partial secret key PSK_i by asking the device to run the UpdD algorithm. By executing UpdU, the user transforms, with the help of USK_{i-1}, the partial secret key received from the device into a signing key SK_i for time period i which may be used to sign messages during this time period. Furthermore, the user updates USK_{i-1} to USK_i and erases USK_{i-1} and SK_{i-1}.

For simplicity, we suppose that an adversary may

- ask for signatures on adaptively chosen messages for adaptively chosen time periods;
- expose the insecure signing device for up to t adaptively chosen time periods.

If the adversary cannot succeed to forge a valid signature $< i, s >$ on a message M for which he never requested a signature for time period i and he never exposed the insecure device at this time period, the (t, N) key-insulated signature scheme is *secure*.

The forward-secure signature scheme can be regarded as the simplified version of the key-insulated signature scheme. In traditional forward-secure signature schemes, there are no physically secure devices and UpdD phases and the only secure time periods are that prior to the compromised time period.

3 The GMD Key-Insulated Signature Scheme and Its Security

3.1 Review of the GMD Key-Insulated Signature Scheme

KeyGen(k, l) k and l are two security parameters. Let $n = pq$ be a k-bit modulus, where $p = 2p' + 1$ and $q = 2q' + 1$ are safe prime numbers such that p' and q' are also safe primes. Let v be an $(l + 1)$-bit prime number. And let h be a one-way hash function $h : \{0,1\} \rightarrow \{0,1\}^l$. The user randomly chooses $s, t, u \in Z_n^*$, such that $s^2 \neq s^{2^{8+1}} \bmod n$, $t^2 \neq t^{2^{8+1}} \bmod n$ and $u^2 \neq u^{2^{8+1}} \bmod n$. The public key PK is composed of $PK_1 = s^{-v} \bmod n$, $PK_2 = t^{-v} \bmod n$ and $PK_3 = u^{-v} \bmod n$. The master secret key MSK is composed of $MSK_1 = s^2 \bmod n$ and $MSK_2 = t^2 \bmod n$, and the user's initial secret key is $USK_0 = u^2 \bmod n$.

UpdD(i,N,MSK) The physically secure device computes the partial secret key for the i-th time period as follows:

$$PSK_i = (MSK_1)^{2^i} \cdot (MSK_2)^{2^{N-i}} \bmod n = s^{2^{i+1}} \cdot t^{2^{N+1-i}} \bmod n.$$

UpdU(i, USK_{i-1}, PSK_i) The user computes the user's secret key for the time period i

$$USK_i = (USK_{i-1})^2 \bmod n = u^{2^{i+1}} \bmod n$$

and the corresponding signing key

$$SK_i = PSK_i \cdot USK_i \bmod n = s^{2^{i+1}} \cdot t^{2^{N+1-i}} \cdot u^{2^{i+1}} \bmod n.$$

$Sig_{SKi}(i, M)$ In order to sign a message M during the time period i, the user randomly chooses a value $x \in Z_n^*$, computes $y = x^v \bmod n$, $d = h(i, M, y)$ and $D = x \cdot (SK_i)^d \bmod n$. The signature on M for the time period i is (i, d, D).

$Ver_{PK}(M, (i, d, D), N)$ For verifying whether (i, d, D) is a valid signature on M for the time period i, an entity computes

$$h(i, M, D^v \cdot ((PK_1)^{2^{i+1}} \cdot (PK_2)^{2^{N+1-i}} \cdot (PK_3)^{2^{i+1}})^d \bmod n)$$

and accepts the signature only if the result is equal to d.

In paper[6], the authors claimed that it is a $(N-1, N)$-key-insulated signature scheme. It is also be claimed that the scheme can be used for signature delegation. In this context, a user grants to another user the right to sign messages on his behalf during a limited amount of time. It is suggested that this kind of delegation can be simply achieved by giving to this second user a signing key for the corresponding time period. By using a time-stamping service, the delegated user will only produce valid signatures during the designated time period. We demonstrate two attacks on the scheme assuming that an adversary has obtained signing keys SK_i and SK_j $(i < j)$ for time periods i and j. Therefore, an adversary compromising a user for two time periods or two delegated users can carry out the two attacks.

3.2 The First Attack on the Scheme

With the signing keys SK_i and SK_j, the adversary can carry out the attack to derive the signing key for a remaining time period r as follows:

step 1 Computes

$$\begin{aligned}
K_{su} &= SK_j^{2^{j-i}} \cdot SK_i^{-1} \\
&= ((su)^{2^{j+1}} \cdot t^{2^{N+1-j}})^{2^{j-i}} \cdot ((su)^{2^{i+1}} \cdot t^{2^{N+1-i}})^{-1} \\
&= (su)^{2^{2j-i+1}} \cdot t^{2^{N+1-i}} \cdot (su)^{-2^{i+1}} \cdot t^{-2^{N+1-i}} \\
&= (su)^{2^{2j-i+1}-2^{i+1}} \\
&= (su)^{2^{i+1} \cdot (2^{2j-2i}-1)} \bmod n \\
K_t &= SK_i^{2^{j-i}} \cdot SK_j^{-1} \\
&= ((su)^{2^{i+1}} \cdot t^{2^{N+1-i}})^{2^{j-i}} \cdot ((su)^{2^{j+1}} \cdot t^{2^{N+1-j}})^{-1} \\
&= (su)^{2^{j+1}} \cdot t^{2^{N+1+j-2i}} \cdot (su)^{-2^{j+1}} \cdot t^{-2^{N+1-j}} \\
&= t^{2^{N+1+j-2i}-2^{N+1-j}} \\
&= t^{2^{N-j+1} \cdot (2^{2j-2i}-1)} \bmod n
\end{aligned}$$

step 2 Computes

$$\begin{aligned}
(su)^v &= (PK_1 \cdot PK_3)^{-1} \bmod n \\
t^v &= (PK_2)^{-1} \bmod n
\end{aligned}$$

step 3 Using the extended Euclidean algorithm, the attacker finds integers a_{su} and b_{su} such that

$$a_{su} \cdot v + b_{su} \cdot (2^{i+1} \cdot (2^{2j-2i} - 1)) = 1$$

If so, we have

$$su = (su)^{a_{su} \cdot v + b_{su} \cdot (2^{i+1} \cdot (2^{2j-2i}-1))} \bmod n = ((su)^v)^{a_{su}} \cdot K_{su}^{b_{su}} \bmod n$$

step 4 Using the extended Euclidean algorithm, the attacker finds integers a_t and b_t such that

$$a_t \cdot v + b_t \cdot 2^{N-j+1} \cdot (2^{2j-2i} - 1) = 1$$

If so, we have

$$t = t^{a_t \cdot v + b_t \cdot 2^{N-j+1} \cdot (2^{2j-2i}-1)} \bmod n = (t^v)^{a_t} \cdot K_t^{b_t} \bmod n$$

step 5 For the time period r, the adversary computes the corresponding signing key

$$SK_r = (su)^{2^{r+1}} \cdot t^{2^{N+1-r}} \bmod n$$

It should be noticed that the computation of a_{su} and b_{su} in step 3 or a_t and b_t in step 4 will not always succeed. The computations will succeed when $\gcd(v, 2^{i+1} \cdot (2^{2j-2i} - 1)) = 1$ and $\gcd(v, 2^{N-j+1} \cdot (2^{2j-2i} - 1)) = 1$. Since v is a prime number, the only case when $\gcd(v, 2^{i+1} \cdot (2^{2j-2i} - 1)) \neq 1$ or $\gcd(v, 2^{N-j+1} \cdot (2^{2j-2i} - 1)) \neq 1$ is that v is a factor of $2^{2j-2i} - 1$. When $v > 2^{2j-2i} - 1$, obviously, v is not a factor of $2^{2j-2i} - 1$. Else, since v is a $(l+1)$-bit big prime number, for example $l = 128$ (MD5 for h), if v is randomly selected in the **KeyGen** phase, the probability that v exactly divides $2^{2j-2i} - 1$ is negligible. The user is not able to intend to select a $(l+1)$-bit prime number v that divides $2^{2j-2i} - 1$ for all different i and j. Therefore the adversary will succeed to carry out the attack with high probability.

3.3 The Second Attack on the Scheme

In this attack, we assume that the adversary also has the signing keys SK_i and SK_j. The adversary may be able to forge signatures for some remaining time periods on an arbitrary message m with non-negligible probabilities, without computing the corresponding signing keys. If $2j - 2i < l$, to forge a signature for a time period r ($r \neq i, j$), the adversary carries out as follows:

step 1 Computes

$$K_{su} = SK_j^{2^{j-i}} \cdot SK_i^{-1} \bmod n$$
$$K_t = SK_i^{2^{j-i}} \cdot SK_j^{-1} \bmod n$$

step 2 Randomly chooses a value $x \in Z_n^*$, sets $w = 0$ and $y = 1$, computes $x_v = x^v \bmod n$.

step 3 Computes $w = w + 1$ and $y = y \cdot x_v \bmod n$. If $y = 1$, turns back to step 2, else computes $d = h(i, m, y)$.

step 4 Checks whether d can be exactly divided by

$$\begin{cases} (2^{2j-2i} - 1) \cdot 2^{i-r} & case \quad r < i; \\ (2^{2j-2i} - 1) & case \quad i < r < j; \\ (2^{2j-2i} - 1) \cdot 2^{r-j} & case \quad r > j; \end{cases}$$

If not the adversary turns back to step 3, else continues.

step 5 Computes D as

$$\begin{cases} x^w \cdot K_{su}^{(d \ div \ (2^{2j-2i}-1)\cdot 2^{i-r})} \cdot K_t^{(d \ div \ (2^{2j-2i}-1))\cdot 2^{j-r}} & mod \ n \quad case \quad r < i; \\ x^w \cdot K_{su}^{(d \ div \ (2^{2j-2i}-1))\cdot 2^{r-i}} \cdot K_t^{(d \ div \ (2^{2j-2i}-1))\cdot 2^{j-r}} & mod \ n \quad case \quad i < r < j; \\ x^w \cdot K_{su}^{(d \ div \ (2^{2j-2i}-1))\cdot 2^{r-i}} \cdot K_t^{(d \ div \ (2^{2j-2i}-1)\cdot 2^{r-j})} & mod \ n \quad case \quad r > j; \end{cases}$$

The signature on m for the time period r is (r, d, D).

Correctness For simplicity we only demonstrate the validity of the signature in case $r < i$.

$$\begin{aligned} & h(r, m, D^v \cdot ((PK_1)^{2^{r+1}} \cdot (PK_2)^{2^{N+1-r}} \cdot (PK_3)^{2^{r+1}})^d \bmod n) \\ &= h(r, m, D^v \cdot (PK_1 \cdot PK_3)^{d\cdot 2^{r+1}} \cdot (PK_2)^{d\cdot 2^{N+1-r}} \bmod n) \\ &= h(r, m, x^{wv} \cdot K_{su}^{(d \ div \ (2^{2j-2i}-1)\cdot 2^{i-r})\cdot v} \cdot K_t^{(d \ div \ (2^{2j-2i}-1))\cdot 2^{j-r}\cdot v} \\ &\qquad \cdot (su)^{-v\cdot d\cdot 2^{r+1}} \cdot t^{-v\cdot d\cdot 2^{N+1-r}} \bmod n) \\ &= h(r, m, x^{wv} \cdot (su)^{2^{i+1}\cdot (2^{2j-2i}-1)(d \ div \ (2^{2j-2i}-1)\cdot 2^{i-r})\cdot v} \\ &\quad \cdot t^{2^{N-j+1}\cdot (2^{2j-2i}-1)(d \ div \ (2^{2j-2i}-1))\cdot 2^{j-r}\cdot v} \cdot (su)^{-v\cdot d\cdot 2^{r+1}} \cdot t^{-v\cdot d\cdot 2^{N+1-r}} \bmod n) \\ &= h(r, m, x^{wv} \cdot (su)^{2^{r+1}\cdot d\cdot v} \cdot t^{2^{N-r+1}\cdot d\cdot v} \cdot (su)^{-v\cdot d\cdot 2^{r+1}} \cdot t^{-v\cdot d\cdot 2^{N+1-r}} \bmod n) \\ &= h(r, m, x^{wv} \bmod n) \\ &= d. \end{aligned}$$

Efficiency of the attack We take the case $r < i$ for example. The efficiency of the attack mostly depends on finding d that can be divided by $(2^{2j-2i} - 1) \cdot 2^{i-r}$ by trail and error. Since h is a hash function, its output distribution will be uniform in $[0, 2^l)$. Hence the success probability of finding a proper d is $\frac{1}{(2^{2j-2i}-1)\cdot 2^{i-r}}$ with 1 try and $1-(1-\frac{1}{(2^{2j-2i}-1)\cdot 2^{i-r}})^n$ with n tries. The attack is most efficient in the case $j = i + 1$ and $r = i - 1$ while the success probability of finding a proper d is more than 99% with 26 tries. The attack will be more inefficient when i is more less than j, r more less than i when $r < i$ and j more less than r when $r > j$. But as long as $2j - 2i \ll l$, we think that there always are some

time periods that can be attacked with non-negligible probabilities in polynomial time.

In the GMD key-insulated signature scheme, if an adversary obtains more singing keys or compromises a user at more time periods, he will carry out some variant attacks. The above two attacks show that the scheme is only equivalent to a $(1, N)$-key-insulated signature scheme.

4 The GMD Forward-Secure Signature Scheme and Its Security

4.1 Review of the GMD Forward-Secure Signature Scheme

KeyGen(k, l) n, v and h are selected as same as that in the key-insulated scheme. The user randomly chooses $t, u \in Z_n^*$, such that $t^2 \neq t^{2^{8+1}} \bmod n$ and $u^2 \neq u^{2^{8+1}} \bmod n$. The public key PK is composed of $PK_1 = t^{-v} \bmod n$ and $PK_2 = u^{-v} \bmod n$. The master secret key is $MSK = t^2 \bmod n$ and the user's initial secret key is $USK_0 = u^2 \bmod n$.

UpdD(i,N,MSK) The physically secure device computes the partial secret key

$$PSK_i = (MSK)^{2^{N-i}} \bmod n = t^{2^{N+1-i}} \bmod n.$$

UpdU(i, USK_{i-1}, PSK_i) The user computes the user's secret key for the time period i

$$USK_i = (USK_{i-1})^2 \bmod n = u^{2^{i+1}} \bmod n$$

and the corresponding signing key

$$SK_i = PSK_i \cdot USK_i \bmod n = t^{2^{N+1-i}} \cdot u^{2^{i+1}} \bmod n.$$

$Sig_{SKi}(i, M)$ In order to sign a message M during the time period i, the user randomly chooses a value $x \in Z_n^*$, computes $y = x^v \bmod n$, $d = h(i, M, y)$ and $D = x \cdot (SK_i)^d \bmod n$. The signature on M for the time period i is (i, d, D).

$Ver_{PK}(M, (i, d, D), N)$ For verifying whether (i, d, D) is a valid signature on M for the time period i, an entity computes

$$h(i, M, D^v \cdot ((PK_1)^{2^{N+1-i}} \cdot (PK_2)^{2^{i+1}})^d \bmod n)$$

and accepts the signature only if the result is equal to d.

4.2 The First Attack on the Scheme

We demonstrate an attack quite similar to the first attack on the key-insulated signature scheme, on the assumption that an adversary compromises a user at one time period i and gets SK_i and USK_i. The adversary can derive the signing key for any time period r while $r < i$.

step 1 Computes $PSK_i = SK_i \cdot USK_i^{-1} = t^{2^{N+1-i}} \bmod n$.

step 2 Computes

$$u^v = PK_2^{-1} \bmod n$$
$$t^v = PK_1^{-1} \bmod n$$

step 3 Using the extended Euclidean algorithm, the attacker finds integers a_u and b_u such that

$$a_u \cdot v + b_u \cdot (2^{i+1}) = 1$$

If so, we have

$$u = u^{a_u \cdot v + b_u \cdot 2^{i+1}} \bmod n = (u^v)^{a_u} \cdot USK_i^{b_u} \bmod n$$

step 4 Using the extended Euclidean algorithm, the attacker finds integers a_t and b_t such that

$$a_t \cdot v + b_t \cdot (2^{N+1-i}) = 1$$

If so, we have

$$t = t^{a_t \cdot v + b_t \cdot 2^{N+1-i}} \bmod n = (t^v)^{a_t} \cdot PSK_i^{b_t} \bmod n$$

step 5 For the time period r, the adversary computes the corresponding signing key

$$SK_r = t^{2^{N+1-r}} \cdot u^{2^{r+1}} \bmod n$$

Since $\gcd(v, 2^{N+1-i}) = 1$ and $\gcd(v, 2^{i+1}) = 1$, the adversary will succeed to compute a_u,b_u in step 3 and a_t,b_t in step 4.

4.3 The Second Attack on the Scheme

This attack is quite similar to the second attack on the key-insulated signature scheme. The adversary with SK_i and USK_i can forge signatures for some time periods, for example r ($r < i$), on an arbitrary message m, with probabilities that are not negligible. The adversary carries out as follows:

step 1 Computes $PSK_i = SK_i \cdot USK_i^{-1} \bmod n$.

step 2 Randomly chooses a value $x \in Z_n^*$, sets $w = 0$ and $y = 1$, computes $x_v = x^v \bmod n$.

step 3 Computes $w = w + 1$ and $y = y \cdot x_v \bmod n$. If $y = 1$, turns back to step 2, else computes $d = h(i, m, y)$.

step 4 Checks whether d can be exactly divided by 2^{i-r}. If not the adversary turns back to step 3, else continues.

step 5 Computes $D = x^w \cdot (PSK_i^{2^{i-r}})^d \cdot (USK_i)^{(d \ div \ 2^{i-r})} \bmod n$. The signature on m for the time period r is (r, d, D).

Correctness (r, d, D) is valid since

$$\begin{aligned}
& h(r, m, D^v \cdot ((PK_1)^{2^{N+1-r}} \cdot (PK_2)^{2^{r+1}})^d \bmod n) \\
= {} & h(r, m, x^{wv} \cdot (PSK_i^{2^{i-r}})^{d \cdot v} \cdot (USK_i)^{(d \ div \ 2^{i-r}) \cdot v} \\
& \qquad \cdot ((t^{-v})^{2^{N+1-r}} \cdot (u^{-v})^{2^{r+1}})^d \bmod n) \\
= {} & h(r, m, x^{wv} \cdot (t^{2^{N+1-i} \cdot 2^{i-r}})^{d \cdot v} \cdot u^{2^{i+1} \cdot (d \ div \ 2^{i-r}) \cdot v} \\
& \qquad \cdot ((t^{-v})^{2^{N+1-r}})^d \cdot ((u^{-v})^{2^{r+1}})^d \bmod n) \\
= {} & h(r, m, x^{wv} \cdot t^{2^{N+1-r} \cdot d \cdot v} \cdot u^{2^{r+1} \cdot d \cdot v} \cdot t^{-v \cdot 2^{N+1-r} \cdot d} \cdot u^{-v \cdot 2^{r+1} \cdot d} \bmod n) \\
= {} & h(r, m, x^{wv} \bmod n) \\
= {} & d.
\end{aligned}$$

Efficiency of the Attack The efficiency of the attack mostly depends on finding d that can be divided by 2^{i-r}. The success probability of finding a proper d is $\frac{1}{2^{i-r}}$ with 1 try and $1 - (1 - \frac{1}{2^{i-r}})^n$ with n tries. The attack on time period $i-1$ is most efficient while the success probability of finding a proper d is more than 99% with 7 tries. The attack will be more inefficient when r is more less than i, but obviously many time periods less than i can be attacked with non-negligible probabilities in polynomial time. The attack cannot work when $i - r \geq l$.

With the above two attacks, an adversary with secret keys for one time period can compute signing keys or forge signatures for some previous time periods. Therefore the scheme is **not** a forward-secure signature scheme.

5 Suggestion to Repair the Two Schemes

For the first attacks, the natural countermeasure is to select a proper scheme parameter v so that the public keys can not be used by the adversary to derive secret keys to be protected. A natural idea to resist the second attacks, since the efficiency of our attacks depends on finding d that can be divided by some certain values, is to make these values larger than d. The second attacks show another weaknesses that the power step in the key-updating phases is too small. Therefore the improved schemes may update secret keys with the power step 2^l or 2^{-l} rather than 2 or 2^{-1}.

In the next section of this report, we try to take the forward-secure signature scheme for example to repair on this suggestion. The key-insulated signature scheme may be repaired in a similar way while v should be selected more carefully.

6 An Example: Attempt to Repair the Forward-Secure Signature Scheme

KeyGen(k, l) In this phase, all parameters are generated as same as that in the original scheme except that $v = 2^{l \cdot (N+2)}$, $MSK = t^{2^l} \bmod n$ and $USK_0 = u^{2^l}$

mod n. Notice that t and u should be chosen so that the secret keys take a large number of values before cycling.

UpdD(i,N,MSK) The physically secure device computes the partial secret key for the i-th time period as follows:

$$PSK_i = MSK^{2^{l\cdot(N-i)}} \bmod n = t^{2^{l\cdot(N+1-i)}} \bmod n.$$

UpdU(i, USK_{i-1}, PSK_i) The user computes the user's secret key for the time period i

$$USK_i = (USK_{i-1})^{2^l} \bmod n = u^{2^{l\cdot(i+1)}} \bmod n$$

and the corresponding signing key

$$SK_i = PSK_i \cdot USK_i \bmod n = t^{2^{l\cdot(N+1-i)}} \cdot u^{2^{l\cdot(i+1)}} \bmod n.$$

$Sig_{SKi}(i, M)$ In order to sign a message M during the time period i, the user randomly chooses a value $x \in Z_n^*$, computes $y = x^v \bmod n$, $d = h(i, M, y)$. The user computes $D = x \cdot (SK_i)^d \bmod n$. The signature on M for the time period i is (i, d, D).

$Ver_{PK}(M, (i, d, D), N)$ For verifying whether (i, d, D) is a valid signature on M for the time period i, an entity computes

$$h(i, M, D^v \cdot (PK_1)^{2^{l\cdot(N+1-i)}} \cdot (PK_2)^{2^{l\cdot(i+1)}})^d \bmod n)$$

and accepts the signature only if the result is equal to d.

An adversary cannot get any additional information from the public keys because he can compute the public keys from the secret keys he possessed for any one time period. Since d is the output of the hash function h and a l-bit integer, it cannot be exactly divided by the values, such as $2^{l\cdot(i-r)}$, that may be conscribed by the adversary in the second attack and its variations. The cost of the security improvement is the lost of some efficiency because of some larger power computation in verification.

7 Conclusions

In this paper, we presented security analysis of Gonzalez-Deleito et al.'s key-insulated signature scheme and forward-secure scheme proposed in[6]. By successfully identifying four attacks, we demonstrated that their schemes are insecure. We give the suggestion to repair the two schemes. In fact, how to design a secure and efficient key-insulated signature scheme is still a hot topic.

References

1. M. Abdalla and L. Reyzin. A new forward-secure digital signature scheme. In Proceedings of Advances in Cryptology-ASIACRYPT 2000, volume 1976 of Lecture Notes in Computer Science, pages 116-129. Springer-Verlag, Dec. 2000.
2. R. Anderson. Two remarks on public key cryptology. Invited lecture, 4th Conference on Computer and Communications Security. ACM, 1997.
3. M. Bellare and S. K. Miner. A forward-secure digital signature scheme. In Proceedings of Advances in Cryptology-CRYPTO 99, volume 1666 of Lecture Notes in Computer Science, pages 431-448. Springer-Verlag, Aug. 1999.
4. Y. Dodis, J. Katz, S. Xu, and M. Yung. Key-insulated public key cryptosystems. In Proceedings of Advances in Cryptology-EUROCRYPT 2002, volume 2332 of Lecture Notes in Computer Science, pages 65-82. Springer-Verlag, Apr. 2002.
5. Y. Dodis, J. Katz, S. Xu, and M. Yung. Strong key-insulated signature schemes. In Proceedings of the 6th International Workshop on Practice and Theory in Public Key Cryptography (PKC 2003), volume 2567 of Lecture Notes in Computer Science, pages 130-144. Springer-Verlag, Jan. 2003.
6. N. Gonzalez-Deleito, O. Markowitch and E. Dall'Olio. A New Key-Insulated Signature Scheme. In Proceedings of the 6th International Conference on Information and Communications Security (ICICS 2004), volume 3269 of Lecture Notes in Computer Science, pages 465-479. Springer-Verlag, October 2004.
7. G. Itkis and L. Reyzin. Forward-secure signatures with optimal signing and verifying. In Proceedings of Advances in Cryptology-CRYPTO 2001, volume 2139 of Lecture Notes in Computer Science, pages 332-354. Springer-Verlag, Aug. 2001.
8. G. Itkis and L. Reyzin. SiBIR: Signer-base intrusion-resilient signatures. In Proceedings of Advances in Cryptology-CRYPTO 2002, volume 2442 of Lecture Notes in Computer Science, pages 499-514. Springer-Verlag, Aug. 2002.
9. A. Kozlov and L. Reyzin. Forward-secure signatures with fast key update. In Proceedings of the 3rd International Conference on Security in Communication Networks (SCN 2002), volume 2576 of Lecture Notes in Computer Science, pages 241-256. Springer-Verlag, Sept. 2002.

Building Secure Tame-like Multivariate Public-Key Cryptosystems: The New TTS

Bo-Yin Yang[1] and Jiun-Ming Chen[2]

[1] Dept. of Mathematics, Tamkang University, Tamsui, Taiwan*
by@moscito.org
[2] Chinese Data Security, Inc., & Nat'l Taiwan U., Taipei
jmchen@ntu.edu.tw

Abstract. Multivariate public-key cryptosystems (sometimes polynomial-based PKC's or just multivariates) handle polynomials of many variables over relatively small fields instead of elements of a large ring or group. The "tame-like" or "sparse" class of multivariates are distinguished by the relatively few terms that they have per central equation. We explain how they differ from the "big-field" type of multivariates, represented by derivatives of C^* and HFE, how they are better, and give basic security criteria for them. The last is shown to be satisfied by efficient schemes called "Enhanced TTS" which is built on a combination of the Oil-and-Vinegar and Triangular ideas. Their security levels are estimated. In this process we summarize and in some cases, improve *rank-based attacks*, which seek linear combinations of certain matrices at given ranks. These attacks are responsible for breaking many prior multivariate designs.

1 Introduction: Multivariate and Tame-like PKC's

Multivariate public-key cryptosystems (sometimes just multivariates[3]) operate on long vectors over small fields, in contrast to the huge rings and groups of better-known schemes. A typical multivariate PKC over the base field K has a public map comprising three portions. In the notations of [3, 35], we write it as $V = \phi_3 \circ \phi_2 \circ \phi_1 : K^n \to K^m$. The maps $\phi_1 : \mathbf{w} \mapsto \mathbf{x} = \mathsf{M}_1\mathbf{w} + \mathbf{c}_1$ and $\phi_3 : \mathbf{y} \mapsto \mathbf{z} = \mathsf{M}_3\mathbf{y} + \mathbf{c}_3$ are affine in K^n and K^m respectively and usually invertible. We call ϕ_2 the *central map* and the equations giving each y_j in the x_i's the *central equations*. *The security of the scheme is then based on the NP-hardness [15] in solving a large system of quadratics and difficulty in decomposing V into the components ϕ_i.* The speed of the public map and the size of the keys depend only on m and n. The speed of the private map depends on how fast a preimage for $\phi_2 : \mathbf{x} \mapsto \mathbf{y}$ can be obtained, and key generation on the complexity of ϕ_2. A good quick reference on various multivariates can be found in [33].

* Research supported in part by Taiwan's National Science Council via TWISC (Taiwan Information Security Center at NTUST) project and grant NSC93-M-2115-032-008

[3] also "polynomial-based" along with lattice-based NTU, which differs fundamentally.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 518–531, 2005.

Recently there has been renewed interest in multivariate PKC's, and we will

- Characterize tame-like PKC's, a subset of multivariates (Sec. 1). Show that they are efficient and possibly very useful in low-resource deployments.
- Review the security concerns of tame-like PKC's including linear-algebra related attacks (collectively, "rank attacks", Sec. 5), in some cases generalized and improved viz. Sec. 9, modern Gröbner Bases related methods, and others (Sec. 10).
- Give basic criteria for proper design of a tame-like multivariate scheme (Sec. 10). Build (Sec. 4) a scalable sequence of schemes satisfying these conditions using a combination of the triangular and oil-and-vinegar themes.

Note: old version at e-Print archive report 2004/061; full version to be up later.

2 Pros and Cons for Multivariates

For a long time, cryptologists were not very interested in multivariates because traditional PKC's are considered "good enough". The large keys of multivariates also causes problems in key storage, management, and generation for PKI setup and maintenance. Furthermore, the last two decades saw many proposed multivariates broken, so there is some general distrust of multivariates. But multivariate are getting another look because

1. The relative slowness of RSA does affect deployment (e.g., co-processors cost) and some environments are simply too real-time-oriented or resource-poor for RSA (i.e. lower-cost RFID). A multivariate-like structure may do better [14].
2. In some multivariate schemes, keys can be generated blockwise possible in real time on a smart card, which ameliorates the on-card storage problem.
3. Quantum computing may become reality in two decades, bringing a sea change.

The slowness of current progress [30] belies the lack of recent advances in factoring technique, but at CHES 2004, Dr. Issac Chuang reported on QC and estimated less than 2 decades to practicality. RSA and discrete-log based schemes will then be broken by Shor's Algorithm [28], but multivariates are more resistant[4]. Quantum physics can also accomplish a secure key exchange, but so far lacks the functionality of digital signatures. Thus alternative digital signature schemes are being sought.

3 Tame-like Multivariates

In one type of multivariates including the HFE [26] and $C^\star$ families [22, 27], ϕ_2 represents a function in a huge field. They are termed *big-field* or *two-field* [33], and generate keys via an interpolation-based procedure in $\sim n^6$ time [31].

[4] A QC attack with Grover's Algorithm [17] only halves the log-complexity [24].

Lower-powered systems, especially low-end embedded ones, needs to do better. **A multivariate is termed tame-like if its central equations average a small number (vs. $\sim n^2/2$ terms for a random quadratic) of terms** — say $\leq 2n$ each — and **can be inverted quickly**, e.g., faster than evaluating the public map. Since a tame-like map takes only $O(n^2)$ instead of $O(n^3)$ time to evaluate, key generation via interpolation would take at most $O(n^5)$ time. However, we can do even better than that:

Proposition 1. *Keys can be generated for a tame-like multivariate in time* $O(n^4)$.

Proof. Following Imai and Matsumoto [22], we divide the coefficients involved in each public key polynomial into linear, square, and crossterm portions thus:

$$z_k = \sum_i P_{ik} w_i + \sum_i Q_{ik} w_i^2 + \sum_{i<j} R_{ijk} w_i w_j = \sum_i w_i \left[P_{ik} + Q_{ik} w_i + \sum_{i<j} R_{ijk} w_j \right].$$

R_{ijk}, which comprise most of the public key, may be computed as follows (as in [35]):

$$R_{ijk} = \sum_{\ell=n-m}^{n-1} \left[(\mathsf{M}_3)_{k,(\ell-n+m)} \left(\sum_{p\, x_\alpha x_\beta \text{ in } y_\ell} p\, ((\mathsf{M}_1)_{\alpha i}(\mathsf{M}_1)_{\beta j} + (\mathsf{M}_1)_{\alpha j}(\mathsf{M}_1)_{\beta i}) \right) \right] \quad (1)$$

The second sum is over all cross-terms $p\, x_\alpha x_\beta$ in the central equation for y_ℓ. For every pair $i < j$, we can compute at once R_{ijk} for every k in $O(n^2)$ totalling $O(n^4)$. Similar computations for P_{ik} and Q_{ik} take even less time.

Therefore set-up times for a tame-like multivariate be two-orders-of-magnitudes faster than non-tame-like ones. On a low-cost smartcard, on-demand public-key generation from private info ($O(n^2)$ storage) can be done in real time (cf. Tab. 1).

4 Triangular Maps, Tame Maps and the TTS Family

The prototype of tame-like ϕ_2 is the *tame transformation* from algebraic geometry. With dimensions $m \geq n$ over the *base field* K, this is a polynomial map $\phi : K^n \to K^m$, taking $\mathbf{x}$ to $\mathbf{y}$ either affinely ($\mathbf{y} = \mathsf{M}\mathbf{x} + \mathbf{c}$) or in *de Jonquiere* form with $y_1 = x_1$; $y_j = x_j + q_j(x_1, \ldots, x_{j-1})$, $j = 2 \cdots n$;
$y_j = q_j(x_1, \ldots, x_n)$, $j = n+1 \cdots m$. If bijective, it is a *tame automorphism* over K, in which case obviously $m = n$.

On tame transformations, sometimes called *triangular maps*, is based the public-key encryption scheme TTM [23]. This concept was adapted and extended the concept [3] to include **all polynomial maps without a low degree explicit inverse for which an inverse can be found without solving anything higher than linear equations.** We will call term such maps *tame*, and

([3]) **TTS is defined as a multivariate DSS with a tame central map.** For example, with $n = 28$, $m = 20$, ϕ_2 :

$$y_k = x_k + a_k x_{k-8} x_{k-1} + b_k x_{k-7} x_{k-2} + c_k x_{k-6} x_{k-3} + d_k x_{k-5} x_{k-4},\ 8 \le k \le 26;$$
$$y_{27} = x_{27} + a_{27} x_{19} x_{26} + b_{27} x_{20} x_{25} + c_{27} x_{21} x_{24} + d_{27} x_{\mathbf{0}} x_{\mathbf{27}};$$

is tame since we can assign any $x_1, \ldots, x_7$ and $x_0 \neq -d_{27}^{-1}$ and find a preimage. We will illustrate with the multivariate signature scheme TTS/2′ that has this central map.

5 Rank-Based Attacks Against Tame-like Multivariates

Many tame-like PKC's were broken through finding linear combinations associated matrices at some given rank. There are three distinct types of these *rank-based attacks*:

The Rank (or Low Rank) Attack [21] seems well-known in other circles before introduced to cryptography by Shamir and Kipnis against HFE.

The Dual (or High) Rank Attack [5] likely first invented by Coppersmith *et al.* Goubin and Courtois somewhat simplified the procedures of the above two attacks against an instance of the encryption scheme TTM [23]. They further expanded their scope to all "TPM" (triangular-plus-minus) systems [16].

Oil-and-Vinegar attacks invented by Kipnis *et al* [19, 20] against OV/UOV schemes.

6 The Rank or Low Rank Attack

Let $q = |K|$, and r be the smallest rank in linear combinations of central equations, which without loss of generality we take to be the first central equation itself. Goubin and Courtois [16] outline how to break TPM in expected time $O(q^{\lceil \frac{m}{n} \rceil r} m^3)$:

1. Take $P = \sum_{i=1}^{m} \lambda_i H_i$, an undetermined linear combination of the symmetric matrices representing the homogeneous quadratic portions of the public keys. [16] did not mention this, but when char $K = 2$ the quadratic portion of z_i cannot be written as $\mathbf{w}^T Q_i \mathbf{w}$, with the matrices Q_i symmetric. However there is still a unique symmetric matrix that can represent z_i, namely $H_i = Q_i + Q_i^T$ [5].
   A quadratic $C_{ab} x_a x_b + C_{cd} x_c x_d + \cdots$ with all indices distinct will have a corresponding symmetric matrix with kernel $\{\mathbf{x} :\ 0 = x_a = x_b = x_c = x_d = \cdots\}$. We will call this the kernel of the quadratic and use the shorthand $\ker y_i$ (or $\ker_{\mathbf{x}} y_i$ to specify what space). With ℓ cross-terms with distinct indices, the rank of the matrix is 2ℓ. Hence $\ker_{\mathbf{x}} y_k = \{\mathbf{x} :\ x_{k-8} = \cdots = x_{k-1} = 0\}$ for TTS/2′.

2. Guess at a random k-tuple $(\mathbf{w}_1, \ldots, \mathbf{w}_k)$ of vectors in K^n, where $k = \lceil \frac{m}{n} \rceil$. Set $P\mathbf{w}_1 = \cdots = P\mathbf{w}_k = \mathbf{0}$ and solve for λ_i via Gaussian elimination. When this is uniquely solvable P is likely the quadratic part of y_1, the first central equation.
3. Assume the matrix corresponding to y_1 has a rank of r, then its kernel (the inverse image $H_1^{-1}(\mathbf{0})$) has dimension $n - r$, hence when we guess at $(\mathbf{w}_1, \ldots, \mathbf{w}_k)$ randomly, they have a probability of at least q^{-kr} to be all in $H_1^{-1}(\mathbf{0})$. This P is the quadratic portion of y_1 and the coefficients λ_i the row of M_3^{-1} (up to a factor).

The Rank Attack should be at its most effective against a signature scheme, as $k = 1$. Obviously, not all multivariates are TPM. However, if a central equation has too few terms then the above works. Further remarks are due in Sec. 9.

Proposition 2 (Time to Find a Vector in any Given Kernel). *Regardless of the form of ϕ_2, if one unique linear combination $H = \sum_{i=1}^{m} \alpha_i H_i$ has the minimum rank r then the algorithm above will always find a vector in* $\ker H$ *with an expected time of* $\approx q^{kr}\left(m^2(nk/2 - m/6) + mn^2k\right)$ *field multiplications.*

7 The Dual Rank or High Rank Attack

The Rank Attack finds a large kernel shared by a small subset of the space spanned by the matrices H_i. The converse, to find a small kernel shared by a many linear combinations of the H_i, may be called a Dual Rank attack or High Rank attack. It happens when a variable appears in too few central equations.

In Birational Permutation Schemes, the last central variables x_n appears the cross-terms of only one equation. This critical weakness [5] means we can find linear combinations $\sum_i \alpha_i z_i$ whose kernels share a non-empty intersection. Coppersmith, Stern, and Vaudenay [5] then construct an ascending chain of kernels in the matrix algebra over a ring without needing to search. In [16], a simpler version of the dual rank attack was run via searching, and we can describe this as follows:

Without loss of generality, let the fewest number of appearances of all variables in the cross-terms of the central equations be the last variable x_{n-1} appearing u times.

In TTS/2′, this is x_{27}, which only appears in y_{27}. So whenever $\alpha_{27} = 0$, the subspace $U = \{x_0 = \cdots = x_{26} = 0\} \subset \ker \sum_{i=8}^{27} \alpha_i Q_i$. (Here H_i and Q_i are as in Sec. 6.) If we denote by m_{ij} the (i, j)-entry of M_3, then almost every (H_i, H_j) pair has a linear combination with a kernel containing the same subset U. In general, with almost any $(u+1)$-subset picked from the H_i, a unique linear combination of these matrices has a kernel containing $U = \{\mathbf{x} : \; x_0 = \cdots = x_{n-2} = 0\}$. We try to find U.

1. Form an arbitrary linear combination $H = \sum_i \alpha_i H_i$. Find $V = \ker H$.
2. When $\dim V = 1$, set $(\sum_j \lambda_j H_j)V = \{\mathbf{0}\}$ and check if the solution set $\hat{V}$ of the (λ_i) form a subspace dimension $m - u$. Because a matrix in $K^{n \times n}$

can have at most n different eigenvalues, less than n/q of the time we would need to do this.

3. With probability q^{-u} we have $V = U$. The cost of one trial is bounded by one elimination plus possible testing, so total cost is $\left[mn^2 + \frac{n^3}{6} + \frac{n}{q}(m^3/3 + mn^2)\right] q^u$. We can do with a little more than $\left(un^2 + \frac{n^3}{6}\right) q^u$ field multiplications if we only consider linear combinations of $(u+1)$ of the matrices H_i, and are not too unlucky.

From this subspace, we can find bigger kernels. [5] does this through taking a sequence of derivatives. For TPM as for TTS/2', the next bigger kernel (which is $U' = \{x_0 = x_1 = \cdots = x_{25} = 0\}$) can be found by examing subspaces of V, which will get us U' with probability $1/q$. So for TTS/2′, the flaw is severe and cryptanalysis is swift.

8 Unbalanced Oil-and-Vinegar Attacks and a Simplification

An (Unbalanced) Oil-and-Vinegar attack [19, 21] on a multivariate takes place if we may partition the variables x_i into sets $\mathcal{O}$ and $\mathcal{V}$, such that there is no cross-term with both variable in $\mathcal{O}$. The two sets are called the *oil* and *vinegar* variables respectively.

Suppose a maximal set of vinegar variables is at least size v, then Kipnis *et al* find the oil subspace (the space spanned by the oil variables) by looking at certain linear combinations that become degenerate. The average time complexity is $q^{2v-n-1}(n-v)^4$.

TTS/2′ fits this description with $v = 14$ ($\mathcal{V}$ is the variables with even indices). An OV or UOV attack in essence let the attacker eliminate some variables. This often let the attacker get around whatever devices that defend against a rank attack. In [11], Ding and Yin cryptanalyze the instance of TTS given in [35] on such an oversight. They used a sequence of fairly intricate manuevers after the UOV stage. In this and certain other cases, we could make cryptanalysis using the UOV attack a little simpler, as below:

Proposition 3 (Unbalanced Oil-and-Vinegar with Guessing). *If a multivariate digital signature scheme with a public map $K^n \to K^m$ have minimal vinegar variable set size v, then a solution may be found in* $\max(q^{2v-n-1}(n-v)^4, q^{m+v-n}(n-v)^3/3)$ *multiplications, regardless of other structure.*

Proof. Follow the steps in [19] to distill the oil subspace. Now, if it were really an UOV scheme, we would be able to find a solution in time $(n-v)^3/3$ (i.e., time for one Gaussian elimination). However, this requires us to be able to guess at v variables. Since we can only fix $n-m$ variables and expect to find a solution, on average we rate $q^{v-(n-m)}$ random guesses during the the cryptanalysis.

9 More About Rank-Based Attacks

Rank-based attacks are important considerations against tame-like (and perhaps other) multivariates. The various authors already did a fine job of presenting the methods. One notable correction we would like to make is the estimate for dual-rank attacks in [16] (unquestioned by later works) is given as n^6q^u when it should be $\left(un^2 + \frac{n^3}{6}\right) q^u$ (field multiplications) as given in Sec. 7. It is easy to fall to any of these three attacks if one is careless, e.g., in the RSE(2)PKC and RSSE(2)PKC schemes of Kasahara-Sakai that falls ([32]) to an almost verbatim attack from [16]. These are generalizations of TPM that C. Wolf *et al* call Stepwise Triangular Schemes (STS). As discussed in [33], the basic STS constructions cannot be used alone. We may also surmise that to depend fundamentally on guessing can be a very bad idea for non-big-field multivariates.

There is one potential improvement to the Rank Attack that has not been mentioned by previous investigators. In Sec. 6 we assume y_1 to have the smallest rank r; other y_i and even many linear combinations of the y_i (hence the H_i) with different kernels can also share the same minimum rank r. For example, in TTS/2′, for non-zero α, the ranks of $y_i + \alpha y_{i+1}$ and $y_i + \alpha y_{i+2}$ are both 8. So is $y_i + \alpha y_{i+1} + \beta y_{i+2}$ if $\alpha^2 a_{i+1} b_{i+1} d_{i+1} = \beta(c_i a_{i+1} d_{i+2} + b_i d_{i+1} a_{i+2})$. That is at least 10,000 total combinations. We call this *interlinks*. When the largest kernels and equations interlink, the Rank Attack can be made faster by the *crawl* process below. Odds of finding a kernel vector in the [16] attack is then essentially multiplied by the number of distinct kernels.

Proposition 4 (Interlinked Kernels). *If there are c kernels of codimension r that interlink, then we can cryptanalyze in an expected $q^{kr}kmn(m+n)/c$ field multiplications.*

We take again as the example TTS/2′. For simplicity, let all coefficients be 1, then

$$\begin{aligned}
\ker y_8 &= \{\mathbf{x} : x_0 = x_1 = \cdots = x_7 = 0\};\\
\ker y_9 &= \{\mathbf{x} : x_1 = x_2 = \cdots = x_8 = 0\};\\
\ker y_{10} &= \{\mathbf{x} : x_2 = x_3 = \cdots = x_9 = 0\};\\
\ker(y_8 + \alpha y_9) &= \{\mathbf{x} : x_1 = x_3 = x_5 = x_7 = 0,\ x_0 : x_2 : x_4 : x_6 : x_8 = \alpha^4 : \alpha^3 : \alpha^2 : \alpha : 1\};\\
\ker(y_8 + \alpha y_{10}) &= \{\mathbf{x} : x_2 = x_3 = x_6 = x_7 = 0,\ x_0 : x_4 : x_8 = x_1 : x_5 : x_9 = \alpha^2 : \alpha : 1\};
\end{aligned}$$

With generic coefficients, there will be a three-term combination that has rank 8 exists (here it does not). Its kernel would be vectors $\mathbf{x}$ with $x_4 = x_5 = 0$ and $x_0 : x_2 : x_6 : x_8$ and $x_1 : x_3 : x_7 : x_9$ in fixed ratios. We now proceed along these steps:

1. Run the algorithm of Sec. 6 to find a kernel vector $\mathbf{u}$ and its associated quadratic $z = \sum_i \lambda_i z_i$ of rank 8. Verify $U = \ker z$ to be of codimension 8, and find a basis for U. Given any rank 8 kernel when $(m, n) = (20, 28)$, according to Prop. 2, we should need $256^8 \cdot \left[20^2 \cdot (28/2 - 20/6) + 20 \cdot 28^2\right] \approx 2^{78}$ field

multiplications (or $\approx 2^{71}$ 3DES units) to find a vector in that kernel. But kernels of the 10000+ rank-8 forms are mostly distinct, so we expect only $\sim 2^{65}$ multiplications. There being only 20 forms y_i and about 5000 forms $y_i + \alpha y_{i+1}$ (and almost as many $y_i + \alpha y_{i+2}$), the first vector yielding a codimension-8 kernel will likely come from a mixed form rather than from one of the y_i's, and we therefore need to isolate y_i's.

2. Repeat the same algorithm but we restrict test vectors $\mathbf{w}$ to U, and only accept a tested vector if it lies in more than one kernel, i.e., we solve $\sum_i \lambda_i H_i \mathbf{v} = 0$, finding a basis $(\hat{y}_i)_{i=1\cdots s}$ in quadratic forms, and keep $\mathbf{v}$ if the solution space is of dimension two or higher. Let this solution space be expressed in quadratic forms as $v \in \ker(\sum_{\ell=1}^{s} \alpha_\ell \hat{y}_\ell)$ for $s \geq 2$. We expect the dimension s to be 2 or 3. If we find two distinct sets of results ($\mathbf{v}$ and $(\hat{y}_i)$) in say 5000 tests, then we have just found a y_i for some $9 \leq i \leq 25$, and the results would match the forms $\mathrm{span}(y_i, y_{i\pm 1})$.
   If, as is normally the case, we find only one solution space for λ_i's, then that must be $\mathrm{span}(y_i, y_{i+1})$ or $\mathrm{span}(y_i, y_{i+1}, y_{i+2})$ depending on its dimension. As an example, assume that we initially hit a vector that lies in the kernel U of $y_8 + \alpha y_9$ and no other quadratic form. With probability 2^{-8} a random vector $\mathbf{v} \in U$ will lie in $\ker y_8 \cap \ker y_9 = \{\mathbf{x} : x_0 = x_1 = \cdots = x_8 = 0\}$. Ditto for any $z = y_i + \alpha y_{i+1}$.
   Similarly if $z = y_i + \alpha y_{i+2}$, or any three-term combination that has rank 8, the odds of finding a vector $\mathbf{v}$ in more than one kernel is 2^{-16}, and we would find $(\ker y_i) \cap (\ker y_{i+1}) \cap (\ker y_{i+2})$. *This step should take little time in this step, equivalent to trying* 2^{16} *random vectors* $\mathbf{w}$ *in Sec. 6.*
3. For all the materially different quadratic forms f_i that we locate we find the kernels U_i associated with them. There will be either 257 or $256^2 + 256 + 1 = 65793$ distinct linear combinations. Among the forms f_i we should have either two or three of the y_i's. Repeat the search in each U_i as above to find kernels corresponding to the y_i's. *Checking* 2^{12} *vectors from each of* $\sim 2^{16}$ *kernels* U_i *take* $< 2^{42}$ *multiplications.*
4. Say we have found y_9, since $y_9 = x_9 + a_9 x_1 x_8 + b_9 x_2 x_7 + c_9 x_3 x_6 + d_9 x_4 x_5$, we should be able to identify one linear combination of the w_i as x_9 and 8 others as $x_1, \ldots, x_8$. Indeed finding any y_i should give quickly all y_j and x_j where $j < i$. Incremental search will then locate all forms y_i and x_i, i.e. matrices M_1 and M_3.

In TTS/2′, a kernel vector should be found in $\approx 2^{63}$ multiplications ($\approx 2^{57}$ 3DES blocks[5]). Experiments on smaller analogues to TTS/2′ schemes was in reasonable accord with the cryptanalysis described above. There are other possibilities, viz.:

Proposition 5 (Accumulating Kernels). *With equations of rank 2 with sole cross-terms are* $x_i x_{j_1}, x_i x_{j_2}, \ldots, x_i x_{j_s}$*, then any vector with* $x_i = 0$ *becomes a kernel vector.*

[5] NESSIE [24] measures security in 3DES blocks and we count multiplications in GF(2^8). To calibrate, we use NESSIE's performance data [24], and compare against our runs. We find one 3DES block to be $\approx 2^6$ multiplications.

Remark: Such equations would effectively have a minrank of 1. A similar situation occurs in multi-term combinations. This implies that TTM is very hard to secure – there can only be rank-4+ equations and not too many interlinks.

10 Other Attacks and Security Criteria for a Multivariate

What non-rank-based attacks are there? There are no other generic attack aside from[6] Linearization-like Methods, i.e. XL [7] and Gröbner Bases Algorithms [12, 13]. There are also attacks tailored against specific schemes. The most important is Bilinear Relations [25], used against C^*. It only works if the central maps are rank-2 in some embedding field. Neither this nor any other specific attacks work against the tame-like systems that we will construct below (see [3]).

Proposition 6. *To build a tame-like Digital Signature Scheme needing a security of C:*

1. *If k linear combinations of central equations share a minimal rank r, then we need*
$$q^r \cdot \left(m^2(n/2 - m/6) + mn^2\right)/k \geq C. \tag{2}$$
Here usually $r = 2\ell$ where ℓ is the smallest number of cross-terms in an equation.
2. *If every central variable appears in at least u central equations, then*
$$q^u \left(un^2 + n^3/6\right) \geq C. \tag{3}$$
3. *Let v be the size of the smallest maximal set A of indices $0 \leq i < n$ such that every cross-term in the central map has at least one index in A, then we require*[7]
$$q^{2v-n-1}(n-v)^4 \geq C. \tag{4}$$
4. *Let $D_0 = D_0(m,k) := \min\{D : [t^D]\ \left((1-t)^{k-1}\ (1+t)^m\right)$, $T := \binom{m-k+D_0}{D_0}$, then ($c_0$, c_1, γ are constants, ω is the order of the equation-solver):*
$$\min_k q^k \cdot m^{\gamma_0} T^\omega (c_0 + c_1 \lg T) \geq C. \tag{5}$$
5. *There should not be any over-determined subsystems in the central equations.*[8]

The reader will need to refer to [1, 8, 34] to understand how Eq. 5 came about. The executive summary of the formula is the security when an attacker guesses at an optimal number of variables then runs either the Gröbner Bases algorithm

[6] The search methods of [6] is only useful against Signature Schemes over small fields [3].

[7] Within this restraint, lower v means higher FXL/F$\mathbf{F_5}$ complexity, see [34].

[8] Otherwise XL-type attacks can function at a much lower degree. This ([18]) breaks up more careless constructs like the latest version of TRMC [29].

$\mathbf{F_5}$ [12] or the FXL algorithm [7] using a sparse solver with speed comparable to Lanczos.

We do not know for sure what the parameters should be in Eq. 5. The theoretical best limit for $\mathbf{F_5}$ is given by ([34]) is roughly $c_0 = 4$, $c_1 = \frac{1}{4}$, $\gamma = 2$, $\omega = 2$ when counting field multiplications. To our knowledge no one comes close. Indeed, all commercially available software (including MAGMA, of the University of Sydney, which is reputed to be the best) have $\omega = 3$, according to many tests. A rough implementation of FXL with a sparse solver can currently do about $c_0 \approx 20$, $c_1 \approx 1$, $\gamma \approx 4$, $\omega = 2$.

11 Building Example Schemes: Enhanced TTS

What fast tame-like signature scheme would we come up with in full knowledge of what we now understand, to get to a complexity of 2^{80} 3DES blocks (2^{86} multiplications)?

1. The hash needs to be 160-bit, or $m \geq 20$ (birthday attacks), and $n \geq m$.
2. We need $m \geq 20$ for XL/$\mathbf{F_5}$ attacks (we would need $m \geq 22$ if $q = 2^7$).
3. We need $r > 8$, so there must be at least 5 independent cross-terms in each equation, probably 6 or 7 to account for the "crawl" of Sec. 9.
4. We do not want n too large because that adds to the key length and running times, and we may open ourselves to searching attacks cf. [3].
5. We need $u \geq 9$, so every variable must appear in at least 9 equations.

The following seems to be reasonable approaches to ensure the above:

- We choose not to search. Therefore we are restricted to an "Oil-and-Vinegar"-like approach of taking random values for some variables and solving for the rest.
- We need an initial segment with 6 or 7 cross-terms per equation. This will be solved as a linear system when the "vinegar-like" variables have been assigned.
- We need a final segment in at least the last 9 variables.
- One vinegar variable can provide one cross term per equation in the initial segment.
- If possible, the two systems we solve should be of equal dimension.

So we may do a signature scheme with the following central map ϕ_2:

$$
\begin{aligned}
y_i &= x_i + \textstyle\sum_{j=1}^{7} p_{ij} x_j x_{8+(i+j \bmod 9)},\ i = 8 \cdots 16;\\
y_{17} &= x_{17} + p_{17,1}x_1x_6 + p_{17,2}x_2x_5 + p_{17,3}x_3x_4\\
&\quad + p_{17,4}x_9x_{16} + p_{17,5}x_{10}x_{15} + p_{17,6}x_{11}x_{14} + p_{17,7}x_{12}x_{13};\\
y_{18} &= x_{18} + p_{18,1}x_2x_7 + p_{18,2}x_3x_6 + p_{18,3}x_4x_5\\
&\quad + p_{18,4}x_{10}x_{17} + p_{18,5}x_{11}x_{16} + p_{18,6}x_{12}x_{15} + p_{18,7}x_{13}x_{14};\\
y_i &= x_i + p_{i,0}x_{i-11}x_{i-9} + \textstyle\sum_{j=19}^{i-1} p_{i,j-18}\ x_{2(i-j)-(i \bmod 2)}\ x_j + p_{i,i-18}x_0x_i\\
&\quad + \textstyle\sum_{j=i+1}^{27} p_{i,j-18}\ x_{i-j+19}\ x_j,\ i = 19 \cdots 27.
\end{aligned}
$$

This is the [35] central map modified to avoid the UOV attack. Of course, we need to show that the new variant can scale up if our estimate is somewhat off, or to meet future, higher security requirements. We will discuss this next in Sec. 12. Note that our ϕ_2 above can be inverted reliably as follows:

1. Assign $x_1, \ldots, x_7$ and try to solve the first nine equations for x_8 to x_{16}.
2. If we fail to solve the first system of equations, just redo everything from scratch. The probability is around 255/256 that this system can be solved. As the determinant of the first system (for any x_1 through x_6) is a degree-9 polynomial in x_1 there can only be at most 9 choices of x_1 to make the first system degenerate, so the odds to solve this system is at least 247/256 and we will eventually hit upon a solution.
3. Solve serially for x_{17} and x_{18} using the next two equations (y_{17} and y_{18}).
4. Assign a random x_0 and try to solve the second system for x_{19} through x_{27}. Again, there will be at most nine x_0 that makes the determinant of the second system zero. So, if the first attempt to solve it fails, try other x_0 until a solution is found.

We will call this TTS/5 or Enhanced TTS (20,28). Its operates as follows:

To Generate Keys: Assign non-zero random values in $K = \mathrm{GF}(2^8)$ to parameters p_{ij}; generate random nonsingular matrices $\mathsf{M}_1 \in K^{28\times 28}$ and $\mathsf{M}_3 \in K^{20\times 20}$ (usually via LU decomposition) and vector $\mathbf{c}_1 \in K^{28}$. Compose $V = \phi_3 \circ \phi_2 \circ \phi_1$; assign $\mathbf{c}_3 \in K^{20}$ so that V has no constant part. Save quadratic and linear coefficients of V as public key (8680 bytes). Find M_1^{-1}, M_3^{-1}; save them with $\mathbf{c}_1$, $\mathbf{c}_3$, and the parameters p_{ij} as the private key (1399 bytes).

To Sign: From the message M, first take its digest $\mathbf{z} = H(M) \in K^{20}$, then compute $\mathbf{y} = \mathsf{M}_3^{-1}(\mathbf{z} - \mathbf{c}_3)$, then compute a possible $\mathbf{x} \in \phi_2^{-1}(\mathbf{y})$ as above: Our desired signature is $\mathbf{w} = \mathsf{M}_1^{-1}(\mathbf{x} - \mathbf{c}_1)$. Release $(M, \mathbf{w})$.

To Verify: On receiving $(M, \mathbf{w})$, compute $\mathbf{z} = H(M)$ and match with $V(\mathbf{w})$.

Scheme	Signature	PublKey	SecrKey	Setup	Signing	Verifying
RSA-PSS	1024 bits	128 B	320 B	2.7 sec	84 ms	2.0 ms
ECDSA	326 bits	48 B	24 B	1.6 ms	1.9 ms	5.1 ms
ESIGN	1152 bits	145 B	96 B	0.21 sec	1.2 ms	0.74 ms
QUARTZ	128 bits	71.0 kB	3.9 kB	3.1 sec	11 sec	0.24 ms
SFLASH^{v2}	259 bits	15.4 kB	2.4 kB	1.5 sec	2.8 ms	0.39 ms
TTS(20,28)	224 bits	8.6 kB	1.3 kB	1.5 ms	51 μs	0.11 ms
TTS(24,32)	256 bits	13.4 kB	1.8 kB	2.5 ms	67 μs	0.18 ms

Table 1. TTS and NESSIE round 2 candidate signature schemes on a 500MHz P3

12 Scaling Up Enhanced TTS

We can scale up Enhanced TTS to provide for a security of $C \gtrsim 2^{16k}$. This sequence of TTS instances we will call the "odd sequence" because u is odd. We

have (for $\ell \geq 4$) the $(m, n) = (4\ell, 6\ell - 2)$, with security parameters $(u, r, v) = (2\ell - 1, 4\ell - 6, 4\ell - 1)$

$$y_i = x_i + \sum_{j=1}^{2\ell-3} p_{ij} x_j x_{2\ell-2+(i+j+1 \bmod 2\ell-1)}, \text{ for } 2\ell - 2 \leq i \leq 4\ell - 4;$$

$$y_i = x_i + \sum_{j=1}^{\ell-2} p_{ij} x_{i+j-(4\ell-3)} x_{i-j-2\ell)} + \sum_{j=\ell-1}^{2\ell-3} p_{ij} x_{i+j-3\ell+6} x_{i+\ell-5-j}, \; i = 4\ell - 3 \text{ or } 4\ell - 2;$$

$$y_i = x_i + p_{i0} x_{i-2\ell+1} x_{i-2\ell-1} + \sum_{j=4\ell-1}^{i-1} p_{i,j-(4\ell-2)} x_{2(i-j)-(i \bmod 2)} x_j + p_{i,i-(4\ell-2)} x_0 x_i + \sum_{j=i+1}^{6\ell-3} p_{i,j-(4\ell-2)} x_{4\ell-1+i-j} x_j, \text{ for } 4\ell - 1 \leq i \leq 6\ell - 3.$$

To account for more optimistic estimates for FXL/F$\mathbf{F_5}$, there is a different sequence of Enhanced TTS instances with the same Rank Attack estimates. These instances are called the "even sequence" because the parameter u is even. In ϕ_2 below, we have $(m, n) = (4\ell, 6\ell - 4)$, with security parameters $(u, r, v) = (2\ell - 2, 4\ell - 10, 4\ell - 2)$.

$$y_i = x_i + \sum_{j=1}^{2\ell-5} p_{ij} x_j x_{2\ell-4+(i+j+1 \bmod 2\ell-2)}, \text{ for } 2\ell - 4 \leq i \leq 4\ell - 7;$$

$$y_i = x_i + \sum_{j=1}^{\ell-4} p_{ij} x_{i+j-(4\ell-6)} x_{i-j-(2\ell+1)} + \sum_{j=\ell-3}^{2\ell-5} p_{ij} x_{i+j-3\ell+5} x_{i+\ell-4-j}, \text{ for } 4\ell - 6 \leq i \leq 4\ell - 3;$$

$$y_i = x_i + p_{i0} x_{i-2(\ell+1)} x_{i-2(\ell-1)} + \sum_{j=4\ell-2}^{i-1} p_{i,j-(4\ell-3)} x_{2(i-j)-(i \bmod 2)} x_j + p_{i,i-(4\ell-3)} x_0 x_i + \sum_{j=i+1}^{6\ell-5} p_{i,j-(4\ell-3)} x_{4\ell-2+i-j} x_j, \text{ for } 4\ell - 2 \leq i \leq 6\ell - 5.$$

This ϕ_2 gives about $2^{16}\times$ higher FXL/F$\mathbf{F_5}$ complexity for corresponding instances. The performance of Enhanced TTS $(24, 32)$ is also given in Tab. 1.

Remark: A program for finding maximum cliques can verify that the UOV-attack parameter v is as given above. We have no space to explain the design.

We can estimate ϕ_2^{-1} to do $\approx 6k^2(k + 2)$ multiplications for small k. This almost equals the work done in matrices M_1 and M_3 at $m = 20$, $n = 28$, and will overtake them when m increases. We further know that asymptotically as k increases, the dimensions n and m to build a TTS instance or another tame-like scheme with security level 2^{16k} both increase linearly (cf. [34]). Thus, time cost of a TTS-like signature scheme goes up roughly with k^ω, where $2 < \omega \leq 3$ is the order of an elimination. Private map timings for RSA and ECC also increase between the quadratic and cubic to size. So the Triangular+OV construction will remain hundreds of times faster than RSA at comparable security levels. Table 2 gives this comparison. Timings on an 8051-compatible is essentially the same as in [35] and maintains a good lead over comparable schemes.

13 Discussions and Conclusion

There is recently a small resurgence of interest in multivariates, with perturbed variations of HFE [10] and C^* and the non-big-field signature schemes TRMS

m	n	PubKey	SecKey	Rank	Dual Rank	FXL	RSA bits	ECC bits	i8051 keygen	i8051 sign	i8051 code
20	28	8680 B	1399 B	2^{120}	2^{80}	2^{81}	≥ 1024	144	78.5s	170ms	1.6kB
24	32	13440 B	1864 B	2^{122}	2^{88}	2^{93}	≥ 1536	160	134s	227ms	1.6kB
28	38	21812 B	2594 B	2^{154}	2^{105}	2^{104}	≥ 2560	192	$\sim$ 300s	$\sim$ 500ms	$\sim$ 2kB
32	44	33088 B	3444 B	2^{186}	2^{121}	2^{115}	≥ 4096	224			
36	50	47700 B	4414 B	2^{220}	2^{138}	2^{133}	≥ 6144	256			

Table 2. Security Estimates of TTS instances, (m, n) = hash and signature sizes

([4], this resembles a tame-like system) and Rainbow [9], essentially a pre-sparsified version of TTS. This is a welcome development, obviously.

At the moment there are no serious reductionist "proof of security" study for multivariates. In that context, We have explained how the central map can affect the security under rank-based attacks and showed how combining the oil-and-vinegar and triangular approaches leads to tame-like signature schemes that are less susceptible to attack on rank.

Tame-like schemes are very fast. The Enhanced TTS instances given here needs no co-processor to run on a really low-end smart card [35]. There is however much research to be done before sparse variants can gain wide currency and trust.

References

1. M. Bardet, J.-C. Faugère, B. Salvy, and B.-Y. Yang, *Asymptotic Expansion of the Degree of Regularity for Semi-Regular Systems of Equations*, to be presented MEGA'05.
2. A. Braeken, C. Wolf and B. Preneel, *A Study of the Security of Unbalanced Oil and Vinegar Signature Schemes*, CT-RSA'05, LNCS 3376, pp. 29–43.
3. J.-M. Chen and B.-Y. Yang, *A More Secure and Efficacious TTS Scheme*, ICISC'03, LNCS 2971, pp. 320-338.
4. C.-Y. Chou, Y.-H. Hu, F.-P. Lai, L.-C. Wang, and B.-Y. Yang, *Tractable Rational Map Signature*, PKC'05, LNCS 3386, pp. 244–257.
5. D. Coppersmith, J. Stern, and S. Vaudenay, *Attacks on the Birational Permutation Signature Schemes*, Crypto'93, LNCS 773, pp. 435–443.
6. N. Courtois, L. Goubin, W. Meier, and J. Tacier, *Solving Underdefined Systems of Multivariate Quadratic Equations*, PKC'02, LNCS 2274, pp. 211–227.
7. N. Courtois, A. Klimov, J. Patarin, and A. Shamir, *Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations*, Eurocrypt'00, LNCS 1807, pp. 392–407.
8. C. Diem, *The XL-algorithm and a conjecture from commutative algebra*, Asiacrypt'04, LNCS 3329, pp. 338–353.
9. J. Ding and D. Schmidt, *Rainbow, a new multivariate polynomial signature system*, to appear at ACNS'05.
10. J. Ding and D. Schmidt, *Cryptanalysis of HFEv and Internal Perturbation of HFE*, PKC'05, LNCS 3386, pp. 288–301.
11. J. Ding and Z. Yin, *Cryptanalysis of TTS and Tame-like Multivariable Signature Schemes*, presentation at IWAP'04.

12. J.-C. Faugère, *A New Efficient Algorithm for Computing Gröbner Bases without Reduction to Zero (F5)*, Proceedings of ISSAC, ACM Press, 2002.
13. J.-C. Faugère and A. Joux, *Algebraic Cryptanalysis of Hidden Field Equations (HFE) Cryptosystems Using Gröbner Bases*, Crypto 2003, LNCS 2729, pp. 44-60.
14. M. Feldhofer, S. Dominikus, and J. Wolkerstorfer, *Strong Authentication for RFID Systems Using the AES Algorithm*, CHES '04, LNCS 3156, pp. 357–370.
15. M. Garey and D. Johnson, *Computers and Intractability, A Guide to the Theory of NP-completeness*, Freeman and Co., 1979, p. 251.
16. L. Goubin and N. Courtois, *Cryptanalysis of the TTM Cryptosystem*, Asiacrypt'00, LNCS 1976, pp. 44–57.
17. L. K. Grover, *A fast quantum mechanical algorithm for database search*, Proc. 28th Annual ACM Symposium on the Theory of Computing, (May '96) pp. 212–220.
18. A. Joux, S. Kunz-Jacques, F. Muller, P.-M. Ricordel, *Cryptanalysis of the Tractable Rational Map Cryptosystem*, PKC'05, LNCS 3386, pp. 258–274.
19. A. Kipnis, J. Patarin, and L. Goubin, *Unbalanced Oil and Vinegar Signature Schemes*, Crypto'99, LNCS 1592, pp. 206–222.
20. A. Kipnis and A. Shamir, *Cryptanalysis of the Oil and Vinegar Signature Scheme*, Crypto'98, LNCS 1462, pp. 257–266.
21. A. Kipnis and A. Shamir, *Cryptanalysis of the HFE Public Key Cryptosystem*, Crypto'99, LNCS 1666, pp. 19–30.
22. T. Matsumoto and H. Imai, *Public Quadratic Polynomial-Tuples for Efficient Signature-Verification and Message-Encryption*, Eurocrypt'88, LNCS 330, pp. 419–453.
23. T. Moh, *A Public Key System with Signature and Master Key Functions*, Communications in Algebra, 27 (1999), pp. 2207–2222.
24. NESSIE project, `www.cryptonessie.org`.
25. J. Patarin, *Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'88*, Crypto'95, LNCS 963, pp. 248–261.
26. J. Patarin, *Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms*, Eurocrypt'96, LNCS 1070, pp. 33–48.
27. J. Patarin, N. Courtois, and L. Goubin, *FLASH, a Fast Multivariate Signature Algorithm*, CT-RSA'01, LNCS 2020, pp. 298–307. Update at [24].
28. P. W. Shor, *Algorithms for quantum computation: Discrete logarithms and factoring*, Proc. 35th Ann. Symp. on Foundations of Comp. Sci., IEEE Comp. Soc. Press (1994), pp. 124-134.
29. L. Wang, *Tractable Rational Map Cryptosystem*, see ePrint 2004/046.
30. E. Weisstein, *RSA-576 Factored*, `mathworld.wolfram.com/news/2003-12-05/rsa`
31. C. Wolf, *Efficient Public Key Generation for Multivariate Cryptosystems*, Int'l Workshop on Cryptographic Algorithms and their Uses 2004, pp. 78–93, also ePrint 2003/089.
32. C. Wolf, A. Braeken, and B. Preneel, *Efficient Cryptanalysis of RSE(2)PKC and RSSE(2)PKC*, SCN '04, LNCS 3352, pp. 294–309.
33. C. Wolf and B. Preneel, *Taxonomy of Public-Key Schemes based on the Problem of Multivariate Quadratic Equations*, ePrint 2005/077.
34. B.-Y. Yang and J.-M. Chen, *All in the XL Family: Theory and Practice*, ICISC'04, LNCS 3506, pp. 67–86.
35. B.-Y. Yang, J.-M. Chen, and Y.-H. Chen, *TTS: High-Speed Signatures from Low-End Smartcards,*, CHES '04, LNCS 3156, pp. 371-385.

Potential Impacts of a Growing Gap Between Theory and Practice in Information Security (Extended Abstract)

Yvo Desmedt*

University College London, UK

Abstract. We have seen an explosion in the number of: conferences and workshops on all aspects of information security, topics studied in this field, and papers published. At the same time we have only seen a few widely deployed applications of this research. Meanwhile, computers are becoming less secure and criminals are exploiting new tricks. So, there is clearly a gap between theory and practice in information security. In this paper we analyze it and discuss the potential impacts of this gap.

1 Introduction

Outsiders often expect information security researchers to work on making a more secure version of Windows. They clearly do not understand what the impact is of "closed software". However, this should not be an excuse to avoid questioning the impact of modern research on the practical world.

Information security is supposed to be an applied discipline. However, when comparing it to many other applied disciplines, we could question whether all researchers working in this area are aware of it.

Modern software is usually unable to withstand elementary hacking attempts. One wonders what impact three decades of educating programmers in "the *art* of computer programming" [1] had[1]. Software engineering has failed to learn from other engineering disciplines, in which failures are rare. Moreover, if they occur, organizations as for example the Federal Aviation Administration, investigate the causes. During the internet boom the industry hired several programmers who had no formal training in computer science, let alone computer security.

We have seen plenty of examples of software in which information security was a low priority in its design. Often the state of the art is completely ignored. Worse many implementations do not even satisfy the state of the art of 20 years ago. Some experts even believe that patching software is preferable over a careful design of secure code. So, we see that the practical world is moving away from applying the results of research, even old ones.

* The author is BT Chair of Information Security. This work has also been partially supported by EPSRC EP/C538285/1.

[1] Still today many Computer Science departments are part of the college of *Arts* and Sciences.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 532–536, 2005.

While computer security is in a crisis, we see that conferences, such as ESORICS and the IEEE Symposium on Security and Privacy (Oakland) are now accepting several papers on cryptography! Moreover, at conferences on cryptography, we still find plenty of papers on digital cash, undeniable signatures, anonymous MIX servers, etc., topics that 10-20 years ago seemed very promising, but which today seem far away from being widely deployed. Moreover, we find plenty of theory papers at such venues. Moreover, several papers at conferences on cryptography continue to be based on discrete logarithm and RSA, assumptions that will be broken with future quantum computers.

In conclusion, we find a growing gap between between theory and practice. Actually, when we analyze this gap further in Section 2, we see that this gap consists of multiple ones. We discuss the many gaps that one finds within academia, within industry, between the theoretical and practical community, etc. (The possible origins and reasons for these gaps are discussed in Section 3.)

Since the discipline should be an applied one, a growing gap may have quite some negative impacts. We discuss a few potential ones in Section 4. It is clear that one should try to prevent these negative impacts by taking corrective measures. Some of these are enumerated in Section 5.

2 Identifying Some Gaps

Inside academia we see a gap between computer security experts and cryptographers. Some have made broad statements, such as:

> "Using encryption on the Internet is the equivalent of arranging an armored car to deliver credit card information from someone living in a cardboard box to someone living on a park bench." (Gene Spafford)

Although the comment may offend the researcher working on cryptography, the statement should be seen in the context of the principle of easiest penetration. This means, while eavesdropping was once one of the easiest way to obtain confidential information, today the fact that *many* computers on the internet are not secure enough, implies that there are other means to obtain the information. At the same time, we see that people who did work on cryptography, blame programmers for incompetence:

> "If buffer overflows are ever controlled, it won't be due to mere crashes, but due to their making systems vulnerable to hackers. Software crashes due to mere incompetence apparently don't raise any eyebrows, because no one wants to fault the incompetent programmer **and his incompetent boss**." (Henry Baker)

Another gap is the one between the research that seems applicable, but which has not been implemented. We just take one example. Electronic voting based on cryptography, has now generated several research papers. Regardless the fact that the topic was studied for 20 years in the cryptographic community, the US National Science Foundation 2001 panel on Internet Voting [4, pp. 1–2] wrote an overly cautions report. The report for example stated:

> *Poll site Internet voting systems offer some benefits and could be responsibly fielded within the next several election cycles.* While many issues remain to be addressed, the problems associated with these systems appear likely to be resolvable in the near term. ...

Evidently, it is no surprise that industry with good marketing strategies which had voting technology ready, pushed their solutions instead of the one based on cryptography! This overly cautions behavior could be put in sharp contrast with the early days of the research on cryptography. Indeed the RSA corporation was set up a few years after the invention of the RSA algorithm. It is ironic, to say the least, that some people on the panel on Internet Voting, criticized the Diebold scheme heavily [2]. One can wonder whether less cautionary statements would have contributed to the widespread deployment of the research.

Some gaps are primarily based on lack of information. In the 1980's and early 1990's, proven security was primarily a theoretical concept. People who have not followed the latest research results unfortunately still believe that this is the case today. This lack of knowledge creates further a conceived gap.

3 Possible Origins of the Gaps

We already identified several issues that may have contributed to the gaps.

Many researchers in computer security and beyond learned about cryptography from Schneier's book [5]. When the book was published, several practical zero-knowledge proofs existed, and the Fiat-Shamir trick to turn them into digital signature schemes was well known. However, a lot of research on practical oriented proven secure cryptography has been done since Schneier's publication. Unfortunately, outsiders have not realized this.

We have also seen a boom of universities that offer courses in information security, e.g. cryptography and/or computer security. In the USA this was encouraged by having the "NSA Centers of Excellence in Information Security and Assurance Education." Today 50 such centers exist. We also see that several universities offer an MSc in information security. A problem is that some of these universities do not have the expertise in the area. Sometimes outdated textbooks are used or the material has not been updated. For example several so called practical courses fail to even mention OAEP. In an area that evolves quickly, an education not based on the state of the art will only enlarge the gap.

The cryptographic applications an average user comes in contact with are ssh and ssl. Very few other cryptographic protocols, schemes are implemented and widely deployed. There are several reasons for this. First of all, the road from a great idea to a successful implementations is a bumpy one. Theoreticians tend not to be aware of this. One needs to carefully match theory with needs. Often standards and prototypes are developed. To become a successful product issues as user-friendliness and marketing are important.

One could wonder why few academics in information security implement their own ideas, or ideas of others. Without a major conference[2] promoting such *industrial* implementations, focusing on implementations is a kiss of death in academia for an assistant professor in information security.

Another problem is the difference in language spoken between system managers, software developers and information security researchers. System managers often had no formal training in information security. Sometimes this leads to settings of the system that are not user friendly, a major requirement in a world where most users had no computer training.

4 Potential Negative Impacts

One potential problem is that funding agencies may be willing to invest heavily in information security research today since it seems our research may bring more secure computers. However, since this is an illusionary goal for many researchers, funding agencies may become critical to fund research in this discipline. So, the gap may haunt the research community in the long run. Evidently, this does not mean that theoretical researchers should start promising deliverables in vain. This would only aggravate the problem.

Since computer security seems to be in much worse shape than cryptography, it may seem that society has invested too much in cryptography. The recent cryptanalysis of MD4, MD5 and SHA-1, etc. clearly show that such a conclusion is incorrect.

One should also be careful not to blame the theoretical community. If the computer security community would have properly balanced long term research with short term development, may be, as much progress would had been made in the field as in modern cryptography. Indeed recently some academics have started to criticize the US DARPA funding agency for its short sighted approach [3]. However, this still does not eliminate potential criticism that an unbalanced emphasis on theory may in the long term undermine the funding for theoretical foundations of information security, even though this may be very important from a long term viewpoint.

Another potential problem is that we may be graduating too many experts in the area compared to the demand. Such a conclusion is only correct if industry continues to develop software without taking security into account. This situation may change depending on pressure put on the software industry, which may originate from new laws, lawsuits, etc. However, even if industry is forced to address this, it may need more computer security experts than cryptographers, except if more applications are deployed. Evidently future experts need to be aware of issues one often ignores in information security, such as user-friendliness, lack of infrastructures (such as PKI), etc.

[2] Although Usenix is interested in implementations, it does not focus on software which is commercially deployable.

5 Corrective Measures

By being aware of the potential negative impacts one can start thinking about what corrective measures need to be taken. Those who should get involved are information security experts, information security advocates, people concerned about their privacy, etc. They may be based in academia or industry, be in research or development or in system management, etc.

We should *consider* encouraging lobbyists to mandate more secure operating systems and software. A long term process, as was followed in the US for reducing car pollution, balances the need of society and spreads out over years the cost for industry to adapt. The current atmosphere of deregulation will not help. However, history teaches us that experiments with anarchy failed badly.

It seems clear that we need to set up a respected venue (conference, workshop) to promote the development of research ideas into practical products and in particular to encourage widespread implementation. Academics should also collaborate closer with software engineers, etc. Evidently, this is only one step in the promotion of technology transfer. One should encourage researchers to be involved in standards. However, this is far from trivial in academia.

One should encourage researchers in computer security to analyze its foundations and to build bridges with cryptography. For example, can software be built that is guaranteed hacking free, except if, let say, factoring is hard? Also, can multi-party computation help address privacy concerns in databases, without reducing their performance too much?

We need to revise the education in information security. One solution, which was proposed to the author, is to have accreditation. A proper education should balance foundations with practical needs. If properly done, it is one way to guarantee students learn the state of the art.

Acknowledgment

The author is grateful for discussions with information security experts, in particular with Fred Piper, Victor Shoup and Yuliang Zheng.

References

1. D. E. Knuth. *The Art of Computer Programming, Vol. 1, Fundamental Algorithms.* Addison-Wesley, Reading, MA, 1973.
2. T. Kohno, A. Stubblefield, A. D. Rubin, and D. Wallach. Analysis of an electronic voting system. In *Proceedings IEEE Symposium on Security and Privacy*, pp. 27–42. IEEE Computer Society, May 2004. Oakland, California.
3. Assessment of department of defense basic research. National Research Council, `http://www.nap.edu/openbook.php?record_id=11177&page=R1`, 2005.
4. Report of the national workshop on internet voting. news.findlaw.com/cnn/docs/voting/nsfe-voterprt.pdf, March 2001.
5. B. Schneier. *Applied Cryptography.* J. Wiley, New York, first edition, 1994.

Security Analysis and Fix of an Anonymous Credential System

Yanjiang Yang[1,2], Feng Bao[1], and Robert H. Deng[2]

[1] Institute for Infocomm Research,
21 Heng Mui Keng Terrace, Singapore 119613
[2] School of Information Systems
Singapore Management University, Singapore 259756

Abstract. Anonymous credentials are an important privacy-enhancing technique that allows users to convince a service provider of their legitimacy for service accesses in an anonymous manner. Among others, a fundamental feature of anonymous credentials is *unlinkability*, that is, multiple showings of the same credential should not be linked by the service providers, the issuing organization, or the coalition of the two. Recently, Persiano et. al. proposed an interesting anonymous credential system, which was claimed to be unlinkable. In this paper, we prove that their unlinkability claim is false. In particular, we show that the issuing organization can easily relate two showings of the same credential, point out the flaw in their original security proof and present a fix to avoid our attack.

Keyword: Anonymous Credentials, Privacy, Unlinkability, Chameleon Certificate.

1 Introduction

Individual privacy is increasingly becoming a concern in the digital age. Anonymous credentials are an important privacy-enhancing technique that achieves individual privacy in user identification and access control. Typically, three types of parties are involved in a credential system: users, credential issuing organizations and service providers. A user is issued credentials (or credential certificates)[3] by an issuing organization, asserting the user's legitimacy to receive services from certain service providers. To access a service, the user engages in a *credential showing* with the corresponding service provider, proving possession of the required credential(s) as qualification for the service in question. If the credential caters for the established access policy, the service provider grants service to the user; otherwise it rejects the user's request. Credential systems are deemed

[3] In the simplest form, a credential is a statement stating its holder's qualification or capability of a certain type, and a credential certificate is a signature on the credential(s).

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 537–547, 2005.

a promising solution for user identification and access control in current progressively distributed environment, where traditional centralized access control mechanisms such as password and ACL [14] are not effective. Upon regular credentials, anonymous credentials further enable credential showings to proceed in an anonymous manner, that is, different showings of the same credential cannot be recognized as such (*unlinkability*). Unlinkability is among the fundamental features of anonymous credentials. Anonymous credentials are either one-show or multi-show, depending on the number of times a credential can be used: a one-show credential is not allowed to be used more than once while a multi-show credential can be repeatedly used.

Anonymous credentials achieve strong user privacy towards the service providers during service accessing. This is thus of paramount importance as users become more and more concerned with their information being collected and compiled by the service providers. As such, a considerable amount of effort has been dedicated to anonymous credential systems. The idea of transferring credentials from one organization to another while keeping anonymous to the organizations was first introduced in [4], and soon the first concrete implementation based on a semi-trusted third party involved in all transactions was proposed in [5]. However, the active involvement of a third party to achieve unlinkability is deemed a big disadvantage. Later, a credential scheme was presented in [9] by means of general complexity-theoretic tools such as one-way functions and zero-knowledge proof, which makes the scheme quite inefficient for practical use. The same efficiency problem exists in the general credential system in [11], although this system captures most desirable features of anonymous credentials. The scheme proposed in [6] is efficient and based on DL blind signatures, whereas it is vulnerable to collusion of users. Moreover, both systems in [6,11] require a user to obtain several signatures from the credential issuing organization. It is believed that the first *practical* anonymous credential system belonged to [7], which was based upon the strong RSA assumption and the DDH assumption. Methods for constructing both one-show and multi-show anonymous credentials were presented in [7]. More efficient construction of multi-show credentials was due to [17], built upon the groups over weil pairing where DDH is easy while CDH is hard.

Another approach was attributed to [2], where through zero-knowledge protocols, a user can prove that the credentials encoded in the corresponding credential certificate satisfy a given linear Boolean formula. Such approaches were developed and extended from earlier results on zero-knowledge proof systems [16,1,15]. The constructions in [2] possess an appealing feature that it is possible to use a single credential certificate for many services, given their access policies can be expressed as linear Boolean formulae. Note that this aspect is not enjoyed by the aforementioned schemes, e.g., [7,17]. A main weakness in [2] however, is that the certificates are one-show, and different showings of the same certificate are linkable.

Recently, Persiano et. al. [13] presented a novel anonymous credential system, attempting to simultaneously achieve the advantages of the above separate lines

of work. In particular, a credential (together with the certificate) is multi-show as in [7,17], and the same credential (certificate) can be used for any service as in [2]. Unfortunately, we show in this paper that the issuing organization can readily link different showings of the same credential certificate, thus unlinkability as claimed in [13] is not true. This in turn suggests the failure of achieving unlinkability in the case of a coalition of the issuing organization and the service providers.

The rest of the paper is organized as follows. In Section 2, we review the anonymous credential system proposed in [13]. We then analyze the security of the system in Section 3 and present a fix to the system so as to avoid our attack in Section 4. Section 5 contains the concluding remarks.

2 Review of the Original System

The basic idea in [13] is that the issuing organization issues a *master chameleon certificate* encoding the credentials that are entitled to the user. Each time requesting service from the service provider, the user constructs and shows a distinct *slave chameleon certificate* that is built from the master certificate. To remain anonymous, slave chameleon certificates must be unlinkable with one another and with the master chameleon certificate. The complete system consists of the following phases.

System set-up

In this phase, the issuing organization $\mathcal{O}$ establishes the system parameters as follows:

1. picks a RSA key pair ($P_o = (n, e)$, $S_o = (n, d)$), where $n = pq$ and p, q are k-bit primes, P_o is a public key and S_o the corresponding private key.
2. picks 5 elements $g_0, g_1, g_2, g_3, g_4 \in_R Z_n^*$, and another $g \in_R Z_n^*$ such that the order $ord(g)$ is unknown.
3. computes a signature $s = g^d \pmod n$ of g.
4. publishes the public system parameters $(P_o, g, s, g_0, g_1, g_2, g_3, g_4)$.

User enrolment

In this phase, issuing organization $\mathcal{O}$ issues to user $\mathcal{U}$ a credential certificate on the credentials x_1 and x_2 as well as $\mathcal{U}$'s public key $P_u = g_0^{x_0} \bmod n$, where x_0 is the corresponding private key of $\mathcal{U}$. Specifically,

1. $\mathcal{U}$ sends x_1, x_2, P_u in clear to $\mathcal{O}$, along with a proof of the knowledge of the discrete logarithm of P_u to g_0.
2. after verifying $\mathcal{U}$'s conformance to an access policy, $\mathcal{O}$ chooses $x_3, x_4 \in_R Z_n^*$ and sends back a master chameleon certificate (C, S) together with x_3 and x_4 to $\mathcal{U}$, where

$$C = P_u g_1^{x_1} g_2^{x_2} g_3^{x_3} + g_4^{x_4} \pmod n \tag{1}$$

$$S = C^d \pmod n \tag{2}$$

Refreshing

Each time to show the credentials, $\mathcal{U}$ executes the following *refreshing* procedure to construct a new slave chameleon certificate from the master chameleon certificate.

1. $\mathcal{U}$ picks $x \in_R Z_n^*$ and computes $C' = g^x.C \pmod n$.
2. $\mathcal{U}$ computes a signature S' of C' as $S' = s^x.S \pmod n$. The slave chameleon certificate is $(C',\ S')$.

Credential showing

To show his qualification for the service in question, $\mathcal{U}$ needs to prove possession of a master chameleon certificate $(C,\ S)$ issued by $\mathcal{O}$ to the service Provider $\mathcal{SP}$. Note that the access control policy of $\mathcal{SP}$ is expressed by a linear Boolean formula ψ, and $\mathcal{U}$ is required to prove that the credentials encoded in the master chameleon certificate satisfy ψ. In particular, the *credential showing* proceeds in the following steps:

1. $\mathcal{U}$ constructs a slave chameleon certificate $(C',\ S')$ by choosing $x \in_R Z_n^*$ such that $\psi(x, x_0, x_1, x_2, x_3) = 1$, computes $C'_0 = g^x g_0^{x_0} g_1^{x_1} g_2^{x_2} g_3^{x_3} \pmod n$ and $C'_1 = g^x g_4^{x_4} \pmod n$. Clearly, $C' = C'_0 + C'_1 \pmod n$.
2. $\mathcal{U}$ computes commitments $(\hat{C}'_0,\ \hat{C}'_1,\ \hat{C}')$ of $(C'_0,\ C'_1,\ C')$ using, e.g., the technique of [8]. $\mathcal{U}$ then sends the commitments to $\mathcal{SP}$.
3. $\mathcal{SP}$ replies with $b \in_R \{0, 1, 2\}$ as a challenge.
4. If $b = 0$: $\mathcal{U}$ proves that $(\hat{C}'_0,\ \hat{C}'_1,\ \hat{C}')$ are well formed. In particular, $\hat{C}'$ is the commitment of two values whose commitments are $\hat{C}'_0$ and $\hat{C}'_1$, and $\hat{C}'$ is the commitment of C'. This is accomplished by the techniques in [8]. Upon that, $\mathcal{U}$ sends S', and $\mathcal{SP}$ verifies that S' is a valid signature of C'.
5. If $b = 1$: $\mathcal{U}$ opens $\hat{C}'_0$ and both parties engage in a PoK (Proof of Knowledge) by which $\mathcal{U}$ proves to know (x, x_0, x_1, x_2, x_3) satisfying $C'_0 = g^x g_0^{x_0} g_1^{x_1} g_2^{x_2} g_3^{x_3} \pmod n$ and $\psi(x, x_0, x_1, x_2, x_3) = 1$. This is achieved by the techniques in [2].
6. If $b = 2$: $\mathcal{U}$ opens $\hat{C}'_1$ and proves the knowledge of (x, x_4) satisfying $C'_1 = g^x g_4^{x_4} \pmod n$.

$\mathcal{U}$ and $\mathcal{SP}$ repeat these steps for several times so as to achieve a satisfactory level of soundness. We refer interested readers to [13] for more details.

The system has many desirable features of anonymous credential systems such as unforgeability, privacy, usability, and non-transferability. Moreover, the system is claimed to have *unlinkability*, a central element of anonymous credentials, with respect to the the issuing organization and a coalition of the issuing organization and the service providers. Unfortunately, we show in what follows that the issuing organization can easily compromise unlinkability. This also suggests that the system cannot achieve unlinkability towards a coalition of the issuing organization and the service providers.

3 Security Analysis

In this section, we show how the issuing organization $\mathcal{O}$ makes the different showings of the same anonymous credential certificate *linkable*, thereby compromising the purported unlinkability. We also point out the flaw in the security proof in [13]. We start by listing some relevant facts in number theory that will be extensively used in the sequel.

3.1 Preliminaries

Fact 1. *Let $n = pq$, where $p = 2p' + 1$, $q = 2q' + 1$ are safe primes, that is, p, q, p' and q' are all primes. Then it must hold that:*
*(**a**) the order of an element in Z_n^* is one of $\{1, 2, p', q', 2p', 2q', p'q', 2p'q'\}$;*
*(**b**) given an element $h \in_R Z_n^* \setminus \{-1, 1\}$ such that $ord(h) < p'q'$, then either $gcd(h\text{-}1, n)$ or $gcd(h\text{+}1, n)$ is a factor of n;*
*(**c**) given an element $h \in_R Z_n^* \setminus \{-1, 1\}$ such that $ord(h) = p'q'$ or $2p'q'$, then $gcd(h \pm 1, n) = 1$;*
*(**d**) based on (a), (b) and (c), it is clear that an element $h \in_R Z_n^* \setminus \{-1, 1\}$ has order $p'q'$ and $2p'q'$ with an overwhelming probability of 1/4 and 3/4, respectively,*

Proof. (**a**) It suffices to note that the maximal order of an element in Z_n^* is $2p'q'$ and the order of each element must divide $2p'q'$ (see e.g., [3]).
(**b**) It follows from (a) that $ord(h) \in \{2, p', q', 2p', 2q'\}$ in this case, since the only element with order of 1 is 1. We next separately consider each case:

$ord(h) = 2$: that is $h^2 = 1 \bmod n \Rightarrow n|(h-1)(h+1)$ if $h \neq \pm 1$, so each of $h - 1$ and $h + 1$ must contain a prime factor of n.

$ord(h) = p'$: that is $h^{p'} = 1 \bmod n \Rightarrow h^{p'} = 1 \bmod q$. If $h \neq 1 \bmod q$, then $(\phi(q) = 2q')|p'$, a contradiction. Therefore, it must hold that $h = 1 \pmod q \Rightarrow q|(h-1)$. So $gcd(h-1, n) = q$ as $h - 1 < n$. Similarly, $gcd(h-1, n) = p$ in the case of $ord(h) = q'$.

$ord(h) = 2p'$: that is $h^{2p'} = 1 \pmod n \Rightarrow (h^2)^{p'} = 1 \pmod q$. As just discussed, it must hold that $h^2 = 1 \pmod q \Rightarrow q|(h-1)(h+1)$. So either $gcd(h-1, n) = q$ as $h - 1 < n$ or $gcd(h+1, n) = q$ as $h + 1 < n$. A similar argument holds for $ord(h) = 2q'$.

(**c**) This immediately follows from (b), together with the fact $ord(h) < p'q'$ if $gcd(h \pm 1, n) \neq 1$.

(**d**) Note that the elements in Z_n^* of order $p'q'$ are included in the subgroup QR_n of quadratic residues modulo n. To see this, for any element $h \in Z_n^*$ of order $p'q'$, it holds that $h^{p'q'} = 1 \pmod n \Rightarrow h^{p'q'} = 1 \pmod p$ and $h^{p'q'} = 1 \pmod q$. Suppose if $h^{p'} \neq 1 \pmod p$ and $h^{q'} \neq 1 \pmod q$, then it must hold that $(\phi(p) = 2p')|p'q'$ and $(\phi(q) = 2q')|p'q'$, which is impossible. So it must be that $h^{p'} = 1 \pmod p$ and $h^{q'} = 1 \pmod q \Rightarrow h^{(p-1)/2} = 1 \pmod p$ and $h^{(q-1)/2} = 1 \pmod q$. According to the well known Euler's Criterion, h is a quadratic residue modulo p and q, so h is in turn a quadratic residue modulo n. While QR_n includes elements of order $< p'q'$, the number of these elements is negligibly small according to (a), (b) and (c). It is thus safe to presume that the

elements of order $p'q'$ comprise QR_n. Finally, it remains and suffices to know that 1/4 of the elements in Z_n^* are quadratic residues and 3/4 are quadratic non-residues. □

Fact 2. *With the knowledge of the prime factorization of n, it is easy to compute the square roots of a quadratic residue modulo n.*

The proof for this fact is straightforward and can be found in any textbook for number theory.

3.2 Our Attack

We next present our attack, by which issuing organization $\mathcal{O}$ can link different showings of the same credential certificate. This is achieved by $\mathcal{O}$ manipulating the system parameters as follows:

1. picks a RSA key pair ($P_o = (n, e)$, $S_o = (n, d)$), where $n = pq$ (p and q are k-bit primes) such that the factorization of $\phi(n) = (p-1) * (q-1)$ is known. For purely exposition purposes, we assume that $p = 2p' + 1$ and $q = 2q' + 1$ where p' and q' primes. Consequently, it holds that $\phi(n) = 4p'q'$, where $\phi(.)$ is the Euler function.
2. picks $g_0, g_1, g_2, g_3, g_4, g \in_R Z_n^*$, such that $ord(g)$ is smaller than the orders of g_0, g_1, g_2, g_3, g_4. From **Fact 1**, it is easy to see that in such a case $ord(g) = p'q'$ and $ord(g_0)$, $ord(g_1)$, $ord(g_2)$, $ord(g_3)$, $ord(g_4)$ are all equal to $2p'q'$. $\mathcal{O}$ can easily select those elements since the order a randomly chosen element from Z_n^* is either $2p'q'$ or $p'q'$ as we just discussed in **Fact 1**. It is important to note that as $\mathcal{O}$ has the full control of the system set-up, it is impossible to guarantee that the order of g is unknown as assumed in [13]. Even if the choice of g were beyond the control of $\mathcal{O}$, e.g., g were computed from a publicly known value by applying a one-way hash function, $\mathcal{O}$ still knows $ord(g)$ since $\mathcal{O}$ has the knowledge of p' and q'.
3. $\mathcal{O}$ completes the remaining steps of the system set-up, including the computation of a signature $s = g^d \pmod n$ of g, and the publishing of the public system parameters $(P_o, g, s, g_0, g_1, g_2, g_3, g_4)$.

Clearly, no party other than issuing organization $\mathcal{O}$ itself is able to detect the above tricks. With the system parameters established as such, we next illustrate how different slave credential certificates built from the same master certificate can be linked by $\mathcal{O}$. Recall that in the *credential showing* phase (cf. Section 2), $\mathcal{U}$ submits C' in step 4 ($C' = S'^e \pmod n$), C'_0 in step 5, and C'_1 in step 6, in response to the challenge $b = 0, 1, 2$, respectively. Since steps in credential showing are repeated many times, there is an overwhelming probability that all of C', C'_0 and C'_1 are collected during a single service request. Suppose $\mathcal{O}$ has $(\widetilde{C}', \widetilde{C}'_0, \widetilde{C}'_1)$ and $(\widehat{C}', \widehat{C}'_0, \widehat{C}'_1)$ that correspond to two distinct service requests, where:

$$\begin{aligned}
&\widetilde{x} \in Z_n^* \\
&\widetilde{C}' = g^{\widetilde{x}}(g_0^{\widetilde{x}_0} g_1^{\widetilde{x}_1} g_2^{\widetilde{x}_2} g_3^{x_3} + g_4^{\widetilde{x}_4}) \bmod n \\
&\widetilde{C}'_0 = g^{\widetilde{x}} g_0^{\widetilde{x}_0} g_1^{\widetilde{x}_1} g_2^{\widetilde{x}_2} g_3^{\widetilde{x}_3} \bmod n \\
&\widetilde{C}'_1 = g^{\widetilde{x}} g_4^{\widetilde{x}_4} \bmod n
\end{aligned} \tag{3}$$

and

$$\begin{aligned}
&\widehat{x} \in Z_n^* \\
&\widehat{C}' = g^{\widehat{x}}(g_0^{\widehat{x}_0} g_1^{\widehat{x}_1} g_2^{\widehat{x}_2} g_3^{\widehat{x}_3} + g_4^{\widehat{x}_4}) \bmod n \\
&\widehat{C}'_0 = g^{\widehat{x}} g_0^{\widehat{x}_0} g_1^{\widehat{x}_1} g_2^{\widehat{x}_2} g_3^{\widehat{x}_3} \bmod n \\
&\widehat{C}'_1 = g^{\widehat{x}} g_4^{\widehat{x}_4} \bmod n
\end{aligned} \tag{4}$$

In the above, we assume the master credential certificates correspond to $(\widetilde{C}', \widetilde{C}'_0, \widetilde{C}'_1)$ and $(\widehat{C}', \widehat{C}'_0, \widehat{C}'_1)$ are $(\widetilde{C} = g_0^{\widetilde{x}_0} g_1^{\widetilde{x}_1} g_2^{\widetilde{x}_2} g_3^{\widetilde{x}_3} + g_4^{\widetilde{x}_4}, \widetilde{S} = \widetilde{C}^d \bmod n)$ and $(\widehat{C} = g_0^{\widehat{x}_0} g_1^{\widehat{x}_1} g_2^{\widehat{x}_2} g_3^{\widehat{x}_3} + g_4^{\widehat{x}_4}, \widehat{S} = \widehat{C}^d \bmod n)$, respectively. Clearly, the following computations must hold if $(\widetilde{C}', \widetilde{C}'_0, \widetilde{C}'_1)$ and $(\widehat{C}', \widehat{C}'_0, \widehat{C}'_1)$ are derived from the same master credential certificate:

$$(\widetilde{C}'/\widehat{C}')^{ord(g)} = (g^{\widetilde{x}-\widehat{x}})^{ord(g)} = 1 \bmod n \tag{5}$$

$$(\widetilde{C}'_0/\widehat{C}'_0)^{ord(g)} = (g^{\widetilde{x}-\widehat{x}})^{ord(g)} = 1 \bmod n \tag{6}$$

$$(\widetilde{C}'_1/\widehat{C}'_1)^{ord(g)} = (g^{\widetilde{x}-\widehat{x}})^{ord(g)} = 1 \bmod n \tag{7}$$

Conversely however, if $(\widetilde{C}', \widetilde{C}'_0, \widetilde{C}'_1)$ and $(\widehat{C}', \widehat{C}'_0, \widehat{C}'_1)$ are built from different master credential certificates, we have

$$(\widetilde{C}'/\widehat{C}')^{ord(g)} = (\widetilde{C}'\widehat{C}'^{-1})^{ord(g)} \bmod n \tag{8}$$

$$(\widetilde{C}'_0/\widehat{C}'_0)^{ord(g)} = (g_0^{\widetilde{x}_0-\widehat{x}_0} g_1^{\widetilde{x}_1-\widehat{x}_1} g_2^{\widetilde{x}_2-\widehat{x}_2} g_3^{\widetilde{x}_3-\widehat{x}_3})^{ord(g)} \bmod n \tag{9}$$

$$(\widetilde{C}'_1/\widehat{C}'_1)^{ord(g)} = (g_4^{\widetilde{x}_4-\widehat{x}_4})^{ord(g)} \bmod n \tag{10}$$

Recall that in our attack $ord(g_i) > ord(g)$ for $i = 0, 1, 2, 3, 4$, so the results in (9) and (10) will not be 1 mod n unless $ord(g_i)|(\widetilde{x}_i - \widehat{x}_i)ord(g)$ holds for all $i = 0..4$. Since $ord(g_i) = 2 \times ord(g)$, the probability of $ord(g_i)|(\widetilde{x}_i - \widehat{x}_i)ord(g)$ for $i = 0, 1, 2, 3, 4$ is $1/2^5$. In other words, we have a probability of $1 - 1/2^5$ that (9) and (10) are not simultaneously equal to 1 mod n. In this attack, we have chosen p and q as safe primes for purely exposition purposes. If instead p and q are chosen in other forms, e.g., $p = 2\epsilon p' + 1$ and $q = 2q' + 1$, where ϵ is a small prime, such that $ord(g) = p'q'$ and $ord(g_i) = \epsilon p'q'$, we can tremendously increase the probability of $ord(g_i) \nmid (\widetilde{x}_i - \widehat{x}_i)ord(g)$. The above discussion suffices to show the system in the above attack cannot guarantee unlinkability. Now refer to $(\widetilde{C}'/\widehat{C}')^{ord(g)}$ in Equation (8): even $\widetilde{C}'$ and $\widehat{C}'$ are from distinct master credential certificates, it is still possible that $ord(\widetilde{C}'/\widehat{C}') = ord(g)$. The reason

is that while $ord(g_0)$, $ord(g_1)$, $ord(g_2)$, $ord(g_3)$ and $ord(g_4)$ are equal to $2p'q'$, the value of $g_0^{x_0} g_1^{x_1} g_2^{x_2} g_3^{x_3} + g_4^{x_4} \pmod{n}$ presents itself as an random element in Z_n^*, thus having a probability of 1/4 to have order $p'q'$ according to **Fact 1(d)**.

Coalition of $\mathcal{O}$ and $\mathcal{SP}$

In a coalition of $\mathcal{O}$ and $\mathcal{SP}$, if $\mathcal{O}$ passes $ord(g) = p'q'$ to $\mathcal{SP}$, then $\mathcal{SP}$ can clearly launch the same attack as $\mathcal{O}$ does in the above. This however is equivalent to leaking $\mathcal{O}$'s secret signing key d (where $ed = 1 \mod 4p'q'$) to $\mathcal{SP}$. Usually, $\mathcal{O}$ cannot afford such a disclosure even in the case of coalition. As such, $\mathcal{SP}$ instead supplies protocol transcripts (C', C'_0, C'_1) to $\mathcal{O}$ and relies on $\mathcal{O}$ to launch the attack.

3.3 Flaw in the Security Proof

A security proof on the unlinkability of the credentials was given in [13] based on the assumption that states: *it is not possible for an efficient algorithm on input $x, g \in_R Z_n^*$, where $n = pq$ and p, q are primes, to establish whether $x \in \langle g \rangle$ ($\langle g \rangle$ denotes the group generated by g) even if the factorization of n is known.* That is, the security proof in [13] reduces the validity of the above assumption to the unlinkability of the credentials. More specifically, suppose an adversary $\mathcal{A}$ that breaks the unlinkability of the credentials, then another algorithm $\mathcal{A}'$, taking as inputs two primes p, q of the same length and y, $g \in_R Z_n^*$ where $n = pq$, invokes $\mathcal{A}$ to output a guess on whether $y \in \langle g \rangle$ (the reader is referred to [13] for details). What's wrong with this reduction proof? Without loss of generality, suppose $p = 2p' + 1$ and $q = 2q' + 1$ are both safe primes as in our attack and $ord(g) = p'q'$, then $\mathcal{A}'$ can decide whether $y \in \langle g \rangle$ without referring itself to $\mathcal{A}$ at all. To see this, we know from **Fact 1(d)** that $\langle g \rangle$ refers to the same group as QR_n, so $y \in \langle g \rangle \Leftrightarrow y \in QR_n$. It is well known that deciding the quadratic residuosity of an element modulo a composite is easy if factorization of the composite is known. Clearly, the assumption is false when $g \in QR_n$.

4 Our Fix

In this section, we fix the original system so that it can resist our attack. Note that our attack is essentially due to the fact that $ord(g)$ is smaller than the order of g_i, $i = 0, 1, 2, 3, 4$. Our fix is thus to force issuing organization $\mathcal{O}$ to make $ord(g)$ at least equal to the order of other elements. To that end, we modify the *system set-up* phase and the *user enrolment* phase, while keeping other phases of the original system unchanged.

System set-up:

1. $\mathcal{O}$ picks a RSA key pair $(P_o = (n, e), S_o = (n, d))$, where $n = pq$, $p = 2p' + 1$ and $q = 2q' + 1$ such that p, q, p' and q' are all prime numbers. Note that $\mathcal{O}$ needs to either prove to the users in the later *user enrolment* or prove to a trusted third party that n is a product of two safe primes (e.g., by techniques in [8]). For the latter, the third party will give $\mathcal{O}$ a certificate on n.

2. $\mathcal{O}$ picks an elements $g \in_R QR_n \backslash \{1, -1\}$, where QR_n denotes the subgroup of quadratic residues modulo n. Clearly, $ord(g) = p'q'$. We point out that it is easy to generate QR_n: choose an element $h \in_R Z_n^*$ satisfying $gcd(a \pm 1, n) = 1$ and then $QR_n = \langle h^2 \rangle$, where $\langle h^2 \rangle$ denotes the cyclic group generated by h^2. $\mathcal{O}$ continues to choose other elements g_0, g_1, g_2, g_3, $g_4 \in_R \langle g \rangle$: for each element, $\mathcal{O}$ selects $r_i \in Z_{p'q'}$ and computing $g_i = g^{r_i} \bmod n$. This guarantees that all elements are of the same order $p'q'$.
3. computes a signature $s = g^d \pmod n$ of g.
4. publishes the public system parameters $(P_o, g, s, g_0, g_1, g_2, g_3, g_4)$.

User enrolment

To rule out cheating by issuing organization $\mathcal{O}$, user $\mathcal{U}$ must be assured of that g_0, g_1, g_2, g_3 and g_4 are elements from $\langle g \rangle$ in *user enrolment*. Therefore, the new user enrolment phase works as follows.

1. $\mathcal{O}$ first proves to $\mathcal{U}$ that $g \in QR_n$. This can be achieved by $\mathcal{O}$ sending a square root of g to U. From **Fact 2**, $\mathcal{O}$ can easily compute the square roots of g.
2. $\mathcal{O}$ then convinces $\mathcal{U}$ of the membership of g_0, g_1, g_2, g_3 and g_4 to $\langle g \rangle$. This is readily achieved by O proving knowledge of the respective discrete logarithm of g_0, g_1, g_2, g_3 and g_4 to the base g. Moreover, if n is not certified by a trusted third party, $\mathcal{U}$ requires $\mathcal{O}$ to prove that n is a product of two safe primes by leveraging the techniques in [8].
3. $\mathcal{U}$ sends x_1, x_2, P_u in the clear to $\mathcal{O}$, along with a proof of the knowledge of the discrete logarithm of P_u to g_0.
4. $\mathcal{O}$ upon the verification of the legitimacy of $\mathcal{U}$, chooses $x_3, x_4 \in_R Z_n^*$ to generate a master chameleon certificate (C, S) for $\mathcal{U}$, where (C, S) is in the form of Equations (1) and (2) and such that $C \in QR_n$. It is critical that $C \in QR_n$ in order to guarantee $ord(C) = ord(g)$. Knowing the factorization of n, it is not difficult for $\mathcal{O}$ to decide the quadratic residuosity of C, and further computes its square roots modulo n according to **Fact 2**. Afterwards, $\mathcal{O}$ sends (C, S) together with x_3 and x_4 to $\mathcal{U}$, along with a square root of C proving $C \in QR_n$.

We emphasize again that g_0, g_1, g_2, g_3, g_4 and g, together with C are made to be of the same order in our fix. This is because n is proved to be the product of two safe primes and hence any element of QR_n has an order either $p'q'$ or p' or q'. But the issuing organization dares not publish any g of order p' or q', according to **Fact 1(b)**. As a result, our attack is avoided in the fixed system. It is clear that the fixed system does not incur much efficiency penalty.

As a final note, recall that some commitment schemes (e.g., [12,10]) taking the form of $h_1^r h_2^m \pmod p$ or $h_1^r h_2^m \pmod n$ require $h_2 \in \langle h_1 \rangle$, so as to achieve the "hiding" property of the commitments. Our attack actually explains the reasons for such a requirement. In addition, for the latter case, in a strict sense it should choose $r \in (0, n/4)$ or $r \in (0, n^2)$. This suggests that in *refreshing* and *credential showing* of the fixed system, it is better for $\mathcal{U}$ to pick $x \in_R (0, n/4)$ or $x \in_R (0, n^2)$ in order to construct a slave chameleon certificate.

5 Conclusion

The anonymous credential system proposed in [13] has many attractive features. It supports multi-show credentials and at the same time allows a single credential certificate to be used for any service. The system was claimed to possess *unlinkability*, a central requirement of any anonymous credential system. That is, multiple showings of the same credential certificate can not be linked by the service provider, the issuing organization, or a coalition of the two. We however showed that the issuing organization can easily link the anonymous credentials. This also trivially implied that the system cannot achieve unlinkability against a coalition of the issuing organization and the service providers. We also pointed out the flaw in the original security proof and provided a fix to the system so that it can withstand our attack.

References

1. S. Brands, *Rapid Demonstration of Linear Relations Connected by Boolean Operators*, Proc. Advances in Cryptology, Eurocrypt'97, LNCS 1223, pp. 318-333, 1997.
2. S. Brands, *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privay*, MIT Press, 2000.
3. D. Bressoud and S. Wagon, *A Course in Computational Number Theory*, Key College Publishing, 1999.
4. D. Chaum, *Security Without Identification: Transaction Systems to Make Big Brother Obsolete*, Communications of the ACM, 28(10), pp. 1030-1044, 1985.
5. D. Chaum and J. H. Evertse, *A Secure and Privacy-protection Protocol for Transmitting Personal Information Between Organizations*, Proc. Advances in Cryptology, Crypto'86, LNCS 263, pp. 118-167, 1986.
6. L. Chen, *Acess with Pseudonyms*, Proc. Cryptography: Policy and Algorithms, LNCS 1029, pp. 232-243, 1995.
7. J. Camenisch and A. Lysyanskaya, *An Efficient Non-Transferable Anonymous Multi-Show Credential System with Optional Anonymity Revocation*, Proc. Advances in Cryptology, Eurocrypto'01, LNCS 2656, pp. 93-118, 2001.
8. J. Camenisch and M. Michels, *Proving in Zero-knowledge that a Number is the Product of Two Safe Primes*, Proc. Advances in Cryptology, Eurocrypto'99, LNCS 1592, pp. 107-122, 1999.
9. I. B. Damgard, *Payment Systems and Credential Mechanism with Provable Security Against Abuse by Individuals*, Proc. Advances in Cryptology, Crypto'88, LNCS 403, pp. 328-335, 1988.
10. I. Damgard and E. Fujisaki, *A Statistically-Hiding Integer Commitment Scheme Based on Groups with Hidden Order*, Proc. Asiacrypt'02, LNCS 2501, pp. 125-142, 2002.
11. A. Lysyanskaya, R. Rivest, A. Sahai, and S. Wolf, *Pseudonym Systems*, Proc. Selected Areas in Cryptography, LNCS 1758, pp. 184-199, 1999.
12. T. P. Pedersen, *Non-interactive and Information-Theoretic Secre Verifiable Secret Sharing*, Proc. Crypto'91, LNCS 576, pp. 129-140, 1992.
13. P. Persiano and I. Visconti, *An Anonymous Credential System and A Privacy-Aware PKI*, Proc. ACISP'03, pp. 27-38, 2003.

14. B. Schneier, *Applied Cryptography*, 2^{nd} Edition, John Wiley & Sons, 1996
15. A. de Santis, G. di Grescenzo and P. Persiano, *Communication Efficient Anonymous Group Identification*, Proc. ACM Conference on Computer and Communications Security, pp. 73-82, 1998.
16. A. de Santis, G. di Grescenzo, G. Persiano and M. Yung, *On Monotone Formula Closure of SZK*, Proc. Foundations of Computer Science, FOCS'94, pp. 454-465, 1994.
17. E. Verheul, *Self-Blindable Credential Certificates from the Weil Pairing*, Proc. Advances in Cryptology, Asiacrypto'01, LNCS 2248, pp. 533-551, 2001.

Counting Abuses Using Flexible Off-line Credentials

Kemal Bicakci, Bruno Crispo, and Andrew S. Tanenbaum

Department of Computer Science, Vrije Universiteit
Amsterdam, The Netherlands
{kemal,crispo,ast}@few.vu.nl

Abstract. Mobile and ad-hoc networks allow businesses to provide a new range of applications and services and at the same time they introduce new constraints that have important effects on the way in which security primitives must be designed. This is challenging because it translates to a demand of richer and more flexible security primitives that often need to satisfy stricter requirements than traditional wired network scenarios. In this paper we focus on one of this primitive, namely security credentials. We present a solution that extends the existing protocols used to implement off-line credentials such that, not only abuses can be detected but they can also be counted. Our solution addresses the problem of 1-time and 2-times credentials and we will conclude by discussing the challenges that need to be solved to generalize the primitive to $k = n$.

1 Introduction

With the advent of mobile and ad-hoc networks (MANETs), security researchers face new challenges due to the change of some of the assumptions underlining the security solutions designed for conventional wired networks such as Internet. Global connectivity for example, is replaced by local connectivity, so in MANETs it is not reasonable to assume always the connectivity to a central server or to a trusted third party.

On the other hand, MANETs provide an innovative technology that could potentially enable a wide spectrum of new applications and services. The ability to carry around credentials that can be used at *any time* and at *any place* will increase the opportunities for merchants to sell more services and for customers to buy services following convenient and easy procedures.

Of course, any solution to be acceptable must prevent customers from abusing the use of such credentials (i.e. spending them more times than legitimate, forging them, etc.) and at the same time must prevent merchants from forging credentials and to make them appear as if they were genuinely spent by customers. Besides and not less important, the privacy of customers need to be protected (i.e., merchants cannot record spending habits of customers).

Starting with Chaum [1], researchers proposed several solutions to implement credentials that can be used off-line. Some of those solutions try to detect while

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 548–559, 2005.

others try to prevent possible abuses. Some employ single-use credentials, others allow the same credential to be used many times up to a configurable limit. All the existing schemes however, cannot detect how many times the abuse occurred. This would be a very useful feature to have since existing solutions have to implement fixed penalty schemes that cannot distinguish how many times the customer has abused her credential.

In this paper, we propose a solution called *Flexible Off-Line Credentials* that allows off-line use of credentials and detects not only that the credential has been abused (that means it has been used more than the legitimate k threshold) but also how many times the abuse has occurred. We propose and analyze solutions for the case $k = 1$ and $k = 2$ and then we discuss why the extension to $k = n$ would require further work. Our solution allows fair fine-based schemes that punish abusers proportionally to their abuse. Our scheme can also be combined with a credit-based scheme to provide a single flexible mechanism that offers the advantages of both debit and credit-based solutions without the usability limitations typically of the debit-based approach.

The rest of this paper is organized as follows. Next section summarizes the related work. Section 3 gives the preliminaries for our work. Section 4 explains our protocol for flexible one-time credentials. Section 5 extends this protocol for credentials that can be used 2-times. Section 6 is for the security analysis of proposed protocols. Section 7 concludes the paper.

2 Related Work

The problem of implementing private credentials that cannot be abused in the sense we described above can be easily solved if the system can rely upon a central authority that both issues credentials and later monitors and stores in a database all these credentials as they are used [2]. So when a customer presents her credential to a merchant, the merchant can check with the authority if that credential is still valid and can be accepted, preventing any abuse. Such approach cannot be used with MANETs because the access to the database cannot be guaranteed.

Alternatively, Chaum [3] proposes an off-line method that uses special hardware trusted by the issuer of the credentials that prevents customers to abuse the use of credentials. Such approach however, cannot be used in many contexts where deploying such hardware is simply too expensive.

Credit-based solutions can be also used off-line. Think for instance credit cards. To be able to buy a service from a merchant, customers present an authentic nonrepudiable piece of information (signed credit card voucher) together with her credit card. The problem with this approach is that abuses of credentials may not be promptly detected if an online server is not available. Besides, this solution does not provide anonymity at all since the credit card information given to the merchant is bound to the identity of the customer to allow redemption.

Debit-based solutions use credentials that can be used off-line, too (similar to paper cash). The customer has to decide beforehand how many times she

is going to use the credentials and she pays in advance for their use at the time they are issued rather than when (and if) they are used. After payment she is given credentials which can be used anonymously like spending paper cash in physical world. However unlike paper cash, in digital world credentials are simply a sequence of bits and can easily be copied and spent at different merchants illegitimately.

Some of the solutions prevent these abuses while others aim at detecting after they occur. Both approaches could be suitable depending on the application. In detection based solutions, the user's anonymity is guaranteed if she follows the rules of the game but once she tries to cheat, her identity is learned and her abuse is proved to redeem the losses. Chaum [1] was the first to introduce a solution of this type applied to digital cash as credentials. Of course the aim there was to prevent double spending. Brands [4] proposes a more efficient solution and also extends Chaum's work to the case of credentials that can be used k-times and detects if this threshold is abused by the customer. Another notable example of k-times credentials is due to Camenisch and Lysyanskaya [5]. In a recent work, Bussard and Molva [6] have revisited the idea of offline credentials for MANETs scenarios. They propose a solution where the customer is deterred from abusing the credential by the threat of a fixed fine if an abuse is detected. However all of these solutions cannot distinguish if the credential has been abused once, ten times or one hundred times leading to possible unfair penalties.

3 Preliminaries

3.1 One-Time Signatures

We will summarize the operation of one-time signature (OTS) construction proposed by Lamport [7] as follows:

Suppose the message to be signed has a length of b bits. The signer generates $2b$ random numbers of size sufficiently long (so called pre-images). He then computes the hash of each random number as the hash images which constitutes the one-time public key. Now there are two random numbers associated for each bit of the message to be signed. If the value of the bit is 0, the signer reveals the first of these random numbers, if it is 1, he reveals the other one. The whole bunch of revealed random numbers constitutes the OTS, which can easily be verified using the one-time public key.

Lamport's scheme is severely unoptimized hence there are alot of previous work for optimization (e.g. [8]). But we do not explain them here since they are not relevant to our purposes in this paper.

3.2 Blind Signatures

Blind signatures, first introduced by Chaum [9], allow someone to get a message signed by some other party without revealing any information about the message to him. Blind signatures can easily be implemented using RSA since it has the multiplication commutative property.

3.3 Cut-and-Choose Method

The basic idea is as follows. Suppose Alice wants Bob to sign n blinded messages and Bob needs to ensure these messages have some appropriate format. Now, Alice does not prepare n blinded messages but $n + m$ of them. Upon receipt of these blinded messages, Bob chooses a random subset having m elements and requests Alice to unblind them. If these unblinded messages have the appropriate format, it is highly probable that Alice has also prepared appropriately the blinded remaining of n messages. Then, Bob signs them and sends back to Alice. In other words, Bob *cut*s m messages and *choose*s the remaining n messages to sign. Alice can then perform the unblinding and use the signatures as desired.

Note that there is only a probabilistic guarantee that the signed messages have the appropriate format. However the cheating probability can be made negligible by choosing appropriate m and n values.

4 The Proposed Protocol

In this section, we introduce our protocol when the credentials are valid for one-time. We first introduce the presumed environment where the proposed protocol is to be employed. Then, we describe initialization and operation of the protocol and illustrate how the exact number of multiple spending can be detected. Finally, we show how this feature can be used for seamless conversion from debit to credit based solutions.

4.1 System Model and Assumptions

To make the model more concrete, consider the following example. Let us suppose that Cinderalla Airline (CI) signed an agreement with Aladdin Airport (AA) such that every customer of CI is entitled of a discount of 5 dollars she can use at any shop of the airport upon presentation of her boarding card.

In our example, customer buys, following the existing procedures, the flight ticket from CI. At the time of check-in, CI (as the Credential Issuer) issues to the customer a boarding card carrying the one-time credential. The credentials were redeemed by the shops at AA not at the CI. AA (as the Authorization Authority) then checks whether there is any illegitimate spending of the same credential (twice or more). In case of abuse it can ask CI for the abuser details collected when she bought the ticket.

To generalize, each shop inside AA are the *servers* and all CI's customers are the potential *users*. We assume there is no continuous on-line connectivity between AA and servers but they can exchange some data periodically (e.g. weekly or daily basis). AA and CI can also have a periodic communication possibly less frequently (e.g. monthly basis).

After getting their one-time credentials, the users do not have any connection to CI. Their only interaction is with the servers to request for the service.

In this model as a requirement the privacy of users is guaranteed if they have used their credential legitimately only once. The key issue is assuring that users

use their one-time credentials only once but not more. Our solution is based on the postponed punishment principle as in [6] and most offline e-cash schemes. While obtaining the credential, every user also signs a contract stating that her identity will be learned and some money will be charged if CI can later prove that the user has used the one-time credential more than once.

Unlike previous work, using our protocol, CI (and AA) can determine the exact number of multiple uses therefore while preparing the contract it becomes possible to set the amount of penalty as a function of number of multiple uses (e.g. 5 euros for double spending, 10 euros for triple spending and so on).

The final assumption we make in designing the protocol is that initially AA sets the maximum number of servers that it can support and distributes this piece of information to CI. This assumption seems reasonable because it does not imply that servers can not dynamically registers to or leaves from AA. The only constraint is not to have more servers than the fixed maximum number.

4.2 Initialization

To start with, AA assigns identification numbers (IDs) to registered servers in a way unique to our protocol. The IDs are expressed as binary vectors having zeros in all coordinates except one and the single '1' should appear in a different coordinate for all the vectors. Let us give an example to illustrate the concept:

Example: Suppose the maximum number of servers is set up to be 5 and number of servers registered is currently 3. One way of assigning IDs is as follows:

00001 to 1st server
00010 to 2nd server
00100 to 3rd server

While the servers are registered, AA also sends securely CI's public key so that servers can verify the signatures. After IDs are assigned, to get a one-time credential from CI, the user executes the following steps:

1. The user learns the maximum number of registered servers. Suppose the maximum is b.

2. The user prepares $4b$ random numbers. She then divides these random numbers into two groups having 2b elements each.
   Example: If the maximum limit is 5, the user prepares 20 random numbers.
   Group 1: $x_1, x_2, x_3, ..., x_{10}$
   Group 2: $y_1, y_2, y_3, ..., y_{10}$

3. The user calculates the hash of each of these random numbers.

4. From both groups, the user retrieves the hashes having the same sequence number ($h(x_i)$ and $h(y_i)$) and concatenates them. She then generates random numbers (r_i)'s and prepares 2b blinded messages as follows:

$$B_i = (h(x_i) \parallel h(y_i)) * (r_i)^e \bmod n$$

5. The user also calculates the hash of concatenation of two random numbers having the same sequence number i.e. $C_i = h(x_i \parallel y_i)$. We refer C_i's as $conca - hashes$.

6. The user sends B_i's and C_i's to CI. Blinding is not needed for conca-hashes.

7. For a successful operation, the user should prepare the hash images and do the encoding of all conca-hashes (C_i's) honestly as described above. To verify this, CI uses the cut-and-choose method. More precisely, for a random subset having b elements out of $2b$, CI asks the user to prove that the hash images are correctly prepared and conca-hashes are correctly encoded.

8. As a proof, the user reveals random numbers (r_i)'s and pre-images (x_i and y_i)'s for the subset CI asked for.

9. CI verifies the subset of blinded messages (B_i)'s and conca-hashes (C_i)'s. It then signs the remaining subset of blinded messages as follows (to simplify notation, let's assume the remaining subset has sequence numbers from 1 to b):
$$S = \prod_{1 \le i \le b} (B_i)^d \bmod n$$

10. The user first performs the unblinding (by dividing S with $\prod_{1 \le i \le b} r_i$) and then reindexes lexicographically the rest of hash-images as follows:
$$(h(x_1) \parallel h(y_1)) < (h(x_2) \parallel h(y_2)) < ... < (h(x_b) \parallel h(y_b))$$
$$P = \prod_{1 \le i \le b} (h(x_i) \parallel h(y_i))$$
P constitutes the one-time public key certified by CI's signature. The signature on the one-time public key is the one-time credential issued by CI.
Example: When the number of server is limited to a maximum of 5, the one-time credential has the following format:
One-time Credential $= Signature_{CI}[(h(x_1) \parallel h(y_1)) * (h(x_2) \parallel h(y_2)) * (h(x_3) \parallel h(y_3)) * (h(x_4) \parallel h(y_4)) * (h(x_5) \parallel h(y_5))]$

11. In order to establish the penalty for multiple use, the user and CI prepares a contract. In the contract, the conca-hashes are listed after the re-indexing. Again we illustrate this idea with an example:
Example Penalty Contract: This contract is prepared between CI and user Alice. The conca-hashes are as follows: C_1, C_2, C_3, C_4, C_5. By signing this contract, user Alice accepts to pay 5 euros for the first two conca-hashes CI will be able to show its encoding (reveals x_i and y_i such that $C_i = h(x_i \parallel y_i)$) and 5 euros for the encoding of each extra conca-hash.
We will explain at the end of this section why we require two conca-hashes for the first pay.

4.3 Operation

In the initialization, the user gets her one-time credential from CI. We now want to show how she can use this credential to get a service from a registered server:

1. The user learns the ID of the server she would like to get a service from.

2. The user sends the server her one-time credential together with her one-time public key. She then signs the ID of the server.
   Example: Suppose the server's ID is 00001. The user signs this ID by revealing:
   $x_1, x_2, x_3, x_4, h(x_5)$ and $h(y_1), h(y_2), h(y_3), h(y_4), y_5$.
   (She reveals x_i and $h(y_i)$ if the i-th bit value is 0 and y_i and $h(x_i)$ if it is 1).

3. The server verifies one-time credential using CI's public key, checks the lexicographic order of hash images (i.e. $(h(x_1) \parallel h(y_1)) < (h(x_2) \parallel h(y_2)) < ... < (h(x_5) \parallel h(y_5))$ and verifies one-time signature using one-time public key of the user. The server also stores the credential and the signature.

4.4 Detecting the Number of Multiple Uses

When the user wants to issue the same one-time credential to the same server, this can easily be detected and avoided by the server (To allow the user to spend her credential to the same server more than once by accepting to pay the penalty, we can assign more than one ID to the server.). The more critical cheating is the one where the user issues her credential to a different server. This can not be detected by the server instantly but later AA can detect it after collecting all the one-time credentials from the servers.

Example: Besides server 1, suppose the user reuses her one-time credential also to server 2 with the ID of 00010. He should sign the server's ID by revealing $x_1, x_2, x_3, h(x_4), x_5$ and $h(y_1), h(y_2), h(y_3), y_4, h(y_5)$.

At the end of the day, AA can easily detect that the one-time credential has been used twice because the same pre-images have appeared in two different signatures. AA then asks CI for the redemption by sending both signatures.

After receiving two one-time signatures from AA, CI can show the encoding of C_4 and C_5 since both x_4 and y_4 as well as x_5 and y_5 are obtained as parts of signatures. Due to the signed contract, showing the encoding of C_4 and C_5 would let CI to get the first penalty pay of 5 euros.

Note that the identity of the cheater is written on the signed contract and learned by CI. However depending on the security policy it might be preferred to protect her privacy from AA and other third parties.

AA can detect and CI can prove not only double spending but also subsequent spendings by the same logic.

Example: After spending it to server 1 and 2, suppose now before the day ends the user reuses her one-time credential to server 3 with the ID of 00100. After collecting the one-time signatures, CI can also show the encoding of C_3 and therefore the user should pay an extra amount of 5 euros.

4.5 Seamless Conversion from Debit to Credit-Based Scheme

Since our protocol can count the number of times credentials are being used, it provides the possibility to convert it from a debit to a credit scheme without additional requirements. Mechanisms that cannot quantify abuses clearly have to detect such abuses with the aim of preventing them. In our case, this is not necessarily true and the capability of using the credentials even more than a pre-established threshold can be useful provided that there is an exact counting. A practical example where this can be useful is the following. Let us consider the case of metro tickets valid for one trip. The holder of the ticket, because he is in a hurry, does not have coins, or the ticket office is closed cannot buy a new ticket. In this situation, using the protocol described, he can validate the old ticket to have access to the train. This will be detected later and he will be charged with a premium. This can also occur more than once, maybe with a premium that increases with the number of times. Many customers will be happy to pay a bit more than the normal price, but much less than the fine, for this flexibility. Thus, the same protocol can be converted from a debit to credit based one. Of course the underlying assumption here, as in any credit-base scheme, is that the system has access to the customer details to credit for the premium.

5 Extension for 2-Times Credentials

For one-time credentials, the server IDs are binary vectors and there are two random numbers committed for each coordinate of the ID space. The protocol works as desired because the IDs are assigned in a way that forces the user to reveal both of the random numbers for a fixed number of coordinates in case of multiple spending. Based on these observations, below we list the requirements for 2-times credentials:

1. The server IDs should be 3-ary vectors and there should be 3 random numbers committed for each coordinate.
2. When the credential is used for 3 times, all of the random numbers should be revealed for a deterministic number of coordinates (preferably only one coordinate for convenience).
3. When the credential is used for more than 3 times, the number of coordinates for which all the random numbers are revealed should be a one-to-one function of how many times the credential has been used.

As the final requirement, all these should be satisfied with the minimum length of server IDs because of efficiency reasons.

5.1 An ID Assignment Algorithm for 2-Times Credentials

To satisfy the requirements listed above, the most critical issue is the assignment of server IDs. Pseudocode of an algorithm for this purpose is as follows:

```
Input(max) /* read the maximum limit for the number of servers */
for i = 0 to max-1 do begin
    for j = 0 to max-3 do begin /* IDs have length of max-2 each */
        server[i][j]:='e' /* 'e' means empty */
    end
end
for i1 = 0 to max-3 do begin
    for i2 = i1+1 to max-2; do begin
        for i3 = i2+1 to max-1 do begin
            for j = 0 to max-3 do begin
                if server[i1][j]=='e') then server[i1][j]:='0'
                if server[i2][j]=='e') then server[i2][j]:='1'
                if server[i3][j]=='e') then server[i3][j]:='2'
                if server[i1][j] ≠ server[i2][j] ≠ server[i3][j] then break
            end
        end
    end
end
```

The rationale of this algorithm is to consider every subset of combinations with three elements and assign a different ternary value of each element in every subset for only one coordinate (while keeping the number of coordinates required to a minimum). As an example, let us consider the case of maximum number of servers set to be 7 as shown in Table 1. Note that each of the 5 coordinates is assigned a ternary value (a value in the range of 0 to 2). You can experiment with this table by trying different combinations of spendings to confirm that this ID assignment satisfies the requirements listed above i.e. when the credential is spent for three times all random numbers are revealed for only one coordinate. Additionally, for subsequent spendings all the random numbers are revealed for exactly one more coordinate.

coordinate	#5	#4	#3	#2	#1
server #1	0	0	0	0	0
server #2	0	0	0	0	1
server #3	0	0	0	1	2
server #4	0	0	1	2	2
server #5	0	1	2	2	2
server #6	1	2	2	2	2
server #7	2	2	2	2	2

Table 1. Assignment of Server IDs for 2-times credentials when max=7.

5.2 Initialization of the Protocol for 2-Times Credentials

After IDs are assigned, similar to earlier case the following steps are executed:

1. The user learns the length of server IDs (which is equal to the maximum number of registered servers minus two). Suppose the length is b.
2. The user prepares $6b$ random numbers and divides them into three groups. The user also calculates the hash of these random numbers.
3. The user concatenates three hashes and prepares $2b$ blinded messages:

$$B_i = (h(x_i) \parallel h(y_i) \parallel h(z_i)) * (r_i)^d \bmod n$$

4. The conca-hashes the user prepares also has three random numbers as the input: $C_i = h(x_i \parallel y_i \parallel z_i)$
5. The rest of the initialization strictly follows the earlier case. Instead of two, three random numbers are committed for each coordinate.

5.3 Using 2-Times Credentials

In one-time credentials, the coordinates of server IDs have binary values therefore per each coordinate depending on its value the user should reveal one of two random numbers and the other's hash value. Now since the coordinates have ternary values the user should reveal one of three random numbers and the other two hash values. An example is as follows:

Example: Suppose the server's ID is 00012 (server #3). The user signs this ID by revealing $x_1, x_2, x_3, h(x_4), h(x_5)$ and $h(y_1), h(y_2), h(y_3), y_4, h(y_5)$ and $h(z_1), h(z_2), h(z_3), h(z_4), z_5$.

5.4 Remarks

Our solution for 2-times credentials has some limitations:

- When the credential is valid for more than one-time, in order to allow the user to spend it legitimately to the same server; we should assign two IDs instead of one to each server. This makes the protocol less efficient due to the increase in the ID space required.
- Modifying the ID assignment algorithm for three-times credentials is possible. However as the number of server increases, three-times credentials rapidly become very inefficient. This is due to the nonlinear increase in the size of ID space required.
- One nice feature of the protocols proposed in this paper is the preservation of user privacy. Even the credential issuer cannot link the legitimate spendings to the identity of users. There is a stronger requirement than privacy called **unlinkability** that means inability to link one spending with another one (when both of them are anonymous). The extension for 2-times credentials described in this section does not provide unlinkability.

6 Security Analysis

In the following, we consider the basic security requirements imposed on an off-line credential. We also show that our protocol satisfies the requirements, therefore it is secure.

6.1 Unforgeability

For a secure operation, the credential should not be forged. To forge it, an adversary can attempt to:

- forge CI's public key signature
- find pre-image(s) such that the hash of it is equal to the one of the hash images CI has certified.

Clearly the first attack is not specific to our protocol. The second attack is infeasible if the hash function is pre-image resistant.

6.2 Proving Multiple Use

In subsection 4.4, we have shown how overspending can be proven by CI. However there is a condition for this proof. If the encoding of conca-hashes was not performed as described, the proof fails. Just like earlier work (e.g., [1,6]) our protocol offers only probabilistic guarantees because of using cut-and-choose method. Let us show the calculation of probability of an undetected cheating as a function of protocol parameter b (length of server IDs).

In step 7 of the initialization phase (subsection 4.2), CI chooses b out of $2b$ conca-hashes. There are $C(2b, b)$ subsets in total that CI can choose from (C denotes combination where $C(2b, b) = (2b)!/b!b!$).

Cheating by the user would not be detected only if the user truly did the encoding of only b out of $2b$ conca-hashes and CI has chosen exactly this single subset. The probability of choosing this subset is $p(c) = \frac{1}{C(2b,b)}$.

For instance this probability is approximately equal to 0.004 if $b = 5$. As b increases, the probability of undetected cheating decreases.

6.3 Colluding Cheaters

If, instead of ours, the original protocol proposed by Chaum et al. [1] is used, after the transaction with server 1, Alice can tell the transaction to another server and the second server can send to AA the same information as server 1 for redemption. AA can understand there is double spending but cannot trace which server is the cheater and also cannot trace the identity of Alice. Since in our protocol each server's challenge is fixed, this kind of attack is not possible. Similarly, security is guaranteed in our protocols even when there is a collision between servers, between CI and AA, or between any combinations against any party in the system.

7 Conclusion and Future Work

We described a new approach to privacy-preserving limited-time off-line credentials. Our approach uses the postponed punishment principle just like earlier work but it also has the unique feature of detecting exactly how many times the user has overspent her credential. This gives us the ability to have a single framework incorporating both debit-based and credit-based solutions as well as the flexibility to dynamically adjust the penalty for overspending.

Our protocol is clearly not appropriate for global Internet, it is rather designed for pervasive environments where number of servers is limited.

As a future work, it is promising to explore possibilities to have a more efficient extension of our protocol to the general case of k-times credentials.

Acknowledgment: We thank to members of Security Group in Vrije Universiteit for their support.

References

1. D. Chaum, A. Fiat, M. Naor: Untraceable electronic cash. *In Proc. of CRYPTO'88 (LNCS 403)*, pages 319-327. Springer-Verlag, 1990.
2. D. Chaum: Online cash checks. *In Proc. of EUROCRYPT'89 (LNCS 434)*, pages 288-293. Springer-Verlag, 1990.
3. D. Chaum, T.P. Pedersen: Wallet Databases with Observers. *In Proc. of CRYPTO'92 (LNCS 740)*, pages 89-105. Springer-Verlag, 1993.
4. S. Brands: A technical overview of digital credentials. Research Report, February 2002.
5. J. Camenisch, A. Lysyanskaya: An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation. *In Proc. of EUROCRYPT 2001 (LNCS 2045)*, pages 93-118, Springer-Verlag, 2001.
6. L. Bussard, R. Molva: One-Time Capabilities for Authorizations without Trust. *In Proc. of Second IEEE International Conference on Pervasive Computing and Communications(PerCom 2004)*, Orlando, Florida, March 14-17, 2004.
7. L. Lamport: Constructing digital signatures from a one-way function. Technical report SRI-CS-98. SRI International Computer Science Laboratory, October 1979.
8. K. Bicakci, G. Tsudik, B. Tung: How to construct optimal one-time signatures. *Computer Networks (Elsevier)*, Vol.43(3), pp. 339-349, October 2003.
9. D. Chaum: Blind signatures for untraceable payments. *In Proc. of CRYPTO'82*, pages 199-203.

Cryptanalysis of Two Variants of PCBC Mode When Used for Message Integrity

Chris J. Mitchell

Information Security Group, Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK
http://www.isg.rhul.ac.uk/~ cjm

Abstract. The PCBC block cipher mode of operation has many variants, of which one, due to Meyer and Matyas, dates back over 20 years. Whilst a particularly simple variant of PCBC has long been known to be very weak when used for data integrity protection, the Meyer-Matyas variant has not previously been attacked. In this paper we cryptanalyse this mode, and show that it possesses a serious weakness when used for data integrity protection. Specifically, we show how to construct an existential forgery using only a single known ciphertext message and a modest amount of known plaintext (this could be as little as three plaintext blocks). We also describe a ciphertext-only existential forgery attack against another, recently proposed, PCBC-variant called M-PCBC.

1 Introduction

Traditionally, the recommended way to use a block cipher to provide both integrity and confidentiality protection for a message has been to compute a CBC-MAC and also encrypt the data, using two distinct secret keys. This approach is rather unattractive for some applications because it requires each block of data to be processed twice. This observation has given rise to a number of proposals for combining encryption and integrity protection, including a particular way of using the so called PCBC mode (see, for example, Section 9.6 of [1]).

At the same time, two major problems have recently been identified which have highlighted the need for better-defined integrity and confidentiality modes. Firstly, issues have been identified with certain combinations of encryption and use of a CBC-MAC — see, for example, Bellare, Kohno and Namprempre [2]. That is, it is vital to define precisely how the two operations are combined, including the order of the computations; otherwise there is a danger of possible compromise of the data. Secondly, even where integrity is not explicitly required by the application, if integrity is not provided then in some cases padding oracle attacks may be used to compromise secret data (see, for example, [3,4,5,6]).

This has given rise to a number of proposals for well-defined authenticated-encryption modes, including OCB [7], EAX [8] and CCM [9,10]. These techniques are also the subject of ongoing international standardisation efforts — the first working draft of what is intended to become ISO/IEC 19772 on authenticated encryption was published early in 2004 [11] (see also Dent and Mitchell, [12]).

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 560–571, 2005.

OCB is to some extent analogous to the use of PCBC to provide integrity, in that it involves only a single pass through the data; unlike PCBC, it also possesses a proof of security.

In this paper we examine two different PCBC variants. We first examine a variant, identified below as PCBC+, which was one of the first ever proposals for a block cipher mode of operation designed to provide both integrity and confidentiality protection. We show that this variant is subject to a known plaintext attack, and hence does not provide adequate integrity protection. We go on to examine a newly proposed variant called M-PCBC [13] which is also claimed to provide both integrity and confidentiality protection when used appropriately. Unfortunately, as shown below, this claim is not correct.

It is important to note that the term PCBC is used by different authors to means slightly different things, and hence we first describe what we mean by PCBC. This is followed by analyses of PCBC+ and M-PCBC.

2 PCBC Mode and Variants

The precise origin and definition of PCBC, is unclear. In fact, the acronym PCBC has been used to mean two different things, and we define them both.

2.1 The More General Definition

Section 9.6 of [1] (Example 9.91) defines the *Plaintext-Ciphertext Block Chaining* mode of operation as follows (note that a special case of this definition goes back at least to 1982, since it is contained in the third and subsequent printings of Meyer and Matyas's 1982 book [14]).

First suppose that the data is to be protected using an n-bit block cipher, i.e. a block cipher operating on plaintext and ciphertext blocks of n bits. We write $e_K(P)$ for the result of block cipher encrypting n-bit block P using the secret key K, and $d_K(C)$ for the result of block cipher decrypting the n-bit block C using the key K. Suppose the plaintext to be protected is divided into a sequence of n-bit blocks (if necessary, first having been padded): $P_1, P_2, \ldots, P_t$.

Then, if the n-bit Initialisation Vector (IV) is S, the PCBC encryption of the plaintext $P_1, P_2, \ldots, P_t$ is defined as:

$$C_i = e_K(P_i \oplus G_{i-1}), \;\; 1 \leq i \leq t,$$

where $G_0 = S$, $G_i = g(P_i, C_i)$, $1 \leq i \leq t$, and g is a simple function that maps a pair of n-bit blocks to a single n-bit block.

Menezes, Van Oorschot and Vanstone [1] make two remarks regarding the choice of g. Firstly they suggest the use of $g(P, C) = P + C \bmod 2^n$, where the n-bit blocks P and C are treated as integers by regarding them as binary representations, and the modulo 2^n sum is converted back to an n-bit block by taking the binary representation (left-padded as necessary with zeros). Secondly they suggest that g should not be equal to the bit-wise exclusive-or of the two

inputs when the mode is to be used to protect the integrity of data (a precaution which they point out is necessary to avoid a known-plaintext attack).

The first choice for g listed above is the technique that is also described by Meyer and Matyas [14]. For convenience we call this mode PCBC+. It is this mode that we consider in detail in this paper. Despite the fact that it has been included in two well known books on cryptography, there would not appear to be any literature at all discussing the security of PCBC+.

We should, at this point, explain how PCBC+ mode (or any other variant of PCBC) can be used to provide both encryption and integrity-protection. The idea is very simple. First divide the data to be encrypted into a sequence of n-bit blocks, padding as necessary. Then append an additional n-bit block to the end of the message, where this block can be predicted by the decrypter (e.g. a fixed block). When the message is decrypted, a check is made that the final block is the expected value and, if it is, then the message is deemed authentic.

Before proceeding observe that this general approach possesses an intrinsic weakness. That is, suppose that a fixed final block (the *terminator block*) is used to detect message manipulations (as above). Then an attacker might be able to persuade the legitimate originator of protected messages to encrypt a message which contains the fixed terminator block somewhere in the middle of the message. The attacker will then be able to delete all ciphertext blocks following the encrypted terminator block, and such a change will not be detectable. Despite this weakness, using an appropriate encryption mode combined with a method for adding verifiable redundancy to a message is still used for message integrity protection — e.g. in Kerberos (see, for example, [12]). As far as this paper is concerned we note that such an attack could be prevented by ensuring that the legitimate encrypter refuses to encrypt any plaintext message containing the terminator block. We further note that such an attack requires chosen plaintext, and the attacks we demonstrate later in this paper require only either a limited amount of known plaintext, or just known ciphertext.

2.2 The More Specific Definition

PCBC is also sometimes defined [15,16] to mean *Plaintext Cipher Block Chaining*. In this case PCBC mode is a special case of PCBC as defined above, i.e. where $g(P, C) = P \oplus C$ and where $\oplus$ represents the bit-wise exclusive-or of the blocks P and C. To avoid any confusion we refer to the version of PCBC defined in Menezes et al. [1], and Section 2.1 above, as G-PCBC (for *Generalised PCBC*), and the specific case where g is equal to exclusive-or simply as PCBC.

PCBC in this more specific sense was used in Kerberos version 4 [16] to provide encryption and integrity protection. This was achieved by the means described above, i.e. by checking the final decrypted block of the message.

It is important to observe that PCBC is precisely the version of G-PCBC that Menezes, van Oorschot and Vanstone [1] state should not be used to protect the integrity of data! The weakness of PCBC for use as an integrity-protection mode was first pointed out by Kohl [15]. As is simple to verify, Kohl pointed out that if two of the ciphertext blocks of a PCBC-encrypted message are interchanged,

then this does not affect the decryption of the final block, i.e. it is extremely simple to make undetectable changes to messages. Note that this is actually a stronger attack than is implied by [1] who refer only to the danger of known-plaintext attacks. Finally note that yet another variant of PCBC was proposed by Gligor and Donescu [17]; however, this scheme, known as iaPCBC, was shown to possess serious vulnerabilities by Ferguson et al. [18].

3 IV Management Strategy for G-PCBC

In the definition of G-PCBC in Section 2.1, the encrypter and decrypter are required to have access to the same n-bit IV, S. However, it is not completely clear from the discussions in [1,14] how this is meant to be achieved, although it is clear that S should be different for each message. One possible approach is for the sender to choose S and send it in plaintext with the encrypted message. In such a case S might either be chosen at random or generated by a simple counter. In the latter case S will potentially be predictable to an intercepting attacker, and we next point out that this would be a potentially dangerous option, regardless of the choice of g.

Suppose $C_1, C_2, \ldots, C_s$ are the first s blocks of an intercepted ciphertext message, encrypted using PCBC+, for which the attacker knows the plaintext P_s (corresponding to C_s). Suppose also that message integrity is protected by appending the fixed terminator block P^* to the end of the message prior to encryption. If we further suppose that the attacker has access to a chosen plaintext encryption oracle (a strong assumption, admittedly), then the attacker submits for encryption a message with first block $S \oplus P^* \oplus g(P_s, C_s)$, where S is the 'predicted' IV to be used by the oracle. The first block of the encrypted message will be $C^* = e_K(P^* \oplus g(P_s, C_s))$. The attacker now knows that the ciphertext message $C_1, C_2, \ldots, C_s, C^*$ will be accepted as genuine by the legitimate decrypter, since it is easy to check that the decryption of C^* will yield P^*.

For the remainder of this paper we therefore assume that the IV S is unknown to any attacker — e.g. as would be the case if it is chosen by the sender and sent in encrypted form with the encrypted message. This will nevertheless enable an attacker to force the decrypter to re-use an IV employed to encrypt an intercepted message, without knowing what the value of the IV is — we (implicitly) assume that this attack model applies in the remainder of this paper.

4 An Existential Forgery Attack on PCBC+

We now describe an attack which requires one known ciphertext message, some partial known plaintext corresponding to this ciphertext, and computation with complexity of the order of $2^{n/2}$ operations, where each operation is very simple (much simpler than an encryption operation). Here, as above, n is used to denote the plaintext/ciphertext block length. We also use $\boxplus$ to denote addition modulo 2^n; similarly, $\boxminus$ denotes subtraction modulo 2^n. We first make a trivial observation, whose proof follows immediately from the definition of PCBC+.

Observation 1 *Suppose an attacker knows $C_1, C_2, \ldots, C_t$, a PCBC+ ciphertext message (where C_i is an n-bit block for every i). Suppose also that the attacker knows two consecutive plaintext blocks corresponding to this message: (P_{s-1}, P_s) say, where $1 < s \leq t$. Then the attacker can compute $d_K(C_s)$ as*

$$d_K(C_s) = P_s \oplus (P_{s-1} \boxplus C_{s-1}).$$

We can now give our main result.

Theorem 1. *Suppose, for some $r \geq 2$, an attacker knows r pairs of blocks:*

$$\{(B_i, D_i) \,:\, 1 \leq i \leq r,\ D_i = d_K(B_i)\},$$

where K is a key used to compute one or more PCBC+ ciphertexts. (Such a set can be obtained using r pairs of consecutive known plaintext blocks — see Observation 1). Suppose also that $C_1, C_2, \ldots, C_t$ is a PCBC+ encrypted version of the message $P_1, P_2, \ldots, P_t$, where the final plaintext block P_t is equal to a fixed pattern P^* used to detect changes in the ciphertext and thereby guarantee message integrity. Further suppose that the attacker knows plaintext block P_s for some s satisfying $1 \leq s < t$.

Then, if the integer sequence (u_1, u_2,..., u_w), $w \geq 1$, $1 \leq u_i \leq r$, satisfies

$$E_i = (E_{i-1} \oplus D_{u_i}) \boxplus B_{u_i},\ 1 \leq i \leq w$$

where $E_0 = E_w = P_s \boxplus C_s$, then the PCBC+ decrypted version of the ciphertext

$$C_1, C_2, \ldots, C_s, B_{u_1}, B_{u_2}, \ldots, B_{u_w}, C_{s+1}, C_{s+2}, \ldots, C_t$$

is equal to

$$P_1, P_2, \ldots, P_s, E_1 \boxminus B_{u_1}, E_2 \boxminus B_{u_2}, \ldots, E_w \boxminus B_{u_w}, P_{s+1}, P_{s+2}, \ldots, P_t.$$

That is, the modified message is an existential forgery, since the final recovered plaintext block is P^*.

Proof. By definition, the decryption of $C_1, C_2, \ldots, C_s$ will clearly yield the first s plaintext blocks $P_1, P_2, \ldots, P_s$. Next consider the decryption of B_{u_1}. By definition of PCBC+, the recovered plaintext block will equal

$$d_K(B_{u_1}) \oplus (C_s \boxplus P_s) = D_{u_1} \oplus E_0 = E_1 \boxminus B_{u_1}$$

as required. Working inductively, the decrypted version of B_{u_i} $(1 < i \leq w)$ is equal to

$$d_K(B_{u_i}) \oplus (B_{u_{i-1}} \boxplus (E_{i-1} \boxminus B_{u_{i-1}})) = D_{u_i} \oplus E_{i-1} = E_i \boxminus B_{u_i},$$

again as required. The decrypted version of C_{s+1} is equal to

$$\begin{aligned} d_K(c_{s+1}) \oplus (B_{u_w} \boxplus (E_w \boxminus B_{u_w})) &= d_K(C_{s+1}) \oplus E_w \\ &= d_K(C_{s+1}) \oplus (P_s \boxplus C_s) \\ &= P_{s+1}, \end{aligned}$$

and the result follows immediately. □

Thus to find an existential forgery we simply need to find a sequence of positive integers $(u_1, u_2, \ldots, u_w)$, $w \geq 1$, $1 \leq u_i \leq r$, with the property that:

(i) $E_0 = E_w = P_s \boxplus C_s$, and
(ii) $E_i = (E_{i-1} \oplus D_{u_i}) \boxplus B_{u_i}$, $1 \leq i \leq w$.

Such a sequence can be constructed using the 'standard' Birthday Paradox arguments (see, for example, Section 2.1.5 of [1]). The procedure is as follows.

First choose a positive integer v such that $\lfloor r^v \rfloor = 2^{n/2}$. For example, if $n = 64$ and $r = 4$ then $v = 16$. Then generate all r^v possible sequences $(u_1, u_2, \ldots, u_v)$, $1 \leq u_i \leq r$, and for each such sequence compute $E_0, E_1, \ldots, E_v$ using the equations $E_0 = P_s \boxplus C_s$ and $E_i = (E_{i-1} \oplus D_{u_i}) \boxplus B_{u_i}$, $1 \leq i \leq v$. Sort and store all the E_v values.

Now repeat the same process working 'backwards' from $P_s \boxplus C_s$. That is, generate all r^v possible sequences $(u_1, u_2, \ldots, u_v)$, $1 \leq u_i \leq r$, and for each such sequence compute $F_0, F_1, \ldots, F_v$ using the equations $F_v = P_s \boxplus C_s$ and $F_{i-1} = (F_i \boxminus B_{u_i}) \oplus D_{u_i}$, $1 \leq i \leq v$. If any of the values F_0 equal any of the E_v values then the corresponding two sequences can be concatenated to yield a sequence with the desired properties. Because of the choice of the parameter v, there is a good chance of such a match occurring. The attack is now complete.

5 Remarks on the Attack on PCBC+

We now analyse the existential forgery attack on PCBC+ presented in Section 4.

5.1 Effectiveness of the Attack

The discussion above of the use of a Birthday Paradox search makes the implicit assumption that the values of E_v and F_0 will be randomly distributed across the range of all possible n-bit values. This is clearly not a completely sound assumption, since there is no rigorous evidence that recursively adding and then ex-oring pairs of values from a fixed (small) set of pairs will result in the desired random distribution. Indeed, in some ways this process will clearly not give a random distribution, since if the least significant values of B and D are the same, and $E' = (E \oplus D) \boxplus B$, then E and E' will have the same least significant bit. The precise effectiveness of the attack thus remains to be determined. However, regardless of the choices of D and B, the function $f(X) = (X \oplus D) \boxplus B$ is a permutation on the set of all n-bit values X. Hence it does seem reasonable to assume that the birthday attack will have a good chance of working.

5.2 Attack Complexity

First observe that the known plaintext block P_s used in the attack could be one of those used to deduce a value of $d_K(C)$, as per Observation 1. Hence the minimum number of known plaintext blocks necessary to perform the attack is just three, as long as they are all consecutive (since three consecutive blocks will yield

two pairs of consecutive plaintext blocks). Second note that the computations required to perform the attack involve purely comparisons of bit strings, ex-ors of bit strings, and modulo 2^n additions of bit strings. All these operations can be computed very quickly. The total number of operations is clearly $O(2^{n/2})$.

The attack could be performed using a number of known ciphertext messages created using the same key, as long as a plaintext block is known for each. In this case, the first half of one ciphertext message could be joined to the second half of a different ciphertext message (with some intermediary blocks inserted).

5.3 Other Integrity Protection Measures

The above attack assumes that a final plaintext block P^* is used for integrity protection. However, precisely the same approach would work if either the last r blocks of plaintext were set to a fixed pattern, or the final block (or r blocks) were set equal to the first block (or r blocks) of the message.

If the message length is fixed, e.g. by including a string indicating the message length as the first or last plaintext block, then the described attack apparently fails. However, a slightly more complex version of the attack will still work if the attacker knows two plaintext blocks in a ciphertext message $C_1, C_2, \ldots, C_t$ at a distance of precisely w blocks apart: C_r and C_{r+w} say. After a search of the same complexity as described above, a string of w replacement ciphertext blocks can be inserted into the message to replace blocks $C_{r+1}, C_{r+2}, \ldots, C_{r+w}$, without altering the decryption of any subsequent blocks. (The details are straightforward — the main difference is that E_0 and E_w will be different).

In fact, even if the final plaintext block is a CRC computed as a function of all the previous plaintext blocks, an attack along similar lines is still probably possible. In this case, only strings of ciphertext blocks $E_0, E_1, \ldots, E_v$ and $F_0, F_1, \ldots, F_v$ that do not affect the CRC computations should be considered. This will increase the attack complexity to some extent, but otherwise it seems that everything should still work.

5.4 Generalisations of the Attack

Finally note that the entire attack strategy outlined in Section 4 could be generalised to other variants of G-PCBC, i.e. to other choices of the function g. Re-examination of the attack reveals that it will still work in exactly the same way as long as g has the following 'invertibility' property. That is, if, given any n-bit blocks X and Y, it is possible to find an n-bit block Z such that $X = g(Y, Z)$, then precisely the same attack strategy will work.

To see this, we outline how to modify the arguments in section 4 to this new scenario. First let g^{-1} be defined such that $g^{-1}(g(X, Y), Y) = X$ for any n-bit blocks X and Y. This 'inverse function' exists by our assumption immediately above. Observation 1 generalises trivially, yielding

$$d_K(C_s) = P_S \oplus g(P_{i-1}, C_{i-1}).$$

Next, as in Theorem 1, suppose $(u_1, u_2, \ldots, u_w)$, $w \geq 1$, $1 \leq u_i \leq r$, satisfies

$$E_i = g(E_{i-1} \oplus D_{u_i}, B_{u_i}), \ 1 \leq i \leq w$$

where $E_0 = E_w = g(P_s, C_s)$. Note that this means that

$$E_{i-1} \oplus D_{u_i} = g^{-1}(E_i, B_{u_i}).$$

Then the G-PCBC decrypted version of the ciphertext message

$$C_1, C_2, \ldots, C_s, B_{u_1}, B_{u_2}, \ldots, B_{u_w}, C_{s+1}, C_{s+2}, \ldots, C_t$$

is equal to

$$P_1, P_2, \ldots, P_s, g^{-1}(E_1, B_{u_1}), g^{-1}(E_2, B_{u_2}), \ldots, g^{-1}(E_w, B_{u_w}), P_{s+1}, P_{s+2}, \ldots, P_t.$$

This follows since:

- the decryption of B_{u_1} yields $d_K(B_{u_1}) \oplus g(P_s, C_s) = D_{u_1} \oplus E_0 = g^{-1}(E_1, B_{u_1})$,
- the decryption of B_{u_i}, $(i > 1)$ yields $d_K(B_{u_i}) \oplus g(B_{u_{i-1}}, g^{-1}(E_{i-1}, B_{u_{i-1}})) = D_{u_i} \oplus E_{i-1} = g^{-1}(E_i, B_{u_i})$, and
- the decrypted version of C_{s+1} equals $d_K(c_{s+1}) \oplus g(B_{u_w}, g^{-1}(E_w, B_{u_w})) = d_K(C_{s+1}) \oplus E_w = d_K(C_{s+1}) \oplus g(P_s, C_s) = P_{s+1}$.

These observations suggest that it is probably dangerous to use any variant of G-PCBC for message integrity, almost regardless of how redundancy is added to the message prior to encryption.

6 M-PCBC

PCBC is appealingly simple to implement, and this motivated recent work by Sierra et al. [13], who define a PCBC variant they call M-PCBC (for *Memory* PCBC). Like G-PCBC and PCBC, M-PCBC requires the message to be divided into a sequence $P_1, P_2, \ldots, P_t$ of n-bit blocks prior to encryption. M-PCBC uses a pair of Initialisation Vectors, which we denote by S_W and S_P, and also a series of intermediate values $W_0, W_1, \ldots, W_t$. Encryption then operates as follows:

$$C_i = e_K(P_i \oplus W_i \oplus P_{i-1}), \ 1 \leq i \leq t,$$

where $W_1 = S_W$, $P_0 = S_P$ and $W_i = G(W_{i-1}, C_{i-1})$, $(1 < i \leq t)$.

The function G maps a pair of n-bit blocks to a single n-bit block, and is defined as follows[1]. Suppose $W = W_L || W_R$ and $C = C_L || C_R$, where $||$ denotes concatenation and W_L, W_R, C_L and C_R are blocks of bits of length $n/2$ (we suppose that n is even, as is always the case in practice). Then

$$G(W, C) = (W_L \oplus W_R) || (C_L \oplus C_R).$$

[1] In order to permit the simplest presentation of the scheme, the notation of [13] has been revised slightly; in the notation of [13], $IV = S_P \oplus S_W$, $IV_2 = S_W$ and $R_i = W_i$.

Hence decryption operates as follows:

$$P_i = d_K(C_i) \oplus W_i \oplus P_{i-1}, \ \ 1 \le i \le t.$$

To use M-PCBC to protect data integrity, Sierra et al. [13] suggest using the same method as proposed for G-PCBC and PCBC, i.e., they propose adding a fixed final plaintext block prior to encryption. Below, we analyse the effectiveness of M-PCBC for integrity protection on the assumption that this approach is used.

7 Some Elementary Properties of M-PCBC

We first give an alternative expression for M-PCBC decryption.

Lemma 1. *If $P_1, P_2, \ldots, P_t$ are obtained from the M-PCBC decryption of ciphertext $C_1, C_2, \ldots, C_t$ using key K and Initialisation Vectors S_P and S_W then, if i satisfies $1 \le i \le t$:*

$$P_i = S_P \oplus \bigoplus_{j=1}^{i} (d_K(C_j) \oplus W_j).$$

where $W_1 = S_W$, $W_i = G(W_{i-1}, C_{i-1})$, $(1 < i \le t)$, and we denote the leftmost $n/2$ bits of C_k and W_k by C_k^L and W_k^L respectively, and the rightmost $n/2$ bits of C_k and W_k by C_k^R and W_k^R.

Proof. We prove the result by induction on i.

For the case $i = 1$, observe that $P_1 = d_K(C_1) \oplus W_1 \oplus P_0$, as required. Now suppose that the result holds for $i = r$ (for some r satisfying $1 \le r < t$). Then:

$$\begin{aligned} P_{r+1} &= d_K(C_{r+1}) \oplus W_{r+1} \oplus P_r \\ &= d_K(C_{r+1}) \oplus W_{r+1} \oplus S_P \oplus \bigoplus_{j=1}^{r} (d_K(C_j) \oplus W_j) \\ &\qquad \text{(by the inductive hypothesis)} \\ &= S_P \oplus \bigoplus_{j=1}^{r+1} (d_K(C_j) \oplus W_j), \end{aligned}$$

as required. The result now follows. □

Lemma 2. *Using the notation and assumptions of Lemma 1,*

$$W_i = (S_W^L \oplus S_W^R \oplus \bigoplus_{k=1}^{i-2} (C_k^L \oplus C_k^R)) || (C_{i-1}^L \oplus C_{i-1}^R), \ \ (1 < i \le t),$$

where S_W^L and S_W^R are the leftmost and rightmost $n/2$ bits of S_W, respectively. That is, for $1 < i \le t$:

$$W_i^L = S_W^L \oplus S_W^R \oplus \bigoplus_{k=1}^{i-2} (C_k^L \oplus C_k^R)$$

and

$$W_i^R = C_{i-1}^L \oplus C_{i-1}^R.$$

Proof. We prove the result by induction on i. If $i = 2$, by definition of G and since $W_1 = S_W$, we have $W_2 = G(W_1, C_2) = (S_W^L \oplus S_W^R)||(C_2^L \oplus C_2^R)$, as required. If the result holds for $i = r$ (for some r satisfying $2 \leq r < t$), then:

$$\begin{aligned} W_{r+1} &= G(W_r, C_r) \ \text{(by definition)}, \\ &= (W_r^L \oplus W_r^R)||(C_r^L \oplus C_r^R) \ \text{(by definition of G)}, \\ &= (S_W^L \oplus S_W^R \oplus (\bigoplus_{k=1}^{r-2}(C_k^L \oplus C_k^R)) \oplus (C_{r-1}^L \oplus C_{r-1}^R))||(C_r^L \oplus C_r^R) \end{aligned}$$

(by the inductive hypothesis), and the result now follows. □

These lemmas then enable us to establish the following result, in which the plaintext recovered from an encrypted message can be expressed as a function only of the ciphertext blocks, the Initialisation Vectors, and the secret key K.

Theorem 2. *If $P_1, P_2, \ldots, P_t$ are obtained from the M-PCBC decryption of $C_1, C_2, \ldots, C_t$ using key K and Initialisation Vectors S_W and S_P then:*

$$P_1 = S_P \oplus S_W \oplus d_K(C_1),$$

$$\begin{aligned} P_i = S_P \oplus S_W \oplus \bigoplus_{j=1}^{i} d_K(C_j) \oplus \\ (S_W^L \oplus S_W^R \oplus \bigoplus_{\substack{k=1 \\ k \ even}}^{i-2} (C_k^L \oplus C_k^R))||(\bigoplus_{j=1}^{i-1}(C_{j-1}^L \oplus C_{j-1}^R)), \end{aligned}$$

$$(i \ \textit{even}, \ 2 < i \leq t),$$

$$P_i = S_P \oplus S_W \oplus \bigoplus_{j=1}^{i} d_K(C_j) \oplus ((\bigoplus_{\substack{k=1 \\ k \ odd}}^{i-2} (C_k^L \oplus C_k^R))||(\bigoplus_{j=1}^{i-1}(C_{j-1}^L \oplus C_{j-1}^R))),$$

$$(i \ \textit{odd}, \ 2 < i \leq t).$$

where C_k^L and C_k^R are as defined previously.

Proof. The equation for P_1 follows immediately from Lemma 1. Now suppose i satisfies $2 \leq i \leq t$. Then the result follows from substituting the equation for W_i from Lemma 2 into the equation from Lemma 1. □

8 Breaking M-PCBC

It follows from Theorem 2 that the recovered plaintext P_i is a function only of the set of ciphertext blocks $\{C_1, C_2, \ldots, C_i\}$ (and the key and IVs) and not of the order in which the ciphertext blocks occur (except with respect to whether

the ciphertext blocks appear in an even or an odd position and the values of C_i and C_{i-1}). This enables trivial ciphertext-only 'forgeries' to be constructed, i.e. manipulations of valid messages for which the decrypted version of the final block will remain unchanged (and hence the integrity check will succeed).

For example, in a five-block encrypted message, interchanging the first and third ciphertext blocks will not affect the decryption of the fifth block, although, of course, the first four blocks will be corrupted. This is directly analogous to the simple attacks on PCBC mode due to Kohl [15].

9 Conclusions

The PCBC+ and M-PCBC modes have been described and various properties of each mode exhibited. These properties imply that both modes are unacceptably weak for one of their main intended uses, namely the protection of data integrity. The M-PCBC mode is particularly weak, in that a simple known ciphertext based forgery attack exists, which is easy to perform regardless of the block cipher block length n. The use of one of the recently designed authenticated encryption modes for which a proof of security exists, such as OCB, CCM or EAX, is recommended instead of either of these modes.

Acknowledgements

The author would like to acknowledge helpful comments and advice provided by Lars Knudsen and Caroline Kudla.

References

1. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1997)
2. Bellare, M., Kohno, T., Namprempre, C.: Breaking and provably repairing the SSH authenticated encryption scheme: A case study of the encode-then-encrypt-and-MAC paradigm. ACM Transactions on Information and System Security **7** (2004) 206–241
3. Black, J., Urtubia, H.: Side-channel attacks on symmetric encryption schemes: The case for authenticated encryption. In: Proceedings of the 11th USENIX Security Symposium, San Francisco, CA, USA, August 5-9, 2002, USENIX (2002) 327–338
4. Canvel, B., Hiltgen, A., Vaudenay, S., Vuagnoux, M.: Password interception in a SSL/TLS channel. In Boneh, D., ed.: Advances in Cryptology — CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings. Volume 2729 of Lecture Notes in Computer Science., Springer-Verlag, Berlin (2003) 583–599
5. Paterson, K.G., Yau, A.: Padding oracle attacks on the ISO CBC mode padding standard. In Okamoto, T., ed.: Topics in Cryptology — CT-RSA 2004, The Cryptographers' Track at the RSA Conference 2004, San Francisco, CA, USA, February 23-27, 2004, Proceedings. Volume 2964 of Lecture Notes in Computer Science., Springer-Verlag, Berlin (2004) 305–323

6. Vaudenay, S.: Security flaws induced by CBC padding — Applications to SSL, IPSEC, WTLS .... In Knudsen, L., ed.: Advances in Cryptology — EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 – May 2, 2002, Proceedings. Volume 2332 of Lecture Notes in Computer Science., Springer-Verlag, Berlin (2002) 534–545
7. Rogaway, P., Bellare, M., Black, J.: OCB: A block-cipher mode of operation for efficient authenticated encryption. ACM Transactions on Information and System Security **6** (2003) 365–403
8. Bellare, M., Rogaway, P., Wagner, D.: The EAX mode of operation. In Roy, B., Meier, W., eds.: Fast Software Encryption, 11th International Workshop, FSE 2004, Delhi, India, February 5-7, 2004, Revised Papers. Volume 3017 of Lecture Notes in Computer Science., Springer-Verlag, Berlin (2004) 389–407
9. National Institute of Standards and Technology (NIST): NIST Special Publication 800-38C, Draft Recommendation for Block Cipher Modes of Operation: The CCM Mode For Authentication and Confidentiality. (2003)
10. Whiting, D., Housley, R., Ferguson, N.: RFC 3610, Counter with CBC-MAC (CCM). Internet Engineering Task Force. (2003)
11. International Organization for Standardization Genève, Switzerland: ISO/IEC WD 19772: 2004, Information technology — Security techniques — Authenticated encryption mechanisms. (2004)
12. Dent, A.W., Mitchell, C.J.: User's Guide to Cryptography and Standards. Artech House (2005)
13. Sierra, J.M., Hernandez, J.C., Jayaram, N., Ribagorda, A.: Low computational cost integrity for block ciphers. Future Generation Computer Systems **20** (2004) 857–863
14. Meyer, C.H., Matyas, S.M.: Cryptography: A new dimension in computer data security. John Wiley and Sons, New York (1982)
15. Kohl, J.T.: The use of encryption in Kerberos for network authentication. In Brassard, G., ed.: Advances in Cryptology — CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20–24, 1989, Proceedings. Volume 435 of Lecture Notes in Computer Science., Springer-Verlag, Berlin (1990) 35–43
16. Steiner, J., Neuman, C., Schiller, J.: Kerberos: an authentication service for open network systems. In: Proceedings: Usenix Association, Winter Conference, Dallas 1988, USENIX Association, Berkeley, California (1988) 191–202
17. Gligor, V.G., Donescu, P.: Integrity-aware PCBC encryption schemes. In: Security Protocols, 7th International Workshop, Cambridge, UK, April 19-21, 1999, Proceedings. Volume 1796 of Lecture Notes in Computer Science., Springer-Verlag, Berlin (2000) 153–171
18. Ferguson, N., Whiting, D., Kelsey, J., Wagner, D.: Critical weaknesses of iaPCBC. (1999)

New Cryptographic Applications of Boolean Function Equivalence Classes

William L. Millan

ISI, QUT
GPO Box 2434
Brisbane, Qld, 4001, Australia
millan@isrc.qut.edu.au

Abstract. In this paper we investigate several issues for equivalence classes of Boolean functions which are interesting for cryptology. As well as reviewing the established concepts, we present three new applications of these ideas. Firstly we propose a novel yet natural extension to the existing transform based equivalence class distinguishing algorithm, which can provide improved performance. Secondly, making novel use of the class graph notion, we completely explain the required conditions for high nonlinearity in the concatenation construction of Boolean functions. Finally, we use the linear class graph to comment on algebraic attacks by defining all the equivalence classes possible for the important set of annihilating functions. This approach provides a new solution to the problem of finding (and avoiding) low degree annihilators.

1 Introduction

Boolean functions are a fundamental model of symmetric cryptosystems. Affine (and also linear and matrix) transformations partition the Boolean space into equivalence classes, which has several applications in the design and analysis of cryptographic functions. It is now well-known that the popular finite field exponentiation operation over the extension field $GF(2)^n$ is linearly (matrix only) redundant [12] (that is all the output functions are equivalent under a matrix transform: they share the same equivalence class), and that this property is unexpected in random mappings. Many of the new standard block ciphers (including Rijndael [10] and Camellia [2]) have finite field operations as the only nonlinear component, and there are claims of algebraic attacks (of the kind recently proposed on many ciphers, see for example [9,3]) that are enabled by the resulting overdefined systems of equations. In another direction, equivalence class analysis has recently yielded advances in the heuristic search for cryptographically strong Boolean functions. For example improved ability to generate optimal Boolean functions including bent functions [13] and maximally nonlinear balanced functions[16] have now been reported. In addition, all the functions of 6 inputs have been found and analysed using these methods [16], despite that there are 2^{64} functions to search through.

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 572–583, 2005.

In this paper we discuss several issues of interest for equivalence classes of Boolean functions, including the distinguishing problem (given two Boolean functions, decide if they are equivalent or not under some affine (or lesser) transformation), local connectivity properties of the graph of equivalence classes, and the important differences between the affine, linear and matrix equivalence classes and their graphs. As well as reviewing the established concepts of Boolean Functions in Section 2 and investigating the graphs of equivalence classes in Section 3, we present three entirely new applications of these ideas.

Firstly, in Section 4 we propose modifications to the existing equivalence class distinguishing algorithms, which offer the opportunity for improved performance. These modifications make use of the Transfrom Value Indicator Functions (TVIF) which are functions defining the location of particular values in the Walsh-Hadamard (WH) and Autocorrelation (AC) spectra. Then in Section 5, combining the TVIF notion with the idea of sets of *covering* classes in the linear class graph, we completely explain the required conditions for the concatenation construction to produce highly nonlinear Boolean functions, which has long remained a central open problem in symmetric cryptography. This contribution allows the search for many kinds of optimal functions to be performed both systematically and much more effectively than previously.

Finally in Section 6 we comment on algebraic attacks by completely defining the equivalence classes possible for the important set of *annihilators* [15] (given $f(x)$, the annihilators are those functions $g(x)$ such that $f(x)*g(x) = 0$) using the covering properties of the linear class graph. This approach shows how analysis of the linear equivalence class graphs provides a solution to the problem of searching for low-degree annihilators of a given function, and it also allows identification of those classes *without* low-degree annihilators. These results have application in stream cipher design. We make some concluding remarks in Section 7.

2 Boolean Functions and Their Cryptographic Criteria

In this section we present the basic theory and notation relating to Boolean functions, linear approximation, the Walsh Hadamard Transfrom, the algebraic normal form, and autocorrelation. Some important theorems are recalled from the cryptographic literature.

Boolean Basics A Boolean Function $f : Z_2^n \to Z_2$ is a mapping from n binary inputs to a single binary output. The list of all the 2^n possible outputs is the *truth table*. Often we consider the *polarity truth table* $\hat{f}$ defined by

$$\hat{f}(x) = (-1)^{f(x)} = 1 - 2 * f(x).$$

If a function can be expressed as an XOR sum of input variables, then it is said to be *linear*. Let the n-bit binary vector ω select the variables from input $x = (x_1, x_2, \cdots, x_n)$, then the linear function defined by ω is denoted by $L_\omega(x) = \omega_1 x_1 \oplus \omega_2 x_2 \oplus \cdots \oplus \omega_n x_n$. The set of *affine* functions is the set of all linear functions and their complements: $A_{\omega,c}(x) = L_\omega(x) \oplus c$, where $c \in \{0, 1\}$. A

function is *balanced* when all its output symbols are equally likely. It is clear that $\sum_x \hat{f}(x) = 0$ occurs if and only if the function f is balanced. It should be noted that XOR in the binary domain is equivalent to multiplication over the set $\{1, -1\}$: $h = f \oplus g$ implies that $\hat{h} = \hat{f} \cdot \hat{g}$. Two functions are *uncorrelated* when their XOR sum is balanced.

Walsh Transform The correlation between a function f and the linear function L_ω is proportional to the value $\hat{F}(\omega)$ in the *Walsh-Hadamard Transform* (WHT) defined by $\hat{F}(\omega) = \sum_x \hat{f}(x)\hat{L}_\omega(x)$. WHT values are always divisible by 2. A zero in the WHT at position ω ($\hat{F}(\omega) = 0$) indicates that f is uncorrelated with $L_\omega(x)$. In particular every balanced function (which is uncorrelated with the all-zero function) has $\hat{F}(0) = 0$. In general the correlation between f and L_ω is given by $c(f, L_\omega) = \frac{\hat{F}(\omega)}{2^n}$. The *nonlinearity* of a Boolean function is given by $NL_f = \frac{1}{2}(2^n - WHmax(f))$, where $WHmax(f) = max\{|\hat{F}(\omega)|\}$ over all values of ω, and it shows the minimum number of truth table positions that must be altered to change f into an affine function. Cryptography seeks higher values of nonlinearity (lower values of $WHmax$) as this reduces the value of the best affine approximation. Ciphers using highly nonlinear functions are more difficult to attack.

Another important property in stream cipher design is *resilience* [20,21], which can be seen as a kind of higher order balance. A t-resilient Boolean function is both balanced and has $\hat{F}(\omega) = 0$ for all ω with weight t or less [24]. This is equivalent to saying that any subfunction of f, induced by setting t or fewer inputs constant to any value, is exactly balanced. The structure of resilient functions is always recursive: given any t-resilient function, any subfunction of it selected by fixing m bits is a $(t - m)$-resilient function.

We note that calculation of $\hat{F}$ in the WHT domain directly is natural for the concatenation construction. Let $f = f_0||f_1$, then $\hat{F} = (\hat{F}_0 + \hat{F}_1)||(\hat{F}_0 - \hat{F}_1)$, where $\hat{F}_i = WHT(f_i)$.

Algebraic Normal Form It is possible to represent any Boolean function as an Algebraic Normal Form (ANF) which is the XOR sum of a subset of all the 2^n possible ANDed product terms of the n input variables. The *algebraic degree*, d, is maximum number of variables in any term of the ANF. Linear functions are limited to ANFs with only single variables, so they have $d = 1$. For security reasons, cryptology seeks to use functions with high algebraic degree (and in fact the vast majority of functions have $d \geq n - 1$), however it is known that high degree conflicts with other desirable properties like resilience and nonlinearity and autocorrelation.

Autocorrelation The autocorrelation function (AC) is a vector $r_f(s)$ of 2^n integers similar to the WHT. The autocorrelation values are proportional to the correlation that $f(x)$ has with the "shifted version" $f(x \oplus s)$. The autocorrelation function is defined by $\hat{r}_f(s) = \sum_x \hat{f}(x) \cdot \hat{f}(x \oplus s)$. The values in the AC should be small for security [1], and they are always divisible by 4. We let $ACmax =$

$max\{|\hat{r}_f(s)|\}$ where the maximum is taken over the range $1 \leq s \leq 2^n - 1$ and note that $\hat{r}_f(0) = 2^n$ for all Boolean functions since any function is identical to itself.

We now review some important results in Boolean functions, both well-known and recent.

- **Parseval's Theorem [14].** $\sum_\omega (\hat{F}(\omega))^2 = 2^{2n}$. The sum of the squares of the WHT values is always the same constant for all n-input Boolean functions. This means that every function has some correlation to affine functions, and the best that can be done (to generate high nonlinearity) is to minimise the maximum value in the WHT.
- **Bent Functions [19]** For even n, the set of maximaly nonlinear functions are called *bent*. They have all WHT values with magnitude $2^{\frac{n}{2}}$, thus maximising nonlinearity at $NL_{bent} = 2^{n-1} - 2^{\frac{n}{2}-1}$. The algebriac degree of bent functions is limited in range: $2 \leq d_{bent} \leq \frac{n}{2}$. It is known that $ACmax = 0$ only for bent functions ($\hat{r}_{bent}(s) = 0$ for $s > 0$), so they also optimise this property. Note that bent functions are never balanced or resilient.
- **Siegenthaler Tradeoff [20]** There is a direct conflict between algebric degree, d, and the order of resiliency, t, given by $d + t \leq n - 1$. This result also holds for balanced functions (which can be considered as having $t = 0$) and indeed any function (for which we may let $t = -1$).
- **Fast Autocorrelation Calculation [18]** The autocorrelation vector can be calculated as the inverse WHT of the vector formed by squaring all the values in $\hat{F}$. A direct approach uses $2^n \cdot 2^n = 2^{2n}$ operations compared with $n \cdot 2^n$ operation in a WHT or its inverse! Autocorrelation for moderate n is not feasible unless this method is used.
- **Balance and Nonlinearity [11]** There is a construction for balanced, highly nonlinear functions (BHNL) that is the currently best known and it is conjectured to attain the maximum possible nonlinearity for balanced functions. Given that $NLB(n)$ is the maximum possible nonlinearity for balanced functions with n inputs, one may construct a balanced function on $2n$ inputs with nonlinearity $NLB(2n) = 2^{2n-1} - 2^{\frac{n}{2}} + NLB(n)$.
- **Transform Value Divisibility [22,25,7]** The simplest expression of the several recent results relating the divisibility properties of values in the WHT to other criteria is as follows. Let f be a t-resilient boolean function, then 2^{t+2} divides evenly into $\hat{F}(\omega)$, for all ω. It follows that the nonlinearity of a t-resilient function must be divisible by 2^{t+1}. As above, these results also hold for the extended definition of resiliency to include $t = 0$ (balanced) and $t = -1$ (all) functions.

3 Equivalence Classes

In this section we investigate the properties of equivalence classes of Boolean functions. In particular we consider the effects that affine, linear and matrix transforms have on their Walsh-Hadamard and Autocorrelation spectra, by introducing a new tool: the Transform Value Indicator Function (TVIF). We also

begin to examine the properties of the *graph of classes* which are relevant to the observations that follow.

3.1 Transform Value Indicator Functions

Many important cryptographic properties of boolean functions are conserved by affine transformation. It is well known that the nonlinearity and the autocorrelation spectral maxima are unchanged by any affine transformation, since the transform values are merely re-arranged. In this section we investigate the spectral effects of transformation in more detail, focusing on the structural properties (of sets of values) which do not change under the transformation. We show that using particular boolean functions derived from the transforms allows opportunities to improve the performance of class distinguishing algorithms.

Definition 1. *The Transform Value Indicator Function (TVIF) for the value v in the Walsh-Hadamard spectrum of the boolean function f is defined by*

$$t_{f,v}(y) = 1 \text{ whenever } \hat{F}(y) = v, \text{ else}$$
$$t_{f,v}(y) = 0$$

Similarly we define the TVIF for the values in the autocorrelation function using the notation $q_{f,v}(y)$. We may also define the unsigned (or absolute value) based TVIF for the set of values $v, -v$ as the logical OR of $t_{f,v}$ and $t_{f,-v}$, which for $v > 0$ have disjoint support. In general we can consider the TVIF with respect to any set of values. When we consider finding applications for the effect of affine transformations on the TVIF derived from the Walsh-Hadamard transform and the autocorrelation function, we sometimes want the signed TVIF and sometimes the unsigned TVIF. For applications in constructing highly nonlinear functions, we need the TVIF with respect to all absolute values less than a threshold.

Now we need to define the types of equivalence classes induced by three nested kinds of transformation: affine, linear and matrix. Let $Aclass(f) = Aclass(g)$ if and only if f is equivalent to g under an affine transformation. Similarly we use the notation $Lclass$ and $Mclass$ when referring to equivalence under linear transforms and matrix transforms respectively. Clearly M-equivalence implies L-equivalence implies A-equivalence.

Definition 2. *An* affine *transformation is defined by a nonsingular binary $n*n$ matrix M, two n-bit binary vectors b and β, and a single constant bit, c. We say two Boolean functions f and g are in the same* Aclass *if and only if they are equivalent under some affine transformation $g(x) = f(Mx \oplus b) \oplus L_\beta(x) \oplus c$.*

Definition 3. *Two functions f, g are in the same* Lclass *when there exists M, b such that $g(x) = f(Mx \oplus b)$. Two functions f, g are in the same* Mclass *when there exists a nonsingular binary matrix M such that $g(x) = f(Mx)$.*

Theorem 1. *Let* $g(x) = f(Ax \oplus b) \oplus L_\beta(x) \oplus c$ *be an affine transform on the truth table of* f*, and let* $\boldsymbol{v}$ *denote the set of values* $v, -v$*, then* $Lclass(t_{f,\boldsymbol{v}}) = Lclass(t_{g,\boldsymbol{v}})$*. More precisely:*

$$t_{g,\boldsymbol{v}}(y) = t_{f,\boldsymbol{v}}(A^* y \oplus b^*)$$

where $A^* = tr(A^{-1})$*,* $b^* = tr(A^{-1} \cdot \beta)$ *and* $tr()$ *denotes the transpose of a matirx. Considering the effect on the position of values in the autocorrelation function, we find* $Mclass(q_{f,\boldsymbol{v}}) = Mclass(q_{g,\boldsymbol{v}})$*, since:*

$$q_{g,\boldsymbol{v}}(y) = q_{f,\boldsymbol{v}}(A^* y)$$

This theorem shows that when an affine transformation is applied to the truth table, then each corresponding pair of *unsigned* (absolute value) Walsh-Hadamard based TVIFs must share the same linear class, and the autocorrelation based TVIFs share the same matrix class.

Corollary 1. *Let* $Aclass(f) = Aclass(g)$*. We can say that* $Lclass(t_{f,V}) = Lclass(t_{g,V})$ *(resp.* $Mclass(q_{g,V}) = Mclass(q_{f,V})$*) is true for all sets of WHT values (resp. AC values)* $V \in Z$ *that contain* $-v$ *if and only if they also contain* v*. Note the functions defined by any and all combinations of magnitude (absolute values) share the same Lclass (resp. Mclass).*

Any distribution difference shows inequivalence, so following and comparing the value distributions of TVIFs can be used to improve the complexity of algorithms for distinguishing equivalence classes. Linear and matrix classes can be easier to distinguish than affine classes since the Hamming weight of the function acts as a distinguisher (whereas it is not for affine classes).

Corollary 2. *Let* $Lclass(f) = Lclass(g)$*, then the unsigned WHT-based TVIF share the same Mclass and the signed AC-based TVIF share the same Mclass.*

Corollary 3. *Let* $Mclass(f) = Mclass(g)$*, then* $Mclass(t_{f,v}) = Mclass(t_{g,v})$ *and* $Mclass(q_{f,v}) = Mclass(q_{f,v})$ *for all values* v*.*

3.2 Class Graphs

We now recall from [16] the novel idea of a graph of affine equivalence classes. Each class is a node and the connections between them, due to single truth table changes, are arcs. For *Aclasses*, we arrange the nodes so that the vertical axis shows nonlinearity. This allows heuristic search where any Boolean function is represented by its class node and the shape of the graph shows local nonlinearity maxima clearly. For the *Lclass* and *Mclass* graphs we arrange nodes so that the vertical axis shows Hamming weight, the balanced functions occupy the central row and the whole graph is symmetrical about this axis.

Structural invarients of the local connections of the *Aclass* graph have been previously used [12] to discover the transformations that connect the equivalent

functions of the AES S-boxes. Similar (but simpler) invarients must apply also to the structure of L-classes and M-classes. However, in this paper we consider the relation of *covering* in a class graph. For the applications which follow, we consider covering in *linear* class graphs.

The task of determining whether or not two n-input Boolean function classes are related by covering can be performed by establishing the full L-class graph. This might become an expensive computation, however there are some theoretical observations that allow efficiency improvements. Firstly we define minimum and maximum guaranteed covering weights.

Definition 4. *For every L-class C with hamming weight w_C, there exists a minimum weight $W_{min} \geq w_C$, such that every L-class with hamming weight not less than W_{min} must cover the class C. We say that W_{min} is the* minimum guaranteed covering weight *for class C.*

Definition 5. *For every L-class C with hamming weight w_C, there exists a maximum weight $W_{max} \leq w_C$, such that every L-class with hamming weight not greater than W_{max} must be covered by the class C. We say that W_{max} is the* maximum guaranteed weight covered *by class C.*

In the next few sections we present some applications of these new ideas. Firstly we improve the class distinguishing algorithms by following the TVIF paths. Secondly we introduce novel requirements for optimal construction of highly nonlinear Boolean functions using the fundamental operation of concatenation, by pointing out previously unknown connections between the notion of *spectral compatability* and the covering structure of L-classes. Thirdly, we provide a complete definition of equivalence classes that avoid low-degree *annihilators*, based on the structure of the graph of L-classes. This "no low-degree annihilators" property has recently been added to the classic design criteria for Boolean functions in stream ciphers [15]. Our results define exactly those equivalence classes that have (and do not have) low degree annihilators, which improves on previous work that offers to find all annihilators individually, without the performance improvement offered by the equivalence class partition.

4 Class Distinguishing Using TVIFs

In this section we propose enhancements of the existing equivalence class distinguishing algorithms. These techniques use the invarient properties of transform value indicator functions (TVIF).

Definition 6. *Let two Boolean functions f, g share the same spectral distribution. Then we say that their Aclasses C_f, C_g are a* spectral distribution collision.

Theorem 2. *Let f, g be two Boolean functions from different equivalence classes that nevertheless share the same WHT & AC TVIF distribution trees, then each and every pair of corresponding classes in that tree must be a spectral distribution collision.*

Remark: WHT and AC TVIF distribution trees can be used to distinguish the vast majority of equivalence classes. Exceptions will be very rare, if they exist at all. We can say it is an open problem to find two distinct classes that share identical (value,frequency) distribution data in their TVIF trees. If no such pair exists, then the TVIF tree distinguisher method always works.

It should be clear (from the *duality* property [19][1]) that bent functions cannot be class-distinguished using the basic TVIF trees. However the technique can be extended to apply to the set of functions at some nonzero hamming distance, as used in [13]. That paper used *all* the different functions at the selected radius, and *all* the transforms were required. We now devise a second improvement to our distinguishing algorithm by exploiting the birthay paradox which ensures collisions in a set of N objects can be found with effort around $\sqrt{N}$. This greatly reduces the number of functions required to test.

Consider that if f, g are indeed from the same equivalence class, that the set of TVIF trees at radius r is identical for both functions. Considering a sample of functions at the selected radius from both functions, we expect to find a TVIF-tree collision in the worst case (where all 2^n changes finds a different equivalence class[2]) after about $2^{\frac{n}{2}}$ nearby functions have been analysed. Compare this with the case when f, g are from different classes and hence the samples are coming from two sets of classes that are *not* identical (and *may not* have any TVIF distribution tree collisions). We may conclude with high probability (from a sufficient number of samples without finding a TVIF-tree collision) that f, g are from distinct equivalence classes.

5 Design of Highly Nonlinear Boolean Functions

In this section we examine the required conditions for concatenation construction of highly nonlinear Boolean functions, which has remained an important open problem. Our solution uses the notion of spectral compatability.

Definition 7. *Two Boolean functions f and g have WHT spectra that are v-compatable if and only if $|\hat{F}(\omega)| + |\hat{G}(\omega)| \leq v$ for all ω. This means that*

$$WHT_{max}(f||g) \leq v.$$

Two eqivalence classes are v-compatable if and only if there exist two functions (one from each class) that are v-compatable.

Definition 8. *Let two boolean functions be v-compatable. The pair achieves* optimum compatability *if and only if $v = WHmax_f + WHmin_g$ or $v = WHmax_g + WHmin_f$.*

[1] The signed TVIF of any bent function is always itself bent.
[2] The n=6 survey shows that this does indeed occur.

Pairs that are optimally compatable can be concatenated with the minimum possible increase in the spectral radius (which is WH_{max}). For designing compatable spectra for concatenation, we consider linear equivalence classes, which produce M-type transforms on the unsigned TVIF of the WHT.

Consider concatenation $f||g$ in the WHT domain. Let the high magnitude WHT TVIF be denoted f_H and g_H and the low magnitude WHT TVIF be f_L and g_H respectively. When C_{f_L} covers C_{g_H} and C_{g_L} covers C_{f_H}, then the spectra are said to be compatable with respect to that partition into high and low magnitude sets. Let the low magnitude WHT values be in the range $[0..v]$, so the high magnitude values are in the range $[v+2, ...v_{max}]$. The WH_{max} of $f||g$ is then upper bounded by $v_{max} + v$.

We have a useful basic result on spectral compatability that reduces the required effort.

Theorem 3. *Let f, g be two Boolean functions with the positions of high and low WHT values defined by the TVIF f_H, f_L, g_H and f_L, which are identified with the M-Classes C_{f_H}, C_{f_L}, C_{g_H} and C_{f_L}. Without loss of generality assume that C_{f_L} covers C_{g_H} on the M-graph. Then C_{g_L} also covers C_{f_H}.*

Perfect spectral compatability occurs for two classes that have the property that the high magnitude WHT TVIF class of one *is the same* matrix class as the low magnituide WHT TVIF of the second function, and vice-versa.

We may use the interesting class of highly nonlinear Boolean functions known as PW functions [17] to demonstrate the effectiveness of optimal spectral compatability. The $n = 15$ function can be partitioned into the natural subfunctions, $(f = f_0||f_1)$ each of which has a corrsponding WHT vector (F_0 and F_1). It can be shown by computer assisted inspection that the high and low WHT value TVIFs of these PW-subfunctions are an example of optimal spectral compatability that results in improved Nonlinearity using the concatenation construction.

Example Constructing a PW-function f with 15 inputs, and spliting it into the two natural subfunctions f_0 and f_1, we calculate their WHT transforms and find the (value,frequency) distributions.

When concatenated, the values in these functions' spectra match according to Table 1. These subfunctions are optimally compatable, since the maximum magnitiude of their concatenated WHT is 216, precisely the same as the maximum magnitide in the WHT of f_0. These subfunctions do not display perfect compatability.

6 Searching for Annihilators Using the Linear Equivalence Class Graph

In this section we comment on algebraic attacks by completely defining the equivalence classes possible for the important set of annihilators [15] using the linear class graph. This approach shows how analysis of the linear equivalence class graphs provides a solution to the problem of searching for low degree annihilators, and for finding functions without such vulnerabilities. We first notice

Table 1. Walsh Spectrum Compatability, PW subfunctions

Value	0	64	128	192
24		13		6467
40	157		108	
88	3		1290	
104		1638		
152		90		
168	4031			
216	2587			

that f, g are annihilators if and only if $f(x) = 1$ only in a subset of places where $g(x) = 0$, and $g(x) = 1$ only in a subset of places where $f(x) = 0$. This relates directly to the covering property we defined earlier.

Theorem 4. *Let two Booelan functions f, g have complement functions $\bar{f} = f \oplus 1, \bar{g} = g \oplus 1$, and let g be an annihilator of f, then the linear equivalence class $Lclass(\bar{f})$ covers the linear equivalence class $Lclass(g)$, where $g \in Lclass(g)$ and $\bar{f} \in Lclass(\bar{f})$, and $Lclass(\bar{g})$ covers the linear equivalence class $Lclass(f)$, where $f \in Lclass(f)$ and $\bar{g} \in Lclass(\bar{g})$.*

There can be no doubt about covering relations for classes with extreme hamming weights. However, for classes near balance there may be a (perhaps partially exhaustive) graph search required to determine (or deny) the covering relation. We can improve on that scenario by considering an algorithm for seeking a covering relation between two functions. We need to define a *smooth* path in a class graph as a sequence of classes with strictly increasing (or strictly decreasing) hamming weights. In other words no two classes in the (always locally connected) path have the same hamming weight, w.

Theorem 5. *Given two boolean function L-classes, C_f, C_g, (where w.l.o.g $w_f > w_g$) there exists a covering relation between them if and only if there exists a smooth path (in the graph of Lclasses) between C_f and C_g of length exactly $w_f - w_g$.*

An overview of the improved algorithm we propose to detect covering relations between L-classes is as follows:
1) Choose a parameter a approx. $\frac{w_f - w_g}{2}$.
2) Obtain a sample of L-classes at distance a from C_f, and call it set A.
3) Obtain a sample of L-classes at distance $w_f - w_g - a$ from C_g, call it set B.
4) if a class-collision occurs between sets A and B, then lass C_f covers C_g.
5) repeat for larger samples, choose different value for a.
6) if no collision is ever found in step 4, then the classes are not covering related.

Using a version of this algorithm that finds which classes cover any low degree class, it is possible to find all the classes that do not have low degree annihilators. Fortunately, it is known that this set intersects with the set of classes that contain functions with high nonlinearity.

7 Conclusion

In this paper we have presented three novel applications of the graph of equivalence classes of Boolean functions: faster class discrimination, improved construction of highly nonlinear functions and alternative methods to calculate a solution to the annihilator-freedom problem.

In addition, by combining the second and third contributions with simple heuristic search, we arrive at a new and potentially useful algorithm that searches for all highly nonlinear Boolean functions (optionally resilient) that also do not have any low degree annihilators. Such functions are important for design of stream ciphers.

References

1. C.M. Adams "On Immunity Against Biham and Shamir's Diferential Cryptanalysis", Information Processing Letters, vol. 41, pages 77-80, 14th Feb 1992.
2. K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima and T. Tokita, "Camellia: A 128-Bit Block Cipher Suitable for Multiple Platforms - Design and Analysis", Proceedings of SAC'00, LNCS vol. 2012, pages 39-56, Springer-Verlag, 2001.
3. F. Armknecht and M. Krause. Algebraic Attacks on Combiners with Memory, CRYPTO'03, pages 162-175, LNCS vol. 2729, Springer, 2003.
4. E.R. Berlekamp and L.R. Welch, "Weight distributions of the cosets of the (32,6) Reed-Muller code", IEEE Transactions on Inform. Theory, IT-18(1):203-207, 1972.
5. A. Canteaut and C. Carlet and P. Charpin and C. Fontaine, "On Cryptographic Properties of Cosets of $R(1, m)$", IEEE Trans. Inform. Theory, vol. 47, No. 4, pages 1494-1513, May 2001.
6. A. Canteaut and P. Charpin, "Decomposing Bent Functions", IEEE Trans. Inform. Theory, vol. 49, No. 8, pages 2004-2019, August, 2003.
7. C. Carlet, "On the coset weight divisibility and nonlinearity of resilient and correlation immune functions", In Proceedings of Sequences and Their Applications - SETA 2001, Discrete Mathematics and Theoretical Computer Science, pages 131-144, Springer-Verlag, 2001.
8. C. Carlet and P. Sarkar. Spectral Domain Analysis of Correlation Immune and Resilient Boolean Functions, Finite Fields and Applications, Vol. 8, No. 1, pages 120-130, 2002.
9. C. Courtois and J. Pieprzyk, Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. Proceedings of Asiacrypt'02, LNCS vol. 2501, Springer-Verlag, 2002.
10. J. Daemen and V. Rijmen, "The Design of Rijndael", Springer, 2002.
11. H. Dobbertin, "Construction of Bent Functions and Balanced Boolean functions with High Nonlinearity", In Fast Software Encryption, LNCS vol. 1008, pages 61-74, Springer-verlag, 1994.
12. J. Fuller and W. Millan, Linear Redundancy in S-boxes, FSE'03 Workshop proceedings, pages 79-92, LNCS vol. 2887, Springer-Verlag, 2003. A preliminary version appears as "On Linear Redundancy in the AES S-box", available at IACR e-print archive 111/2002.
13. Fuller, Millan and Dawson, "Evolutionary Generation of Bent Functions for Cryptology", proceedings of CEC 2003, pages 1655-1661, IEEE, 2003.

14. W. Meier and O. Staffelbach, "Nolinearity Criteria for Cryptographic Functions", Advances in Cryptology - EUROCRYPT'89, LNCS vol. 434, pages 549-562, Springer-Verlag, 1990.
15. W. Meier and E. Pasalic and C. Carlet. Algebraic Attacks and Decomposition of Boolean Functions. Proceedings of EUROCRYPT'04, LNCS vol. 3027, pages 474-491, Springer, 2004.
16. W. Millan, J. Fuller and E. Dawson, "New Concepts in Evolutionary Search for Boolean Functions in Cryptography", proceedings of CEC 2003, pages 2157-2164, IEEE, 2003.
17. N.J. Patterson and D.H. Wiedemann, "Correction to - the covering radius of the (2^{15},16) Reed-Muller code is at least 16276", IEEE Trans. Inform. Theory, vol 36, p.443, March 1990.
18. B.Preneel et.al. "Propagation Characteristics of Boolean Functions", In Advances in Cryptoogy - EUROCRYPT'90, LNCS vol. 473, pages 161-173, Springer-Verlag, 1990.
19. O.S. Rothaus, "On Bent Functions", Journal of Combinatorial Theory, Series A, 20:300-305, 1976.
20. T. Siegenthaler, "Correlation immunity of nonlinear combining functions for cryptographic applications", IEEE Trans on IT, IR-30, No. 5, pages 776-780, Sep 1984.
21. T. Siegenthaler, "Decrypting a Class of Stream Ciphers using Ciphertext only.", IEEE Trans on Computers, C-34(1):81-85, Jan 1985.
22. Y. Tarannikov, "On Resilient Boolean Functions with Maximum Possible Nonlinearity", In Progress in cryptology - *INDOCRYPT'2000*, LNCS vol. 1977, pages 19-30, Springer-Verlag, 2000. Originally available as IACR eprint 2000/005 from http://www.iacr.org
23. Webster and Tavares. "On the design of S-boxes". Proceedings of CRYPTO'85, pages 523-534, Springer-Verlag, 1985.
24. G.-Z. Xiao and J.L. Massey, "A Spectral Characterisation of Correlation Immune Combining Functions", IEEE Trans. IT, 34(3):569-571, 1988.
25. Y. Zheng and X.M. Zhang, "Improved Upper Bound on the Nonlinearity of High Order Correlation Immune Functions", In Proceedings of *SAC'2000*, Workshop of Selected Areas in Cryptology, LNCS vol. 2012, pages 264-274, Springer-Verlag, 2001.
26. Y. Zheng and X.M. Zhang, "On Relationship Among Avalanche, Nonlinearity and Propagation Criteria", *Asiacrypt 2000*, LNCS vol 1976, pages 470-483, Springer-Verlag, 2000.

Author Index