Willy Susilo
Yi Mu
Jennifer Seberry (Eds.)

LNCS 7372

Information Security and Privacy

17th Australasian Conference, ACISP 2012
Wollongong, NSW, Australia, July 2012
Proceedings

Springer

Willy Susilo Yi Mu Jennifer Seberry (Eds.)

Information Security and Privacy

17th Australasian Conference, ACISP 2012
Wollongong, NSW, Australia, July 9-11, 2012
Proceedings

Springer

Volume Editors

Willy Susilo
Yi Mu
Jennifer Seberry
University of Wollongong
School of Computer Science and Software Engineering
Northfields Avenue
Wollongong, NSW 2522, Australia
E-mail: {wsusilo; ymu; jennie}@uow.edu.au

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-31447-6 e-ISBN 978-3-642-31448-3
DOI 10.1007/978-3-642-31448-3
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012940392

CR Subject Classification (1998): K.6.5, E.3, D.4.6, E.4, J.1, K.4.4

LNCS Sublibrary: SL 4 – Security and Cryptology

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

Printed on acid-free paper

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

Preface

The 17th Australasian Conference on Information Security and Privacy (ACISP 2012) was held at Wollongong, Australia, during July 9–11, 2012. The conference was sponsored by the Centre for Computer and Information Security of the University of Wollongong. The submission and review process was run using the iChair software, written by Thomas Baigneres and Matthieu Finiasz from EPFL, LASEC, Switzerland. We would like to thank them for letting us use their iChair software.

The conference received 89 submissions, out of which the Program Committee selected 30 full papers and 5 short papers for presentation at the conference after a rigorous review process. These papers are included in the proceedings. The accepted papers cover a range of topics in information security, including some fundamental theory, cryptanalysis, message authentications, hash functions, public key cryptography, digital signatures, identity-based cryptography, attribute-based cryptography, lattice-based cryptography, lightweight cryptography and RFIDs. The conference proceedings contain revised versions of the selected papers. Since some of them were not checked again for correctness before publication, the authors bear full responsibility for the contents of their papers. We would like to thank the authors of all papers for submitting their work to the conference.

In addition to the contributed papers, the program included two invited talks. The invited speakers were Mihir Bellare (University of California, San Diego), with the topic "A Cryptographic Treatment of the Wiretap Channel" and Jorge Munilla (Universidad de Málaga, Spain), with the topic "Operating Principles of RFID Sysems and Attacks Related to the Location." We would like to express our thanks to them.

As in previous years, we selected a "best student paper." To be eligible for selection, a paper has to be co-authored by a postgraduate student, whose contribution was more than 50%. The winner was Xuhua Zhou from Shanghai Jiao Tong University, P.R. China, for the paper "A Generic Construction of Accountable Decryption and Its Applications."

We would like to thank all the people who helped with the conference program and organization. In particular, we heartily thank the Program Committee and the sub-reviewers listed on the following pages for the effort and time they contributed to the review process. A special thanks to the Publication Chair, Man Ho Au, who spent a tremendous amount of time for the success of the conference. We would also like to express our thanks to Springer for continuing to support the ACISP conference and for help in the conference proceedings production.

Finally, we would like to thank the General Chair, Jennifer Seberry, and the Organizing Committee for their excellent contribution to the conference.

July 2012

Willy Susilo
Yi Mu

Organization

General Chair

Jennifer Seberry	University of Wollongong, Australia

Program Chairs

Willy Susilo	University of Wollongong, Australia
Yi Mu	University of Wollongong, Australia

Program Committee

Michel Abdalla	École Normale Supérieure, France
Joonsang Baek	Khalifa University, United Arab Emirates
Alex Biryukov	Université du Luxembourg, Luxembourg
Paulo Barreto	University of São Paulo, Brazil
Feng Bao	Institute for Infocomm Research, Singapore
Lynn Batten	Deakin University, Australia
Colin Boyd	Queensland University of Technology, Australia
Serdar Boztas	RMIT, Australia
Xiaofeng Chen	Xidian University, China
Nicolas T. Courtois	University College London, UK
Yvo Desmedt	University College London, UK
Orr Dunkelman	University of Haifa, Israel
Steven Galbraith	University of Auckland, New Zealand
Qiong Huang	South China Agricultural University, PR China
Xinyi Huang	Fujian Normal University, PR China
Apu Kapadia	Indiana University Bloomington, USA
Xuejia Lai	Shanghai Jiaotong University, China
Dong Hoon Lee	Electronics and Telecommunications Research Institute, Korea
Keith Martin	Royal Holloway, University of London, UK
Chris Mitchell	Royal Holloway, University of London, UK
Fabien Laguilaumie	UCBN and CNRS, LIP (Lyon), France
Atsuko Miyaji	Japan Advanced Institute of Science and Technology, Japan
Juan Gonzales Nieto	Queensland University of Technology, Australia
Miyako Ohkubo	National Institute of Information and Communications Technology, Japan
Claudio Orlandi	Bar-Ilan University, Israel
Udaya Parampalli	University of Melbourne, Australia

Josef Pieprzyk	Macquarie University, Australia
Mark Ryan	University of Birmingham, UK
Rei Safavi-Naini	University of Calgary, Canada
Palash Sarkar	Indian Statistical Institute, India
Ron Steinfeld	Macquarie University, Australia
Douglas Stinson	University of Waterloo, Canada
Tsuyoshi Takagi	Kyushu University, Japan
Vijay Varadharajan	Macquarie University, Australia
Kan Yasuda	NTT Corporation, Japan
Guilin Wang	University of Wollongong, Australia
Huaxiong Wang	Nanyang Technological University, Singapore
Duncan S. Wong	City University of Hong Kong, Hong Kong
Shouhuai Xu	University of Texas at San Antonio, USA
Guomin Yang	National University of Singapore, Singapore

Publication Chair

Man Ho Au	University of Wollongong, Australia

Organizing Committee

Fuchun Guo	University of Wollongong, Australia
Thomas Plantard	University of Wollongong, Australia
Reza Reyhanitabar	University of Wollongong, Australia

External Reviewers

Manal Adham
Yoshinori Aono
Myrto Arapinis
Gilad Asharov
Sergiu Bursuc
Guilhem Castagnos
Melissa Chase
Yuan Chen
Kai-Yuen Cheong
Koji Chida
Bernard Colbert
Anupam Datta
Angelo De Caro
Wei Gao
Jian Guo
Guillaume Hanrot
Takuya Hayashi
Matt Henricksen
Javier Herranz
Mathias Herrmann
Qiong Huang
Hyoseung Kim
Kitak Kim
Milyoung Kim
Aleksandar Kircanski
Woo Kwon Koo
Ozgul Kucuk
Lakshmi Kuppusamy
Junzuo Lai
Yee Wei Law
Hoi Le
Kwangsu Lee
Peter Lee
Gaëtan Leurent
Fagen Li
Jin Li
Shen Li
Tieyan Li
Kaitai Liang
Hoon Wei Lim
Richard Lindner
Zhen Liu
Jiqiang Lu
Weiliang Luo
Shah Mahmood
Cuauhtemoc Mancillas-Lopez
Pedro Maat Massolino
Seiichi Matsuda
Kirill Morozov
Shishir Nagaraja

Prasad G. Naldurg	Sumanta Sarkar	Li Xu
Ta Toan Khoa Nguyen	Yu Sasaki	Piyi Yang
Shirin Nilizadeh	Roman Schlegel	Yanjiang Yang
Ryo Nishimaki	Jae Hong Seo	Masaya Yasuda
Kazumasa Omote	Naoyuki Shinohara	Bin Zhang
Jong Hwan Park	Leonie Simpson	Hui Zhang
Seunghwan Park	Dondong Sun	Kehuan Zhang
Geovandro Pereira	Li Sun	Liangfeng Zhang
Duong Hieu Phan	Katsuyuki Takashima	Mingwu Zhang
Haifeng Qian	Xiao Tan	Rui Zhang
Hasan Qunoo	Keisuke Tanaka	Yinghui Zhang
Kenneth Radke	Fei Tang	Yun Zhang
Somindu C. Ramanna	Ashraful Tuhin	Qingji Zheng
Asha Rao	Fenghe Wang	Huafei Zhu
Arnab Roy	Lihua Wang	Youwen Zhu
Moustafa Saleh	Yongzhuang Wei	

Table of Contents

Optimal Bounds for Multi-Prime Φ-Hiding Assumption

Kaori Tosu[⋆] and Noboru Kunihiro

The University of Tokyo
Kaori_Tosu@mist.i.u-tokyo.ac.jp, kunihiro@k.u-tokyo.ac.jp

Abstract. We propose a novel attack against the Multi-Prime Φ-Hiding Problem, which was introduced by Kiltz et al. at CRYPTO 2010 to show the instantiability of RSA-OAEP. The cryptanalysis of the Multi-Prime Φ-Hiding Problem is also mentioned by them. At Africacrypt 2011, Herrmann improved their result by making use of the special structure of the polynomial that is derived from the problem instance. In his method, the bound on e is reduced by employing a linear equation with fewer variables. In order to optimize the size and number of variables, we examine every possible variable size and number of variables. Then, we show that our attack achieves a better bound than that of Herrmann, which shows that our attack is the best among all known attacks.

Keywords: Multi-Prime Φ-Hiding Assumption, RSA-OAEP, lattice based technique.

1 Introduction

1.1 Background

The RSA-OAEP encryption scheme [1] was introduced by Bellare and Rogaway in 1994. At CRYPTO 2001, Fujisaki et al. [5] discussed the security of the f-OAEP encryption scheme and of RSA-OAEP in the random oracle model, where f is an abstract trapdoor permutation involved in the OAEP conversion. They first proved that f-OAEP is IND-CCA secure if the permutation f has partial-domain one-wayness. They also showed that the RSA function has partial-domain one-wayness and then that RSA-OAEP is IND-CCA secure. In the latter part of the proof, they employed a lattice based technique.

At CRYPTO 2010, Kiltz et al. [9] discussed the security of RSA-OAEP in the standard model. They proved that RSA-OAEP is IND-CPA secure in the standard model under a reasonable assumption. In their proof, the RSA function is *lossy* under the Φ-hiding assumption if e is taken up to almost 1/4 the bit-length of N. Then, they extended the Φ-hiding assumption to the multi-prime RSA setting to introduce multi-prime Φ-hiding assumption and showed that multi-prime RSA gains more lossiness. In the evaluation of lossiness, a lattice based technique is also employed.

[⋆] Currently, she is working at Sumitomo Life Insurance Company.

The original Φ-hiding assumption was introduced by Cachin et al. at EUROCRYPT 1999 [2]. Since its proposal, many cryptographic applications based on the Φ-hiding assumption have been proposed. The Φ-Hiding assumption roughly states that given an RSA modulus $N = pq$ and a prime e it is difficult to decide if e divides $(p-1)(q-1) = \Phi(N)$ or not. If e is prime and divides $\Phi(N)$ then either e divides $p-1$ or e divides $q-1$. So, without loss of generality, we can say that the assumption states that given $N = pq$ and e it is difficult to decide if e divides $p-1$ or not.

The multi-prime Φ-hiding assumption introduced by Kiltz et al. is a natural extension of the Φ-hiding assumption. The multi-prime Φ-hiding assumption roughly states that given a multi-prime RSA modulus $N = p_1 \ldots p_m$, where $p_1, \ldots, p_m$ are distinct primes, and a prime e it is difficult to decide if e divides $p_i - 1$ for $1 \leq i \leq m-1$ or not.

Kiltz et al. perform a cryptanalysis of the Multi-Prime Φ-Hiding Assumption. First, they mention that if $e < N^{1/m - 1/m^2}$ they can solve the Multi-Prime Φ-Hiding Problem similar to the attack on the Φ-Hiding Assumption based on Howgrave-Graham's algorithm. In this paper, we call this Howgrave-Graham Method. They also notice the fact that we can solve the Multi-Prime Φ-Hiding Problem if the modular equation

$$(ex_1 - 1)(ex_2 - 1) \cdots (ex_{m-1} - 1) \equiv 0 \pmod{p_1 p_2 \ldots p_{m-1}} \tag{1}$$

can be solved. In detail, they apply a linearization for Eq. (1) by using $u_1, \ldots, u_{m-1}$. Then they solve the roots of $u_1^{(0)}, \ldots, u_{m-1}^{(0)}$ by making use of the algorithm of Herrmann and May [7]. They obtain the better bound to solve the Multi-Prime Φ-Hiding Problem than the Howgrave-Graham Method. Furthermore, at Africacrypt 2011, Herrmann [6] improved the result of Kiltz et al. He noticed the fact that we do not have to find the roots $x_1^{(0)}, x_2^{(0)}, \ldots, x_{m-1}^{(0)}$ to solve the Multi-Prime Φ-Hiding Problem. Thus, it does not matter that we perform a linearization for Eq. (1) by using 2 variables u_1, u_2. Herrmann attains the better bound to solve the Multi-Prime Φ-Hiding Problem by reducing the number of variables. However, when the number of variables is smaller, the upper bound on the size of the roots is larger. That means the bound to solve the Multi-Prime Φ-Hiding Problem is optimal when the number of variables and the size of the upper bound of the root are well balanced.

1.2 Our Contributions

In this paper, we present an algorithm to attain better bounds for solving the Multi-Prime Φ-Hiding Problem. Our algorithm is based on Herrmann [6]. In detail, we investigate various linearizations for Eq. (1) and optimize the bounds for the Multi-Prime Φ-Hiding Problem.

First, we examine two variables case. That is, we restrict to using two variables and linearizing Eq. (1) to $e^t u_1 + eu_2 + 1 \equiv 0 \pmod{p_1 p_2 \ldots p_{m-1}}$, where t $(2 \leq t \leq m-1)$ is a fixed integer. Let N^{γ_1} be the upper bound of $|u_1^{(0)}|$ and N^{γ_2} be the upper bound of $|u_2^{(0)}|$. By applying the theorem of Herrmann and May [7], we can find all small solutions of Eq. (1) if $\gamma_1 + \gamma_2 < (m-3)/m + 2(1/m)^{3/2}$. Considering the relation $ex_i^{(0)} = p_i - 1 \approx N^{1/m}$, this inequality is satisfied if $e \geq N^{\delta}, \quad \delta = 1/m - 2(1/m)^{3/2}/(t+1)$. Next, we optimize the value of t to minimize δ. Considering $2 \leq t \leq m-1$, we can easily find that δ is the smallest when $t = 2$. This is the same algorithm as Herrmann's. Thus, Herrmann's algorithm is the best when we use two variables to linearize Eq. (1).

Then, we examine the k-variables case. That is, we consider a linearization using k variables. In this case, we perform linearization for Eq. (1) and obtain $e^{t_1}u_1 + e^{t_2}u_2 + \cdots + eu_k + 1 \equiv 0 \pmod{p_1 p_2 \ldots p_{m-1}}$. By using the same approach as the case of 2 variables, we prove that this equation is efficiently solved if

$$e \geq N^{\delta}, \quad \delta = \frac{1}{m} - \frac{k}{t_1 + t_2 + \cdots + t_{k-1} + 1}\left(\frac{1}{m}\right)^{\frac{k+1}{k}}.$$

Then, we show that δ attains the smallest value

$$\delta = \frac{1}{m} - \frac{2}{k+1}\left(\frac{1}{m}\right)^{\frac{k+1}{k}}$$

when $(t_1, t_2, \ldots, t_{k-1}) = (k, k-1, \ldots, 2)$.

Finally we optimize the number of variables k. The lower bound of e attains the smallest value when $k \approx \log m$, where log is a natural logarithm. The optimized bound is approximated by

$$e \geq N^{\delta}, \quad \delta = \frac{1}{m} - \frac{2}{\mathrm{e}m(\log m + 1)},$$

where e is the base of natural logarithm. This bound is the best of all known attacks.

2 Preliminaries

In our analysis we will use a theorem of Herrmann and May [7] that gives upper bounds on the size of the solutions of a linear equation modulo an unknown divisor. As required by most multivariate applications of Coppersmith's algorithm [3,4], it relies on an assumption in order to extract the final solutions efficiently.

Assumption 1 *Let $h_1, \ldots, h_n \in \mathbb{Z}[x_1, \ldots, x_n]$ be the polynomials that are found by Coppersmith's algorithm. Then the ideal generated by the polynomial equations $h_1(x_1, \ldots, x_n) = 0, \ldots, h_n(x_1, \ldots, x_n) = 0$ has dimension zero.*

Under Assumption 1, Howgrave-Graham [8] gives upper bounds on the size of the solutions of a univariate linear equation modulo an unknown divisor.

Theorem 1 (Howgrave-Graham 2001 [8]). *Let N be a sufficiently large composite integer (of unknown factorization) with a divisor $p \geq N^{\beta}$. Let $g(x) \in \mathbb{Z}[x]$ be a univariate linear polynomial. Under Assumption 1, we can find all the solutions $x^{(0)}$ of the equation $g(x) \equiv 0 \pmod{p}$ with $|x^{(0)}| \leq N^{\beta^2}$. The time complexity of the algorithm is polynomial in $\log N$.*

Herrmann and May [7] extend Theorem 1 to multivariate linear equations.

Theorem 2 (Herrmann, May 2008 [7]). *Let $\varepsilon > 0$ and let N be a sufficiently large composite integer (of unknown factorization) with a divisor $p \geq N^{\beta}$. Furthermore, let $f(x_1, \ldots, x_n) \in \mathbb{Z}[x_1, \ldots, x_n]$ be a linear polynomial in n variables. Under Assumption 1, we can find all the solutions $(x_1^{(0)}, \ldots, x_n^{(0)})$ of the equation $f(x_1, \ldots, x_n) \equiv 0 \pmod{p}$ with $|x_1^{(0)}| \leq N^{\gamma_1}, \ldots, |x_n^{(0)}| \leq N^{\gamma_n}$ if*

$$\sum_{i=1}^{n} \gamma_i \leq 1 - (n+1)(1-\beta) + n(1-\beta)^{\frac{n+1}{n}} - \varepsilon.$$

The time complexity of the algorithm is polynomial in $\log N$ and $(\mathrm{e}/\varepsilon)^n$.

For completeness we explicitly define the Multi-Prime Φ-Hiding Problem and Multi-Prime Φ-Hiding Assumption.

Definition 1 (Multi-Prime Φ-Hiding Problem). *Let $N = p_1 p_2 \ldots p_m$ be a composite integer (of unknown factorization) with m (≥ 2) prime factors of equal bit length. Given N and a prime e, decide whether e divides p_i for $1 \leq i \leq m-1$ or not.*

Definition 2 (Multi-Prime Φ-Hiding Assumption). *There is no polynomial-time algorithm that decides the Multi-Prime Φ-Hiding Problem with non-negligible success probability.*

Note that all the analysis in the paper is based on approximations and constant terms are ignored. Furthermore, we ignore "$-\varepsilon$"terms throughout the paper.

3 Previous Works

In this section, we will present three previous methods in attacking Multi-Prime Φ-Hiding Assumption. For easy understanding, we show small examples in the Appendix.

3.1 Howgrave-Graham Method

First method is mentioned by Kiltz et al. [9] making use of the theorem of Howgrave-Graham. In this paper, we will call it "Howgrave-Graham Method".

If a prime e divides $p_i - 1$ for $i = 1, \ldots, m-1$, then e fulfills the equations

$$e x_i^{(0)} + 1 = p_i. \tag{2}$$

More generally, we can write

$$e x_i^{(0)} + 1 \equiv 0 \pmod{p_i}.$$

Now, given a prime e and a modulus N of unknown factorization, we have solved Multi-Prime Φ-Hiding Problem, if we can find the solutions $x_i^{(0)}$ of the equation

$$ex_i + 1 \equiv 0 \pmod{p_i}. \tag{3}$$

Theorem 1 tells us that we can efficiently find all the solutions $x_i^{(0)}$ of Eq. (3) if their absolute value is smaller than N^{1/m^2}. Considering Eq. (2), $|x_i^{(0)}| < N^{1/m^2}$ if $e < N^{1/m-1/m^2}$. Therefore, if

$$e \geq N^{\delta}, \quad \delta = \frac{1}{m} - \frac{1}{m^2},$$

then we can efficiently solve Multi-Prime Φ-Hiding Problem.

3.2 KOS Method

Kiltz et al. [9] proposed another method that gives a better bound on the prime e to solve Multi-Prime Φ-Hiding Problem than the Howgrave-Graham Method. We will call it "KOS Method".

Now, we demonstrate the KOS Method. We consider a modulus N consisting of m primes $p_1, \ldots, p_m$ of the same bitsize. First of all, we multiply the following equations for all $i = 1, \ldots, m-1$: $ex_i + 1 \equiv 0 \pmod{p_i}$. Then we obtain the single equation

$$\sum_{i=1}^{m-1} e^i \sigma_i(x_1, x_2, \ldots, x_{m-1}) + 1 \equiv 0 \pmod{p_1 p_2 \ldots p_{m-1}}, \tag{4}$$

where $\sigma_i(x_1, x_2, \ldots, x_{m-1})$ is the elementary symmetric polynomial of degree i, which is defined by

$$\sigma_i(x_1, x_2, \ldots, x_{m-1}) = \sum_{\lambda \subseteq \Lambda_{m-1}, |\lambda| = i} \left(\prod_{t \in \lambda} x_t \right)$$

for $\Lambda_{m-1} = \{1, 2, \ldots, m-1\}$. Next, we perform a linearization of Eq. (4) and obtain the linear equation

$$e^{m-1}u_1 + e^{m-2}u_2 + \cdots + eu_{m-1} + 1 \equiv 0 \pmod{p_1 p_2 \ldots p_{m-1}}, \tag{5}$$

where $u_i = \sigma_{m-i}(x_1, x_2, \ldots, x_{m-1})$ for $i = 1, \ldots, m-1$.

If we can solve Eq. (5), we can obtain $u_1, \ldots, u_{m-1}$. Then, we have $x_1, \ldots, x_{m-1}$ by solving an equation $X^{m-1} - u_1 X^{m-2} + \cdots + (-1)^{m-1} u_{m-1} = 0$ over the integers. Finally, we have $p_1, \ldots, p_{m-1}$ since each p_i is $p_i = ex_i + 1$. Then, if we can solve Eq. (5), we have p_i. Hence, it is enough to obtain the condition such that Eq. (5) is solvable in polynomial time.

We use Theorem 2 for obtaining such condition for Eq. (5). Let N^{γ} be the upper bound of $|x_i^{(0)}|$ for $i = 1, \ldots, m-1$. We can derive upper bounds on the $|u_i|$, namely $|u_i^{(0)}| \leq N^{(m-i)\gamma}$. By applying Theorem 2 with $\beta = (m-1)/m, n = m-1$, we can find all the solutions of Eq. (5) if

$$\gamma < \frac{2}{m}\left(\frac{1}{m}\right)^{\frac{m}{m-1}}.$$

For the prime e this means that we can efficiently solve the Multi-Prime Φ-Hiding Problem if

$$e \geq N^{\delta}, \quad \delta = \frac{1}{m} - \frac{2}{m}\left(\frac{1}{m}\right)^{\frac{m}{m-1}}.$$

For all $m (\geq 2)$, the inequality

$$\frac{2}{m}\left(\frac{1}{m}\right)^{\frac{m}{m-1}} \geq \frac{1}{m^2}$$

is satisfied. Therefore, the KOS Method is better algorithm than the Howgrave-Graham Method.

3.3 Herrmann Method

Herrmann improved upon the previous two results by making use of the special structure of the polynomial that is derived from the problem instance. If the prime e divides $p_i - 1$ for all $i = 1, \ldots, m-1$, we can write $p_i \equiv 1 \bmod e$, or more generally

$$p_1 p_2 \ldots p_{m-1} \equiv 1 \pmod{e}.$$

On the other hand, if e does not divide $p_i - 1$ for a certain i $(1 \leq i \leq m-1)$, then $p_1 p_2 \ldots p_{m-1} \bmod e$ is random. That means we can solve the Multi-Prime Φ-Hiding Problem if we decide whether $p_1 p_2 \ldots p_{m-1} \pmod{e}$ is 1 or not.

If the prime e divides $p_i - 1$ for all $i = 1, \ldots, m-1$, we can write the equation

$$p_1 p_2 \ldots p_{m-1} = \prod_{i=1}^{m-1} (e x_i^{(0)} + 1),$$

where each $x_i^{(0)} (i = 1, \ldots, m-1)$ is an integer.

It is sufficient to decide whether $p_1 p_2 \ldots p_{m-1} \bmod e$ is 1 or not by solve the Multi-Prime Φ-Hiding Problem. Actually, the solution is not important itself. It is important to decide whether the equation has solutions or not for solving Multi-Prime Φ-Hiding Problem. We do not have to explicitly compute the solution $x_1^{(0)}, \ldots, x_{m-1}^{(0)}$. Therefore, we can apply a linearization for Eq. (4) using fewer than $m-1$ variables. By reducing the number of variables, Herrmann improves the previous methods.

Similar to the KOS method, we expand Eq. (1) and obtain Eq. (4). Then, we perform another kind of linearization for Eq. (4) to obtain an equation

$$e^2 u_1 + e u_2 + 1 \equiv 0 \pmod{p_1 p_2 \ldots p_{m-1}}, \tag{6}$$

where $u_1 = \sum_{i=2}^{m-1} e^{i-2}\sigma_i(x_1, x_2, \dots, x_{m-1})$ and $u_2 = \sigma_1(x_1, x_2, \dots, x_{m-1})$. In this case, we can efficiently solve the Multi-Prime Φ-Hiding Problem if

$$e \geq N^{\delta}, \quad \delta = \frac{1}{m} - \frac{2}{3}\left(\frac{1}{m}\right)^{\frac{3}{2}}.$$

For any m (≥ 3), we have the inequality.

$$\frac{2}{3}\left(\frac{1}{m}\right)^{\frac{3}{2}} \geq \frac{2}{m}\left(\frac{1}{m}\right)^{\frac{m}{m-1}}.$$

Therefore, the Herrmann Method is better than KOS Method.

4 Our New Algorithm

In Section 3, we explained the three previous methods: Howgrave-Graham Method, KOS Method and Herrmann Method. In this section, we will present our new algorithm based on Herrmann Method in attacking the Multi-Prime Φ-Hiding Problem. Herrmann applies the linearization for Eq. (4) to obtain the two variable linear equation (6). He left more improvement in attacking the Multi-Prime Φ-Hiding Problem as future works. The reason for this conjecture is the fact that Herrmann Method does not fully exploit the relation between the coefficients of the polynomial. Therefore, we investigate every way of linearizing Eq. (4) and investigate the bounds to solve Multi-Prime Φ-Hiding Problem.

4.1 The Case of Two Variables

For the warm-up, we examine the two variables case. That is, we restrict to using two variables. Let t ($2 \leq t \leq m-1$) be an integer. Note that Herrmann's analysis corresponds to $t = 2$. We perform a linearization for Eq. (4) and obtain

$$e^t u_1 + e u_2 + 1 \equiv 0 \pmod{p_1 p_2 \dots p_{m-1}}, \tag{7}$$

where $u_1 = \sum_{i=t}^{m-1} e^{i-t}\sigma_i(x_1, x_2, \dots, x_{m-1})$ and $u_2 = \sum_{i=1}^{t-1} e^{i-1}\sigma_i(x_1, x_2, \dots, x_{m-1})$. The same analysis as used in the previous methods applies to the following. Let N^{γ} be the upper bound of $|x_i^{(0)}|$ for $i = 1, \dots, m-1$. Considering the relation $ex_i^{(0)} = p_i - 1 \approx N^{1/m}$, we can derive upper bounds on the $|u_i|$, namely

$$|u_1| \leq N^{\frac{m-t-1}{m} + t\gamma}, \quad |u_2| \leq N^{\frac{t-2}{m} + \gamma}.$$

By applying Theorem 2, we can find all the solutions of Eq. (7) if

$$\gamma < \frac{2}{t+1}\left(\frac{1}{m}\right)^{\frac{3}{2}}.$$

For the prime e this means that we can efficiently solve the Multi-Prime Φ-Hiding Problem if

$$e \geq N^{\delta}, \quad \delta = \frac{1}{m} - \frac{2}{t+1}\left(\frac{1}{m}\right)^{\frac{3}{2}}.$$

Then, we optimize the value of t to obtain the best bound of e. We can easily find that δ is the smallest when $t = 2$. Therefore, the bound to solve Multi-Prime Φ-Hiding Problem is the best when we perform a linearization for Eq. (4) using two variables and obtain

$$e^2 u_1 + e u_2 + 1 \equiv 0 \pmod{p_1 p_2 \dots p_{m-1}}.$$

This is the same as the Herrmann Method. Summing up the discussion, we have the following theorem.

Theorem 3. *The Multi-Prime Φ-Hiding Problem can be solved by finding the solutions of the equation*

$$e^2 u_1 + e u_2 + 1 \equiv 0 \pmod{p_1 p_2 \dots p_{m-1}},$$

where $u_1 = \sum_{i=t}^{m-1} e^{i-t}\sigma_i(x_1, x_2, \dots, x_{m-1})$ *and* $u_2 = \sum_{i=1}^{t-1} e^{i-1}\sigma_i(x_1, x_2, \dots, x_{m-1})$, *if e fulfills the inequality*

$$e \geq N^{\delta}, \quad \delta = \frac{1}{m} - \frac{2}{t+1}\left(\frac{1}{m}\right)^{\frac{3}{2}}.$$

The smallest value of δ is given by

$$\delta = \frac{1}{m} - \frac{2}{3}\left(\frac{1}{m}\right)^{\frac{3}{2}}$$

when $t = 2$.

4.2 The Case of k Variables

Next we consider using k (≥ 2) variables. Let $t_0, \dots, t_k$ be integers satisfying $t_k = 1 < t_{k-1} < t_{k-2} < \dots < t_1 < t_0 = m$, and j ($1 \leq j \leq k$) be an integer. We apply a linearization for Eq. (4) to obtain

$$e^{t_1} u_1 + e^{t_2} u_2 + \dots + e u_k + 1 \equiv 0 \pmod{p_1 p_2 \dots p_{m-1}}, \tag{8}$$

where

$$u_j = \sum_{i=t_j}^{t_{j-1}-1} e^{i-t_j}\sigma_i(x_1, x_2, \dots, x_{m-1})$$

for $1 \leq j \leq k$. Let N^{γ} be the upper bound of $|x_i^{(0)}|$ for $i = 1, \dots, m-1$. Considering the relation $e x_i^{(0)} = p_i - 1 \approx N^{1/m}$, we can derive upper bounds on the $|u_i|$, namely

$$|u_i| \leq N^{\frac{t_{j-1}-t_j-1}{m} + t_j \gamma}.$$

By applying Theorem 2, we can find all the solutions of Eq. (8) if

$$\gamma < \frac{k}{t_1 + t_2 + \cdots + t_{k-1} + 1} \left(\frac{1}{m}\right)^{\frac{k+1}{k}} .$$

For the prime e this means that we can efficiently solve the Multi-Prime Φ-Hiding Problem if

$$e \geq N^{\delta}, \quad \delta = \frac{1}{m} - \frac{k}{t_1 + t_2 + \cdots + t_{k-1} + 1} \left(\frac{1}{m}\right)^{\frac{k+1}{k}} .$$

Then, we optimize the value of $t_1, \ldots, t_k$ to obtain the best bound of e. We can easily find that δ attains the smallest value

$$\delta = \frac{1}{m} - \frac{2}{k+1} \left(\frac{1}{m}\right)^{\frac{k+1}{k}} , \tag{9}$$

when $(t_1, t_2, \ldots, t_{k-1}) = (k, k-1, \ldots, 2)$. This is when we perform a linearization for Eq. (4) to obtain

$$e^k u_1 + e^{k-1} u_2 + \cdots + e u_k + 1 \equiv 0 \pmod{p_1 p_2 \ldots p_{m-1}}.$$

Summing up the discussion, we have the following theorem.

Theorem 4. *Let k be an integer and let $t_0, \ldots, t_k$ be integers satisfying $t_k = 1 < t_{k-1} < t_{k-2} < \cdots < t_1 < t_0 = m$. The Multi-Prime Φ-Hiding Problem can be solved by finding the solutions of the equation*

$$e^k u_1 + e^{k-1} u_2 + \cdots + e u_k + 1 \equiv 0 \pmod{p_1 p_2 \ldots p_{m-1}},$$

where $u_j = \sum_{i=t_j}^{t_{j-1}-1} e^{i-t_j} \sigma_i(x_1, x_2, \ldots, x_{m-1})$ for $j = 1, \ldots, k$, if e holds the inequality

$$e \geq N^{\delta}, \quad \delta = \frac{1}{m} - \frac{k}{t_1 + t_2 + \cdots + t_{k-1} + 1} \left(\frac{1}{m}\right)^{\frac{k+1}{k}} .$$

δ is the smallest when $(t_1, t_2, \ldots, t_{k-1}) = (k, k-1, \ldots, 2)$. The optimized bound is

$$\delta = \frac{1}{m} - \frac{2}{k+1} \left(\frac{1}{m}\right)^{\frac{k+1}{k}} . \tag{10}$$

4.3 Optimizing k

It is worth pointing out that from the definition of our algorithm, we have Howgrave-Graham method, Herrmann method and KOS method by setting $k = 1$, $k = 2$ and $k = m - 1$, respectively. It is not clear whether the choice of $k = 1, 2, m-1$ is the best of all possible choices. Actually, this is not the case; we show the optimal choice of k.

First, we search the optimal value of k by using Eq. (10). For each positive integer $m(\leq 20)$, the optimal value of k is given as follows:

$$k = \begin{cases} 1 & (m = 2) \\ 2 & (3 \leq m \leq 5) \\ 3 & (6 \leq m \leq 14) \\ 4 & (15 \leq m \leq 20). \end{cases}$$

Next, we show an asymptotic performance of our algorithm.

Theorem 5. *Let δ be*

$$\delta = \frac{1}{m} - \frac{2}{em(\log m + 1)}.$$

The Multi-Prime Φ-Hiding Problem can be solved by finding the solutions of the equation

$$e^{k_0} u_1 + e^{k_0 - 1} u_2 + \cdots + e u_{k_0} + 1 \equiv 0 \pmod{p_1 p_2 \ldots p_{m-1}}.$$
$$u_j = \sum_{i=t_j}^{t_{j-1}-1} e^{i - t_j} \sigma_i(x_1, x_2, \ldots, x_{m-1}), \quad (j = 1, \ldots, k_0),$$

if e satisfies the inequality $e \geq N^\delta$ for large m.

Proof. Let

$$f(x) := \frac{1}{m} - \frac{2}{x+1} \left(\frac{1}{m}\right)^{\frac{x+1}{x}}.$$

We differentiate $f(x)$ and obtain

$$f'(x) = \frac{2}{x^2(x+1)^2} \left(\frac{1}{m}\right)^{\frac{x+1}{x}} \left\{x^2 - (x+1)\log m\right\}.$$

When x is a positive real number, $f(x)$ is the smallest when

$$x = \frac{\log m + \sqrt{(\log m)^2 + 4 \log m}}{2}. \qquad (11)$$

Let α be the right hand of Eq. (11). For any positive integer m, α always satisfies the inequality

$$\log m < \alpha < \log m + 1.$$

Therefore, when x is a positive integer, $f(x)$ is the smallest when $x = \lceil \log m \rceil$ or $\lfloor \log m \rfloor$. For large m, $\lceil \log m \rceil \approx \lfloor \log m \rfloor \approx \log m$ and

$$f(\log m) = \frac{1}{m} - \frac{2}{m(\log m + 1)} \left(\frac{1}{m}\right)^{\frac{1}{\log m}}.$$

By using the equation

$$\left(\frac{1}{m}\right)^{\frac{1}{\log m}} = \frac{1}{\mathrm{e}},$$

The optimized value is approximated as follows:

$$f(\log m) = \frac{1}{m} - \frac{2}{\mathrm{e}m(\log m + 1)}.$$

Then, we have the theorem. □

Now, we consider the time complexity. Theorem 2 tells us the time complexity to solve the Multi-Prime Φ-Hiding Problem is exponential in the number of variables n in the target linear equation. That is, it is roughly estimated by e^{cn} for some constant c. If we use our method, n is set to $n \approx \log m$ and the time complexity is evaluated by $\mathrm{e}^{c \log m} = m^c$. This implies that our algorithm works in polynomial time in m. Like our algorithm, the Howgrave-Graham Method and Herrmann Method work also in polynomial time in m. On the other hand, the KOS method needs exponential time in m. Since our algorithm involves more variables than the Herrmann Method, the time complexity of our method is larger than that of the Herrmann Method.

4.4 Discussions

Figure 1 compares the result value of e to efficiently solve the Multi-Prime Φ-Hiding Problem. Note that all values of e that lie above the respective lines are vulnerable to a polynomial time factorization attack. It is clearly verified that our new attack solves the Multi-Prime Φ-Hiding Problem for smaller values of e than the other methods. For $m = 3, 4, 5$, our method uses two variables ($k = 2$), which is the same as Herrmann Method. It is remarkable that the result value of ours is really smaller than that of Herrmann if $m \geq 6$. For example, with 4096 bits modulus and $m = 10$ we reduce the size of e from 369 bits (Howgrave-Graham), 356 bits (KOS), 323 bits (Herrmann) to 314 bits. Our attack is applicable to smaller choice of e than the other methods.

Next, we discuss asymptotic performance. For all four methods described in this paper the values $m\delta$ converge to 1 as m goes to ∞. However, their rates of convergence are significantly different. As easily verified, it is desirable that $m\delta$ is smaller and that the convergent rate is slower. The convergent rate of our proposed algorithm is $O(1/\log(m))$; while that of Howgrave-Graham method is $O(1/m)$ and that of Herrmann method is $O(1/m^{1/2})$. Then, the value $m\delta$ of our method goes to 1 much slower than the other methods.

Note that our algorithm is effective only when e is close to the size of the primes and m is large. In other words, the (Multi-Prime) Φ-hiding problem remains still hard in general.

4.5 Full Description of Our Algorithm

For completeness, we show the full description of our algorithm for solving Multi-Prime Φ-Hiding Problem.

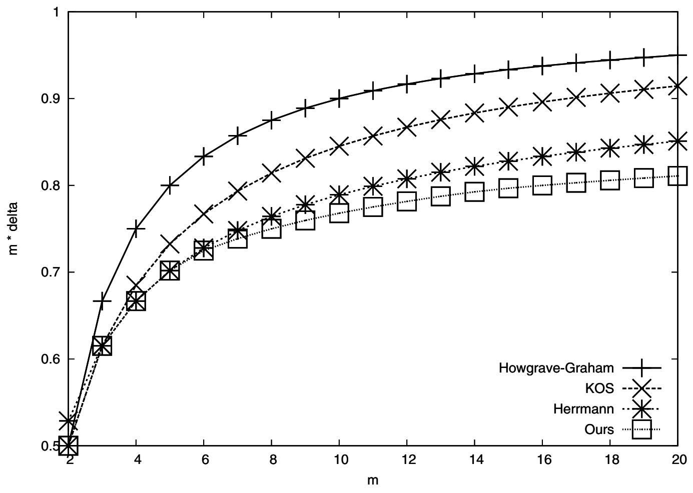


Fig. 1. Comparison of attacks on the Multi-Prime Φ-Hiding Problem

Input: Public key (N, e) and the number m of prime factors of N
Output: Decide whether e divides p_i for $1 \leq i \leq m-1$ or not.
Step1: Choose an optimal value k, which depends on m.
Step2: Solve the linear modular equation:

$$e^k u_1 + e^{k-1} u_2 + \cdots + e u_k + 1 \equiv 0 \pmod{p_1 p_2 \ldots p_{m-1}}$$

by using Herrmann–May's algorithm with auxiliary inputs $X_i = N^{(k+1-i)\gamma}$ for $1 \leq i \leq k$, where $\gamma = 2/(k+1)(1/m)^{(k+1)/k}$. Note that we need not know the value: $p_1 p_2 \ldots p_{m-1}$.
Step3: If the equation has the solution, output *yes*. Otherwise, output *no*.

Acknowledgement. We would like to thank Steven Galbraith for improving our paper and the anonymous reviewers for helpful comments. The second author was supported by KAKENHI 22700006.

References

1. Bellare, M., Rogaway, P.: Optimal Asymmetric Encryption. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 92–111. Springer, Heidelberg (1995)
2. Cachin, C., Micali, S., Stadler, M.A.: Computationally Private Information Retrieval with Polylogarithmic Communication. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 402–414. Springer, Heidelberg (1999)
3. Coppersmith, D.: Finding a Small Root of a Bivariate Integer Equation; Factoring with High Bits Known. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 178–189. Springer, Heidelberg (1996)

4. Coppersmith, D.: Small solutions to polynomial equations, and low exponent rsa vulnerabilities. J. of Cryptology 10(4), 233–260 (1997)
5. Fujisaki, E., Okamoto, T., Pointchval, D., Stern, J.: Rsa-oaep is secure under the rsa assumption. J. of Cryptology 17(2), 81–104 (2004)
6. Herrmann, M.: Improved Cryptanalysis of the Multi-Prime ϕ - Hiding Assumption. In: Nitaj, A., Pointcheval, D. (eds.) AFRICACRYPT 2011. LNCS, vol. 6737, pp. 92–99. Springer, Heidelberg (2011)
7. Herrmann, M., May, A.: Solving Linear Equations Modulo Divisors: On Factoring Given Any Bits. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 406–424. Springer, Heidelberg (2008)
8. Howgrave-Graham, N.: Approximate Integer Common Divisors. In: Silverman, J.H. (ed.) CaLC 2001. LNCS, vol. 2146, pp. 51–66. Springer, Heidelberg (2001)
9. Kiltz, E., O'Neill, A., Smith, A.: Instantiability of RSA-OAEP under Chosen-Plaintext Attack. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 295–313. Springer, Heidelberg (2010)

A Small Examples

For easy understanding, we show some small examples for KOS method and Herrmann method.

A.1 A Small Example for KOS Method

We consider a modulus N consisting of four primes p_1, p_2, p_3, p_4 of equal bitsize. First of all, we multiply all the equations

$$ex_1 + 1 \equiv 0 \pmod{p_1}, \quad ex_2 + 1 \equiv 0 \pmod{p_2}, \quad ex_3 + 1 \equiv 0 \pmod{p_3},$$

and obtain the single equation

$$e^3 \underbrace{x_1x_2x_3}_{u_1} + e^2(\underbrace{x_1x_2 + x_2x_3 + x_3x_1}_{u_2}) + e(\underbrace{x_1 + x_2 + x_3}_{u_3}) + 1 \equiv 0 \pmod{p_1p_2p_3}. \tag{12}$$

Next, we perform a linearization of Eq. (12) to obtain the linear equation

$$e^3u_1 + e^2u_2 + eu_3 + 1 \equiv 0 \pmod{p_1p_2p_3}, \tag{13}$$

where $u_1 = x_1x_2x_3$, $u_2 = x_1x_2 + x_2x_3 + x_3x_1$ and $u_3 = x_1 + x_2 + x_3$.

If we can solve Eq. (13), we can obtain u_1, u_2 and u_3. Then, we have x_1, x_2 and x_3 by solving an equation $X^3 - u_1X^2 + u_2X - u_3 = 0$ over integers. Finally, we have p_1, p_2 and p_3 since each p_i is $p_i = ex_i + 1$. Then, if we can solve Eq. (13), we have p_i. Hence, it is enough to obtain the condition such that Eq. (13) is solvable in polynomial time.

We use Theorem 2 to obtain such condition for Eq. (13). Let N^γ be the upper bound of $|x_i^{(0)}|$ for $i = 1, 2, 3$. We can derive upper bounds on the $|u_i|$, namely $|u_1^{(0)}| \leq N^{3\gamma}, |u_2^{(0)}| \leq N^{2\gamma}, |u_3^{(0)}| \leq N^{\gamma}$. By applying Theorem 2 with $\beta = 3/4, n = 3$, we can find all the solutions of Eq. (13) if

$$\gamma < \frac{1}{2}\left(\frac{1}{4}\right)^{\frac{4}{3}}.$$

For the prime e this means that we can efficiently solve the Multi-Prime Φ-Hiding Problem if

$$e \geq N^{\delta}, \quad \delta = \frac{1}{4} - \frac{1}{2}\left(\frac{1}{4}\right)^{\frac{4}{3}}.$$

A.2 A Small Example for Herrmann Method

We consider a modulus N consisting of four primes p_1, p_2, p_3, p_4 of equal bitsize. First, we expand Eq. (1) and obtain

$$\underbrace{e^3 x_1 x_2 x_3 + e^2(x_1 x_2 + x_2 x_3 + x_3 x_1)}_{e^2 u_1} + e(\underbrace{x_1 + x_2 + x_3}_{u_2}) + 1 \equiv 0 \pmod{p_1 p_2 p_3}. \tag{14}$$

Then, we perform a linearization for Eq. (14) to obtain the equation

$$e^2 u_1 + e u_2 + 1 \equiv 0 \pmod{p_1 p_2 p_3}, \tag{15}$$

where $u_1 = e x_1 x_2 x_3 + x_1 x_2 + x_2 x_3 + x_3 x_1$ and $u_2 = x_1 + x_2 + x_3$. Similar to the KOS Method, Eq. (15) (and thus the Multi-Prime Φ-Hiding Problem) can be efficiently solved if

$$e \geq N^{\delta}, \quad \delta = \frac{1}{4} - \frac{1}{12} = \frac{1}{6}.$$

Sufficient Condition for Ephemeral Key-Leakage Resilient Tripartite Key Exchange

Atsushi Fujioka[1], Mark Manulis[2], Koutarou Suzuki[1], and Berkant Ustaoğlu[3]

[1] NTT Secure Platform Laboratories
3-9-11 Midori-cho Musashino-shi Tokyo 180-8585, Japan
{fujioka.atsushi,suzuki.koutarou}@lab.ntt.co.jp
[2] University of Surrey, Guildford, Surrey, GU2 7XH, United Kingdom
mark@manulis.eu
[3] Izmir Institute of Technology, Urla, Izmir, 35430, Turkey
bustaoglu@uwaterloo.ca

Abstract. Tripartite (Diffie-Hellman) Key Exchange (3KE), introduced by Joux (ANTS-IV 2000), represents today the only known class of group key exchange protocols, in which computation of unauthenticated session keys requires one round and proceeds with minimal computation and communication overhead. The first one-round authenticated 3KE version that preserved the unique efficiency properties of the original protocol and strengthened its security towards resilience against leakage of ephemeral (session-dependent) secrets was proposed recently by Manulis, Suzuki, and Ustaoglu (ICISC 2009).

In this work we explore sufficient conditions for building such protocols. We define a set of *admissible polynomials* and show how their construction generically implies 3KE protocols with the desired security and efficiency properties. Our result generalizes the previous 3KE protocol and gives rise to many new authenticated constructions, all of which enjoy forward secrecy and resilience to ephemeral key-leakage under the gap Bilinear Diffie-Hellman assumption in the random oracle model.

1 Introduction

Key Exchange (KE) protocols are crucial research topics with direct practical applications. Although KE was introduced back in 1976 [17], it was not until 1993 when Bellare and Rogaway [7] made the first step towards capturing the security requirements for these protocols in a formal way. Research efforts on provable security in KE protocols, in the public key setting, focused on two-party KE (2KE), e.g. [14,24,15,25,32,37,16], and group KE (GKE), e.g. [10,29,22,12,31,19], reaching out to other flavors such as password-based solutions [8,6,9,3] or flexible combinations of GKE and 2KE [30,1]. The security notion, shared by most KE flavors, takes its roots in [7] and is called authenticated key exchange (AKE) security. Although AKE-security has been modeled for different types of adversaries, the common idea for secure key exchange is *indistinguishability* of a test session key from a randomly chosen one.

Tripartite Key Exchange. A powerful GKE subclass of tripartite KE (3KE) emerged with the use of pairings in the work of Joux [20], where one communication round amongst three parties is sufficient to compute the session key. Each party communicates only one group element and performs one exponentiation and one pairing evaluation. The original protocol in [20] was unauthenticated and so efforts were taken to achieve protection against active attacks, without sacrificing the efficiency of the protocol. Adopting traditional authentication techniques such as digital signatures, as previously applied to unauthenticated 2KE Diffie-Hellman in [14] or GKE in [21,13,22], would require at least two rounds of communication to prevent replay attacks. 3KE protocols with at least two communication rounds have also been known in other authentication settings, e.g. with passwords [2]. The only way to preserve one communication round with constant bit communication complexity from [20] is to resort to an implicitly authenticated solution, in which session key is derived through a mixing of static (long-term) and ephemeral (session-dependent) secrets. Many attempts to achieve such authentication in 3KE, e.g. [36,4,28,27,26] failed (as detailed in [31]). So far the only implicitly authenticated 3KE protocol that provably fulfills this goal is by Manulis, Suzuki, and Ustaoglu [31].

Ephemeral Key Leakage. The security model from [31], stated in a more general GKE setting, considers a very strong attacker, that may adaptively compromise static and ephemeral secret keys used in the protocol sessions (with the restriction that at least one key per participant remains secret). Leakage of ephemeral secrets, typically the exponent used in computing the ephyemeral Diffie-Hellman key, could be damaging for implicitly authenticated protocols, where for better efficiency one may desire to pre-compute store ephemeral public keys off-line. Even if ephemeral secret keys are chosen (and erased) within the protocol session, attacks exploiting side-channels may threaten their secrecy. In general, motivation for considering leakage of ephemeral secrets in KE protocols stems from 2KE domain, e.g. as first mentioned in [14,24] and explicitly modeled in AKE-security definitions from [25,37]. Various efforts towards construction of 2KE leakage-resilient protocols have been taken, e.g. [25,34,37,33,23,18]. In general, modeling and designing ephemeral key-leakage resilient KE protocols should not be taken for granted — Cremers [16] demonstrated how various technical elements of 2KE models such as the notions of session ids and partnering as well as conditions for freshness of the test session may affect the strength of AKE-security definition with ephemeral key-leakage resilience, when it comes to comparability of models and 2KE protocols. The model in [31] is so-far the only GKE security model that focuses on ephemeral key-leakage in test sessions and has recently been applied in [38], for the analysis of a two-round explicitly authenticated ephemeral key-leakage resilient GKE protocol.

Sufficient Condition for Ephemeral Key-Leakage Resilience. Most of KE designs focus on concrete constructions, aiming to achieve particular security goals. Some goals can be obtained generically, using protocol compilers such as to add authentication (without ephemeral key leakage-resilience) to unauthenticated KE protocols [22,13] or to obtain optional insider security [21,11,19] in GKE protocols. Yet another interesting direction is to search for sufficient conditions for achieving a security goal. Only recently, and for 2KE protocols only, sufficient conditions for ephemeral key-leakage resilience (in the eCK model [25]) have been identified by Fujioka and Suzuki [18]. Their key observation is that many eCK-secure implicitly authenticated 2KE protocols derive session keys from a shared secret group element of the form g^z, where g is the generator of a cycling group of prime order q and the exponent $z \in \mathbb{Z}_q^*$ can often be represented as a function that "mixes" products of static and ephemeral private keys. The authors introduced the concept of *admissible* polynomials over $\mathbb{Z}_q$ to describe which representations of z admit AKE-secure 2KE protocols in the eCK model, by offering a general reduction algorithm to the gap Diffie-Hellman (GDH) [35] problem (in the random oracle model). They could explain constructions of existing eCK-secure 2KE protocols and design new more efficient protocols. The beauty of their approach is that instead of designing an eCK-secure 2KE protocol from scratch it suffices to come up with a set of admissible polynomials.

Our Contributions. We identify sufficient conditions for ephemeral key-leakage resilience of (implicitly authenticated) one-round 3KE protocols, that is conditions under which the protocol can achieve AKE-security (with forward secrecy) from [31]. Technically, we build on the work from [18] and adopt their notion of *admissible* polynomials. The main difference to the 2KE case is that we work with three parties and that one-round 3KE protocols generally require bilinear maps, and hence our definition of "admissible" is different. In particular, our admissible polynomials are of degree three and involve six variables as opposed to polynomials of degree two and four variables from [18]. We show that our conditions on such polynomials are sufficient by providing a generic framework for the design of implicitly authenticated one-round 3KE protocols with ephemeral key-leakage resilience and forward secrecy in the model from [31] under the gap Bilinear Diffie-Hellman (gap BDH) assumption [5] in the random oracle model. This framework explains the 3KE protocol from [31] and gives rise to many further 3KE protocols, all of which are resilient to the leakage of ephemeral session secrets and enjoy forward secrecy.

2 The Model and Security Definitions

We recall the security model from [31], termed *g-eCK model*. This model extends strongly-authenticated key exchange model for two-party protocols from [32] to the group setting and it is described using notations and terminology of the state-of-the-art GKE model [19].

Protocol Participants and Initialization. Let $\mathcal{U} := \{U_1, \ldots, U_N\}$ be a set of potential protocol participants and each user $U_i \in \mathcal{U}$ is assumed to hold a static private/public key pair (s_i, S_i) generated by some algorithm $Gen(1^\kappa)$ on a security parameter 1^κ during the initialization phase.

Protocol Sessions and Instances. Any subset of $\mathcal{U}$ can decide at any time to execute a new protocol session and establish a common group key. Participation of some $U \in \mathcal{U}$ in multiple sessions is modeled through a number of *instances* $\{\Pi_U^s \mid s \in [1 \ldots n], U \in \mathcal{U}\}$, i.e., the Π_U^s is the s-th session of U. Each instance is invoked via a message to U with a *partner id*[1] $\mathtt{pid}_U^s \subseteq \mathcal{U}$, which encompasses the identities of all the intended session participants (note that $\mathtt{pid}_U^s$ also includes U). Then, we say that U owns the instance Π_U^s. In the invoked session, Π_U^s *accepts* if the protocol execution was successful, in particular Π_U^s holds then the computed *group key* K_U^s.

Session State. During the session execution, each participating Π_U^s creates and maintains a *session id* $\mathtt{sid}_U^s$ and an associated internal state $\mathtt{state}_U^s$ which in particular is used to maintain ephemeral secrets used by Π_U^s during the protocol execution. We say that U *owns* session $\mathtt{sid}_U^s$ if the instance Π_U^s was invoked at U. Note that the integer s is an internal parameter in the model, used to differentiate amongst the invoked sessions at U, since at the onset of the instance, there may not be enough information to create $\mathtt{sid}_U^s$; until $\mathtt{sid}_U^s$ is created, the instance is identified via $\mathtt{pid}_U^s$ and the outgoing ephemeral public key which is unique per user except with negligible probability. Furthermore, we assume that instances that accepted or aborted delete all information in their respective states.

Partnering. Two instances Π_U^s and $\Pi_{U_*}^t$ are called *partnered* or *matching* if $\mathtt{sid}_U^s \subseteq \mathtt{sid}_{U_*}^t$ or $\mathtt{sid}_{U_*}^t \subseteq \mathtt{sid}_U^s$ and $\mathtt{pid}_U^s = \mathtt{pid}_{U_*}^t$. The first condition models the fact that if session ids are computed during the protocol execution, e.g., from the exchanged messages, then their equality should be guaranteed only at the end of the protocol, i.e., upon the acceptance of Π_U^s and $\Pi_{U_*}^t$.

Note also that the notion of partnering is self-inclusive in the sense that any Π_U^s is partnered with itself. If the protocol allows a user U to initiate sessions with U, then the equality $\mathtt{pid}_U^s = \mathtt{pid}_{U_*}^t$ is a multi-set equality.

Adversarial Model. The adversary $\mathcal{A}$, modeled as a PPT machine, can schedule the protocol execution and mount own attacks via the following queries:

- $\mathsf{AddUser}(U, S_U)$: This query allows $\mathcal{A}$ to introduce new users. In response, if $U \notin \mathcal{U}$ (due to the uniqueness of identities) then U with the static public key S_U is added to $\mathcal{U}$; Note that $\mathcal{A}$ is not required to prove the possession of the corresponding secret key s_U[2].
- $\mathsf{Send}(\Pi_U^s, m)$: With this query, $\mathcal{A}$ can deliver a message m to Π_U^s whereby U denotes the identity of its sender. $\mathcal{A}$ is then given the protocol message generated by Π_U^s in response to m (the output may also be empty if m is not required or if Π_U^s accepts). A special invocation query of the form $\mathsf{Send}(U, (\text{'}\mathsf{start}\text{'}, U_1, \ldots, U_n))$ with $U \in \{U_1, \ldots, U_n\}$ creates a new instance

[1] Invocation should include the order of users and perhaps some additional information.

[2] In our security argument, we will only assume that S_U chosen by $\mathcal{A}$ must come from the ephemeral public key space, e.g., element of $\mathbb{G}$.

Π_U^s with $\mathtt{pid}_U^s = \{U_1, \ldots, U_n\}$ and provides $\mathcal{A}$ with the first protocol message.

– SessionKeyReveal(Π_U^s): This query models the leakage of session group keys and provides $\mathcal{A}$ with K_U^s. It is answered only if Π_U^s has accepted.
– StaticKeyReveal(U): This query provides $\mathcal{A}$ with the static private key s_U.
– StateReveal(Π_U^s): $\mathcal{A}$ is given the ephemeral secret information contained in $\mathtt{state}_U^s$ at the moment the query is asked. Note that the protocol specifies what the state contains.
– Test(Π_U^s): This query models the indistinguishability of the session group key according to the privately flipped bit τ. If $\tau = 0$ then $\mathcal{A}$ is given a random session group key, whereas if $\tau = 1$ the real K_U^s. The query can be queried only once and requires that Π_U^s has accepted.

Correctness. A GKE protocol is said to be *correct* if in the presence of a benign[3] adversary all instances invoked for the same protocol session accept with the same session group key.

Freshness. The classical notion of freshness of some instance Π_U^s is traditionally used to define the goal of AKE-security by specifying the conditions for the Test(Π_U^s) query. For example, the model in [22] defines an instance Π_U^s that has accepted as fresh if none of the following is true: (1) at some point, $\mathcal{A}$ asked SessionKeyReveal to Π_U^s or to any of its partnered instances; or (2) a query StaticKeyReveal(U_*) with $U_* \in \mathtt{pid}_U^s$ was asked before a Send query to Π_U^s or any of its partnered instances.

Unfortunately, these restrictions are not sufficient for our purpose since Π_U^s becomes immediately unfresh if the adversary gets involved into the protocol execution via a Send query after having learned the static key s_{U_*} of some user U_* those instance participates in the same session as Π_U^s.

The recent model in [12] defines freshness using the additional AddUser and StateReveal queries as follows. According to [12], an instance Π_U^s that has accepted is fresh if none of the following is true: (1) $\mathcal{A}$ queried AddUser(U_M, S_{U_M}) with some $U_* \in \mathtt{pid}_U^s$; or (2) at some point, $\mathcal{A}$ asked SessionKeyReveal to Π_U^s or any of its partnered instances; or (3) a query StaticKeyReveal(U_*) with $U_* \in \mathtt{pid}_U^s$ was asked before a Send query to Π_U^s or any of its partnered instances; or (4) $\mathcal{A}$ queried StateReveal to Π_U^s or any of its partnered instances at some point after their invocation but before their acceptance.

Although this definition is already stronger than the one in [22] it is still insufficient for the main reason that it excludes the leakage of ephemeral secrets of instances in the period between the protocol invocation and acceptance. Also this definition of freshness does not model key compromise impersonation attacks.

The recent update of the freshness notion in [19] addressed the lack of key compromise impersonation resilience. In particular, it modifies the above condition (3) by requiring that if there exists an instance $\Pi_{U_*}^t$ which is partnered

[3] Benign adversary executes an instance of the protocol and faithfully delivers messages without any modification.

with Π_U^s and $\mathcal{A}$ asked $\mathsf{StaticKeyReveal}(U_*)$ then all messages sent by $\mathcal{A}$ to Π_U^s on behalf of $\Pi_{U_*}^t$ must come from $\Pi_{U_*}^t$ intended for Π_U^s. This condition should allow the adversary to obtain static private keys of users prior to the execution of the attacked session while requiring its benign behavior with respect to the corrupted user during the attack.

Yet, this freshness requirement still prevents the adversary from obtaining ephemeral secrets of participants during the attacked session. What is needed is a freshness condition that would allow the adversary to corrupt users and reveal the ephemeral secrets used by their instances in the attacked session at will for the only exception that it does not obtain both the static key s_{U_*} *and* the ephemeral secrets used by the corresponding instance of U_*; otherwise security can no longer be guaranteed. In the following we define freshness taking into account all the previously mentioned problems.

Definition 1. *An accepted instance Π_U^s is fresh if none of the following is true:*

1. *$\mathcal{A}$ queried* $\mathsf{AddUser}(U_*, S_{U_*})$ *with some $U_* \in \mathtt{pid}_U^s$; or*
2. *$\mathcal{A}$ queried* $\mathsf{SessionKeyReveal}$ *to Π_U^s or any of its accepted partnered instances; or*
3. *$\mathcal{A}$ queried both* $\mathsf{StaticKeyReveal}(U_*)$ *with $U_* \in \mathtt{pid}_U^s$ and* $\mathsf{StateReveal}(\Pi_{U_*}^t)$ *for some instance $\Pi_{U_*}^t$ partnered with Π_U^s; or*
4. *$\mathcal{A}$ queried* $\mathsf{StaticKeyReveal}(U_*)$ *with $U_* \in \mathtt{pid}_U^s$ and there exists no instance $\Pi_{U_*}^t$ partnered with Π_U^s.*

Note that since $U \in \mathtt{pid}_U^s$ and since the notion of partnering is self-inclusive Condition 3 prevents the simultaneous corruption of static and ephemeral secrets for the corresponding instance Π_U^s as well. In case when users are allowed to own two partnering instances i.e., they can initiate protocols with themselves the last condition should be modified to say that the number of instances of U equals the number of times U appears in $\mathtt{pid}_U^s$. Note also that the above definition captures key-compromise impersonation resilience through Condition 4: $\mathcal{A}$ is allowed to corrupt participants of the test session in advance but then must ensure that instances of such participants have been honestly participating in the test session. In this way we exclude the trivial break of security where $\mathcal{A}$ reveals static keys of users prior to the test session and then actively impersonates those users during the session. On the other hand, as long as $\mathcal{A}$ remains benign with respect to such users their instances will still be considered as fresh.

Definition 2 (g-eCK Security). *Let* P *be a correct GKE protocol and τ be a uniformly chosen bit. We define the adversarial game* $\mathsf{Game}_{\mathcal{A},\mathsf{P}}^{\text{ake-}\tau}(\kappa)$ *as follows: after initialization, $\mathcal{A}$ interacts with instances via queries. At some point, $\mathcal{A}$ queries* $\mathsf{Test}(\Pi_U^s)$*, and continues own interaction with the instances until it outputs a bit τ'. If Π_U^s to which the* Test *query was asked is* fresh *at the end of the experiment then we set* $\mathsf{Game}_{\mathcal{A},\mathsf{P}}^{\text{ake-}\tau}(\kappa) = \tau'$*. We define*

$$\mathsf{Adv}_{\mathcal{A},\mathsf{P}}^{\text{ake}}(\kappa) = |2\Pr[\tau = \tau'] - 1|$$

and denote with $\mathsf{Adv}^{\mathrm{ake}}_{\mathsf{P}}(\kappa)$ *the maximum advantage over all PPT adversaries* $\mathcal{A}$. *We say that a GKE protocol* P *provides* g-eCK security *if this advantage is negligible.*

3 Sufficient Condition for Secure Tripartite Protocols

We identify now sufficient conditions for a 3KE protocol to satisfy g-eCK security from Definition 2. Technically, we build upon [18] and their notion of *admissible* polynomials. We extend definition of admissible polynomials to account for the specifics of 3KE protocols and then present a framework for the generic design of g-eCK secure one-round 3KE protocols out of those polynomials.

3.1 Admissible Polynomials

We define *admissible* polynomials in Definition 3 with respect to multivariate polynomials with six variables over $\mathbb{Z}_q$ and state three conditions that, as we will see, are sufficient for building g-eCK secure one-round 3KE protocols. The first condition from Definition 3 says that each term of polynomial $p^{(i)}$ has degree three and that either u_0 or u_1, either v_0 or v_1, and either w_0 or w_1 appear in each term. The second condition says that there exist four polynomials $p^{(i)}, p^{(j)}, p^{(k)}, p^{(l)}$ such that the four corresponding vectors of the coefficients of their terms containing a specific variable, are linearly independent. The third condition says that for each polynomial $p^{(i)}$ the corresponding polynomial, which consists of the terms containing specific variables, is a product of three linear polynomials.

Definition 3 (Admissible Polynomials). *We say* m $(m \geq 4)$ *polynomials* $p^{(i)} \in \mathbb{Z}_q[u_0, u_1, v_0, v_1, w_0, w_1]$ $(i = 1, ..., m)$ *are* admissible *if the following conditions are satisfied.*

1. *For any* i $(= 1, ..., m)$, *the following condition holds*

$$p^{(i)}(u_0, u_1, v_0, v_1, w_0, w_1) = \sum_{\alpha,\beta,\gamma=0,1} d^{(i)}_{\alpha,\beta,\gamma} u_\alpha v_\beta w_\gamma,$$

   where $d^{(i)}_{\alpha,\beta,\gamma} \in \mathbb{Z}_q$.
2. *We denote*

$$V^{(i)}_{\alpha,*,*} = (d^{(i)}_{\alpha,0,0}, d^{(i)}_{\alpha,0,1}, d^{(i)}_{\alpha,1,0}, d^{(i)}_{\alpha,1,1}),$$

$$V^{(i)}_{*,\beta,*} = (d^{(i)}_{0,\beta,0}, d^{(i)}_{0,\beta,1}, d^{(i)}_{1,\beta,0}, d^{(i)}_{1,\beta,1}),$$

$$V^{(i)}_{*,*,\gamma} = (d^{(i)}_{0,0,\gamma}, d^{(i)}_{0,1,\gamma}, d^{(i)}_{1,0,\gamma}, d^{(i)}_{1,1,\gamma}).$$

   For any α $(= 0, 1)$, *there exist distinct indices* i, j, k, l $(1 \leq i < j < k < l \leq m)$, *s.t.*

$$V^{(i)}_{\alpha,*,*},\ V^{(j)}_{\alpha,*,*},\ V^{(k)}_{\alpha,*,*},\ V^{(l)}_{\alpha,*,*}$$

are linearly independent, and for any β $(= 0, 1)$, *there exist distinct indices* i, j, k, l $(1 \leq i < j < k < l \leq m)$, *s.t.*

$$V^{(i)}_{*,\beta,*},\ V^{(j)}_{*,\beta,*},\ V^{(k)}_{*,\beta,*},\ V^{(l)}_{*,\beta,*}$$

are linearly independent, and for any γ $(= 0, 1)$, *there exist distinct indices* i, j, k, l $(1 \leq i < j < k < l \leq m)$, *s.t.*

$$V^{(i)}_{*,*,\gamma},\ V^{(j)}_{*,*,\gamma},\ V^{(k)}_{*,*,\gamma},\ V^{(l)}_{*,*,\gamma}$$

are linearly independent.

3. *We denote*

$$P^{(i)}_{\alpha,*,*} = d^{(i)}_{\alpha,0,0}u_\alpha v_0 w_0 + d^{(i)}_{\alpha,0,1}u_\alpha v_0 w_1 + d^{(i)}_{\alpha,1,0}u_\alpha v_1 w_0 + d^{(i)}_{\alpha,1,1}u_\alpha v_1 w_1,$$

$$P^{(i)}_{*,\beta,*} = d^{(i)}_{0,\beta,0}u_0 v_\beta w_0 + d^{(i)}_{0,\beta,1}u_0 v_\beta w_1 + d^{(i)}_{1,\beta,0}u_1 v_\beta w_0 + d^{(i)}_{1,\beta,1}u_1 v_\beta w_1,$$

$$P^{(i)}_{*,*,\gamma} = d^{(i)}_{0,0,\gamma}u_0 v_0 w_\gamma + d^{(i)}_{0,1,\gamma}u_0 v_1 w_\gamma + d^{(i)}_{1,0,\gamma}u_1 v_0 w_\gamma + d^{(i)}_{1,1,\gamma}u_1 v_1 w_\gamma.$$

For any i $(= 1, ..., m)$, *the following condition holds: for any* α $(= 0, 1)$, $P^{(i)}_{\alpha,*,*}$ *is expressed as*

$$P^{(i)}_{\alpha,*,*} = \ell^{(i)}_{\alpha,*,*}(u_0, u_1)\ell'^{(i)}_{\alpha,*,*}(v_0, v_1)\ell''^{(i)}_{\alpha,*,*}(w_0, w_1),$$

where $\ell^{(i)}_{\alpha,*,*}(u_0, u_1)$, $\ell'^{(i)}_{\alpha,*,*}(v_0, v_1)$, $\ell''^{(i)}_{\alpha,*,*}(w_0, w_1)$ *are linear combinations of* (u_0, u_1), (v_0, v_1), (w_0, w_1), *respectively, and for any* β $(= 0, 1)$, $P^{(i)}_{*,\beta,*}$ *is expressed as*

$$P^{(i)}_{*,\beta,*} = \ell^{(i)}_{*,\beta,*}(u_0, u_1)\ell'^{(i)}_{*,\beta,*}(v_0, v_1)\ell''^{(i)}_{*,\beta,*}(w_0, w_1),$$

where $\ell^{(i)}_{*,\beta,*}(u_0, u_1)$, $\ell'^{(i)}_{*,\beta,*}(v_0, v_1)$, $\ell''^{(i)}_{*,\beta,*}(w_0, w_1)$ *are linear combinations of* (u_0, u_1), (v_0, v_1), (w_0, w_1), *respectively, and for any* γ $(= 0, 1)$, $P^{(i)}_{*,*,\gamma}$ *is expressed as*

$$P^{(i)}_{*,*,\gamma} = \ell^{(i)}_{*,*,\gamma}(u_0, u_1)\ell'^{(i)}_{*,*,\gamma}(v_0, v_1)\ell''^{(i)}_{*,*,\gamma}(w_0, w_1),$$

where $\ell^{(i)}_{*,*,\gamma}(u_0, u_1)$, $\ell'^{(i)}_{*,*,\gamma}(v_0, v_1)$, $\ell''^{(i)}_{*,*,\gamma}(w_0, w_1)$ *are linear combinations of* (u_0, u_1), (v_0, v_1), (w_0, w_1), *respectively.*

In Section 3.2 we construct a g-eCK secure 3KE protocol from admissible polynomials, where parties compute m shared secrets $Z_i = g_T^{p^{(i)}}$ $(= 1, ..., m)$. The above three conditions will be utilized in the security proof of the designed protocol. Roughly, the first condition ensures that each user is able to compute the shared secret group elements. The second condition enables the simulator to extract a BDH solution from the challenge test session. The third conditions ensures simulator can verify that shared secret group elements are correctly formed. We refer to the proof of Theorem 1 for further details and provide in the following some examples of admissible polynomials.

Example 1

$$p^{(1)} = u_0v_0w_0,\ p^{(2)} = u_0v_0w_1,\ p^{(3)} = u_0v_1w_0,\ p^{(4)} = u_0v_1w_1,$$
$$p^{(5)} = u_1v_0w_0,\ p^{(6)} = u_1v_0w_1,\ p^{(7)} = u_1v_1w_0,\ p^{(8)} = u_1v_1w_1.$$

Example 2

$$p^{(1)} = u_0v_0w_0 + u_1v_1w_1,\quad p^{(2)} = u_0v_1w_1 + u_1v_0w_0,$$
$$p^{(3)} = u_1v_0w_1 + u_0v_1w_0,\quad p^{(4)} = u_1v_1w_0 + u_0v_0w_1.$$

Example 3. This example essentially explains the construction behind the one-round 3KE protocol by Manulis, Suzuki, and Ustaoglu [31].

$$p^{(1)} = (u_0 + Du_1)(v_0 + v_1)(w_0 + w_1),\ p^{(2)} = (u_0 + u_1)(v_0 + Ev_1)(w_0 + w_1),$$
$$p^{(3)} = (u_0 + u_1)(v_0 + v_1)(w_0 + Fw_1),\ p^{(4)} = (u_0 + Du_1)(v_0 + Ev_1)(w_0 + Fw_1),$$

where $D, E, F \neq 1$.

Example 4

$$\begin{aligned} p^{(1)} &= (u_0 + u_1)(v_0 + v_1)(w_0 + w_1)\ , p^{(2)} = u_0v_1w_1 + u_1v_0w_0, \\ p^{(3)} &= u_1v_0w_1 + u_0v_1w_0 \qquad\qquad\quad\ , p^{(4)} = u_1v_1w_0 + u_0v_0w_1. \end{aligned}$$

3.2 Proposed 3KE Protocol

We now propose the 3KE protocol $\Pi_{p^{(1)},\dots,p^{(m)}}$ constructed from admissible polynomials $p^{(i)}$ $(i = 1, \dots, m)$. We then prove in Theorem 1 that if polynomials $p^{(i)}$ $(i = 1, \dots, m)$ satisfy the conditions of admissible polynomials, the proposed 3KE protocol $\Pi_{p^{(1)},\dots,p^{(m)}}$ is g-eCK secure, i.e., we provide a sufficient condition for building g-eCK secure 3KE protocols.

The proposed 3KE protocol $\Pi_{p^{(1)},\dots,p^{(m)}}$ is described as follows. Let $p^{(i)}$ $(i = 1, \dots, m)$ be admissible polynomials. Let κ be the security parameter. Let $\mathbb{G}$ and $\mathbb{G}_T$ be cyclic groups of prime order q. Let $e : \mathbb{G} \times \mathbb{G} \mapsto \mathbb{G}_T$ be a non-degenerate bilinear map, called pairing, from group $\mathbb{G} \times \mathbb{G}$ to group $\mathbb{G}_T$. Let g and $g_T = e(g, g)$ be a generator of $\mathbb{G}$ and $\mathbb{G}_T$, respectively. Let $H : \{0,1\}^* \to \{0,1\}^\kappa$ be cryptographic hash function modeled as a random oracle. Let P be the protocol identifier of the protocol $\Pi_{p^{(1)},\dots,p^{(m)}}$.

For a user U_A, we set U_A's static and ephemeral keys $A_0 = g^{a_0}$ and $A_1 = g^{a_1}$, respectively, and the lowercase letters are the private keys.

In the description, users U_A, U_B, and U_C communicate with each other, and compute the session key.

1. U_A selects a random ephemeral private key $a_1 \in_U \mathbb{Z}_q$, computes the ephemeral public key $A_1 = g^{a_1}$, stores ephemeral private key a_1 as state information, and broadcasts $(\mathsf{P}, (U_A, U_B, U_C), U_A, A_1)$ to U_B and U_C.
2. U_B selects a random ephemeral private key $b_1 \in_U \mathbb{Z}_q$, computes the ephemeral public key $B_1 = g^{b_1}$, stores ephemeral private key b_1 as state information, and broadcasts $(\mathsf{P}, (U_A, U_B, U_C), U_B, B_1)$ to U_C and U_A.

3. U_C selects a random ephemeral private key $c_1 \in_U \mathbb{Z}_q$, computes the ephemeral public key $C_1 = g^{c_1}$, stores ephemeral private key c_1 as state information, and broadcasts $(\mathsf{P}, (U_A, U_B, U_C), U_C, C_1)$ to U_A and U_B.
4. Upon receiving $(\mathsf{P}, (U_A, U_B, U_C), U_B, B_1)$ and $(\mathsf{P}, (U_A, U_B, U_C), U_C, C_1)$, U_A verifies $B_1, C_1 \in \mathbb{G}$, computes m shared secrets
$$Z_i = \prod_{\beta,\gamma=0,1} e(B_\beta, C_\gamma)^{(d^{(i)}_{0,\beta,\gamma} a_0 + d^{(i)}_{1,\beta,\gamma} a_1)} \quad (i = 1, \ldots, m),$$
obtains the session key $K = H(Z_1, \ldots, Z_m, \mathsf{P}, U_A, A_0, A_1, U_B, B_0, B_1, U_C, C_0, C_1)$, and completes the session.
5. Upon receiving $(\mathsf{P}, (U_A, U_B, U_C), U_C, C_1)$ and $(\mathsf{P}, (U_A, U_B, U_C), U_A, A_1)$, U_B verifies $C_1, A_1 \in \mathbb{G}$, computes m shared secrets
$$Z_i = \prod_{\gamma,\alpha=0,1} e(C_\gamma, A_\alpha)^{(d^{(i)}_{\alpha,0,\gamma} b_0 + d^{(i)}_{\alpha,1,\gamma} b_1)} \quad (i = 1, \ldots, m),$$
obtains the session key $K = H(Z_1, \ldots, Z_m, \mathsf{P}, U_A, A_0, A_1, U_B, B_0, B_1, U_C, C_0, C_1)$, and completes the session.
6. Upon receiving $(\mathsf{P}, (U_A, U_B, U_C), U_A, A_1)$ and $(\mathsf{P}, (U_A, U_B, U_C), U_B, B_1)$, U_C verifies $A_1, B_1 \in \mathbb{G}$, computes m shared secrets
$$Z_i = \prod_{\alpha,\beta=0,1} e(A_\alpha, B_\beta)^{(d^{(i)}_{\alpha,\beta,0} c_0 + d^{(i)}_{\alpha,\beta,1} c_1)} \quad (i = 1, \ldots, m),$$
obtains the session key $K = H(Z_1, \ldots, Z_m, \mathsf{P}, U_A, A_0, A_1, U_B, B_0, B_1, U_C, C_0, C_1)$, and completes the session.

All users U_A, U_B, and U_C compute the same shared secrets
$$Z_i = g_T^{p^{(i)}(a_0, a_1, b_0, b_1, c_0, c_1)} \quad (i = 1, \ldots, m),$$
and so compute the same session key K.

The outlined 3KE protocol $\Pi_{p^{(1)},\ldots,p^{(m)}}$ requires exactly m shared secrets, 4 pairing operations at most, and $4m+1$ exponential operations at most (including the exponentiation for the ephemeral public key).

3.3 Security

For the security of the proposed protocol, we need[4] the gap Bilinear Diffie-Hellman (gap BDH) assumption [5] described below. Let BCDH : $\mathbb{G}^3 \to \mathbb{G}_T$ s.t. BCDH$(g^u, g^v, g^w) = e(g, g)^{uvw}$, and BDDH : $\mathbb{G}^4 \to \{0, 1\}$ be a predicate which takes an input $(g^u, g^v, g^w, e(g, g)^x)$ and returns bit 1 if $uvw = x \bmod q$ and bit 0 otherwise. An adversary $\mathcal{A}$ is given input $U, V, W \in_U \mathbb{G}$ selected uniformly random and oracle access to BDDH$(\cdot, \cdot, \cdot, \cdot)$ oracle, and tries to compute BCDH(U, V, W). For adversary $\mathcal{A}$, we define advantage

[4] Gap BDH assumption is used since in bilinear groups no BDDH oracle is available. Using twin BDH technique we could also rely on BDH instead of gap BDH.

$$Adv^{\mathrm{gapBDH}}(\mathcal{A}) = \Pr[U, V, W \in_R \mathbb{G}, \mathcal{A}^{\mathrm{BDDH}(\cdot,\cdot,\cdot,\cdot)}(U, V, W) = \mathrm{BCDH}(U, V, W)],$$

where the probability is taken over the choices of U, V, W and $\mathcal{A}$'s random tape.

Definition 4 (gap BDH assumption). *We say that $\mathbb{G}$ and $\mathbb{G}_T$ satisfy the gap BDH assumption if, for all polynomial-time adversaries $\mathcal{A}$, advantage $Adv^{\mathrm{gapBDH}}(\mathcal{A})$ is negligible in security parameter κ.*

Theorem 1. *If $\mathbb{G}$ and $\mathbb{G}_T$ are groups where the gap BDH assumption holds, H is a random oracle, and $p^{(i)}$ $(i = 1, \ldots, m)$ are admissible polynomials, the proposed 3KE protocol $\Pi_{p^{(1)},\ldots,p^{(m)}}$ constructed from $p^{(i)}$ $(i = 1, \ldots, m)$ is secure in the g-eCK model.*

Proof (Sketch). From the first condition of admissible polynomials, all users U_A, U_B, and U_C can compute the shared secrets as follows. User U_A, who knows secret keys a_0, a_1, can compute shared secrets

$$Z_i = \prod_{\beta,\gamma=0,1} e(B_\beta, C_\gamma)^{(d^{(i)}_{0,\beta,\gamma}a_0 + d^{(i)}_{1,\beta,\gamma}a_1)} = g_T^{\sum_{\alpha,\beta,\gamma=0,1} d^{(i)}_{\alpha,\beta,\gamma}a_\alpha b_\beta c_\gamma},$$

user U_B, who knows secret keys b_0, b_1, can compute shared secrets

$$Z_i = \prod_{\gamma,\alpha=0,1} e(C_\gamma, A_\alpha)^{(d^{(i)}_{\alpha,0,\gamma}b_0 + d^{(i)}_{\alpha,1,\gamma}b_1)} = g_T^{\sum_{\alpha,\beta,\gamma=0,1} d^{(i)}_{\alpha,\beta,\gamma}a_\alpha b_\beta c_\gamma},$$

and user U_C, who knows secret keys c_0, c_1, can compute shared secrets

$$Z_i = \prod_{\alpha,\beta=0,1} e(A_\alpha, B_\beta)^{(d^{(i)}_{\alpha,\beta,0}c_0 + d^{(i)}_{\alpha,\beta,1}c_1)} = g_T^{\sum_{\alpha,\beta,\gamma=0,1} d^{(i)}_{\alpha,\beta,\gamma}a_\alpha b_\beta c_\gamma}.$$

The gap BDH solver $\mathcal{S}$ extracts the answer g_T^{uvw} of an instance $(U = g^u, V = g^v, W = g^w)$ of the gap BDH problem using adversary $\mathcal{A}$. For instance, we assume the case that test session $\mathtt{sid}^*$, owner of which is user U_A, has no partnered sessions $\overline{\mathtt{sid}^*}$, owners of which are users U_B and U_C, adversary $\mathcal{A}$ is given a_0, and adversary $\mathcal{A}$ does not obtain a_1, b_0 and c_0 from the condition of the freshness. In this case, solver $\mathcal{S}$ can perfectly simulate $\mathsf{StaticKeyReveal}$ query by selecting random a_0 and setting $A_0 = g^{a_0}$, and solver $\mathcal{S}$ embeds the instance as $A_1 = U$ $(= g^u)$, $B_0 = V$ $(= g^v)$ and $C_0 = W$ $(= g^w)$ to extract g_T^{uvw} from the shared secrets $Z_i = g_T^{p^{(i)}}$ $(i = 1, \ldots, m)$.

From the second condition of admissible polynomials, solver $\mathcal{S}$ can extract the answer of the gap BDH instance as follows. From the second condition, there exist i, j, k, l $(1 \le i, j, k, l \le m)$, s.t., $V^{(i)}_{1,*,*}$, $V^{(j)}_{1,*,*}$, $V^{(k)}_{1,*,*}$, and $V^{(l)}_{1,*,*}$ are linearly independent. Using knowledge of a_0, solver $\mathcal{S}$ can compute

$$Z'_I = g_T^{a_1(d^{(I)}_{1,0,0}b_0c_0 + d^{(I)}_{1,0,1}b_0c_1 + d^{(I)}_{1,1,0}b_1c_0 + d^{(I)}_{1,1,1}b_1c_1)}$$

$$= Z_I/(e(B_0, C_0)^{d^{(I)}_{0,0,0}a_0} e(B_0, C_1)^{d^{(I)}_{0,0,1}a_0} e(B_1, C_0)^{d^{(I)}_{0,1,0}a_0} e(B_1, C_1)^{d^{(I)}_{0,1,1}a_0})$$

for the indices $I = i, j, k, l$. Solver $\mathcal{S}$ can compute $g_T^{a_1 b_0 c_0}$ from Z'_i, Z'_j, Z'_k, Z'_l since $V^{(i)}_{1,*,*}$, $V^{(j)}_{1,*,*}$, $V^{(k)}_{1,*,*}$, and $V^{(l)}_{1,*,*}$ are linearly independent, and successfully outputs the answer $g_T^{a_1 b_0 c_0} = g_T^{uvw}$ of the gap BDH problem.

From the third condition of admissible polynomials, solver $\mathcal{S}$ can check whether the shared secrets are correctly formed w.r.t. static and ephemeral public keys, and can simulate H and $\mathsf{SessionKeyReveal}$ queries consistently. More precisely, in the simulation of the $H(Z_1, \dots, Z_m, \mathsf{P}, U_A, A_0, A_1, U_B, B_0, B_1, U_C, C_0, C_1)$ query, solver $\mathcal{S}$ must check that the shared secrets Z_i $(i = 1, \dots, m)$ are correctly formed, and if so return session key K that is consistent with the previously answered $\mathsf{SessionKeyReveal}(\mathsf{P}, U_X, U_A, A_0, A_1, U_B, B_0, B_1, U_C, C_0, C_1)$ $(X = A, B, C)$ queries. For all i $(= 1, \dots, m)$, solver $\mathcal{S}$ performs the following procedure. Using the knowledge of a_0, solver $\mathcal{S}$ can compute

$$Z'_i = g_T^{a_1 (d^{(i)}_{1,0,0} b_0 c_0 + d^{(i)}_{1,0,1} b_0 c_1 + d^{(i)}_{1,1,0} b_1 c_0 + d^{(i)}_{1,1,1} b_1 c_1)}$$

$$= Z_i / (e(B_0, C_0)^{d^{(i)}_{0,0,0} a_0} e(B_0, C_1)^{d^{(i)}_{0,0,1} a_0} e(B_1, C_0)^{d^{(i)}_{0,1,0} a_0} e(B_1, C_1)^{d^{(i)}_{0,1,1} a_0})$$

Then, solver $\mathcal{S}$ can check if shared secret Z'_i is correctly formed w.r.t. the static and ephemeral public keys, by asking BDDH oracle

$$\mathrm{BDDH}(g^{\ell^{(i)}_{1,*,*}(a_0,a_1)}, g^{\ell'^{(i)}_{1,*,*}(b_0,b_1)}, g^{\ell''^{(i)}_{1,*,*}(c_0,c_1)}, Z'_i) = 1,$$

since the third condition of admissible polynomials holds, and this implies Z_i is correctly formed. Here solver $\mathcal{S}$ can compute $g^{\ell^{(i)}_{1,*,*}(a_0,a_1)}$, $g^{\ell'^{(i)}_{1,*,*}(b_0,b_1)}$, and $g^{\ell''^{(i)}_{1,*,*}(c_0,c_1)}$, since $\ell^{(i)}_{1,*,*}(a_0, a_1)$, $\ell'^{(i)}_{1,*,*}(b_0, b_1)$, and $\ell''^{(i)}_{1,*,*}(c_0, c_1)$ are linear.

$\square$

4 Conclusion

We presented a sufficient condition for constructing one-round ephemeral key-leakage resilient 3KE protocols where parties are equipped with a static public key and an ephemeral public key, each comprised of only one group element, and where key derivation is performed via a single call to the hash function, modeled as a random oracle. Technically, the proposed 3KE protocol can be seen as a combination of several two-dimensional versions of the original (unauthenticated) tripartite key exchange protocol from [20]. The protocol gives rise to a framework for the design of efficient ephemeral key-leakage resilient one-round 3KE protocols in the model from [31] by choosing different admissible polynomials. The amount of work for proving security of all those protocols essentially reduces to proving that chosen polynomials are admissible according to the conditions stated in this paper.

References

1. Abdalla, M., Chevalier, C., Manulis, M., Pointcheval, D.: Flexible Group Key Exchange with On-demand Computation of Subgroup Keys. In: Bernstein, D.J., Lange, T. (eds.) AFRICACRYPT 2010. LNCS, vol. 6055, pp. 351–368. Springer, Heidelberg (2010)

2. Abdalla, M., Fouque, P.-A., Pointcheval, D.: Password-Based Authenticated Key Exchange in the Three-Party Setting. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 65–84. Springer, Heidelberg (2005)
3. Abdalla, M., Pointcheval, D.: A Scalable Password-Based Group Key Exchange Protocol in the Standard Model. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 332–347. Springer, Heidelberg (2006)
4. Al-Riyami, S.S., Paterson, K.G.: Tripartite Authenticated Key Agreement Protocols from Pairings. In: Paterson, K.G. (ed.) Cryptography and Coding 2003. LNCS, vol. 2898, pp. 332–359. Springer, Heidelberg (2003)
5. Baek, J., Safavi-Naini, R., Susilo, W.: Efficient Multi-receiver Identity-Based Encryption and Its Application to Broadcast Encryption. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 380–397. Springer, Heidelberg (2005)
6. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated Key Exchange Secure against Dictionary Attacks. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 139–155. Springer, Heidelberg (2000)
7. Bellare, M., Rogaway, P.: Entity Authentication and Key Distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994)
8. Bellovin, S.M., Merritt, M.: Augmented Encrypted Key Exchange: A Password-Based Protocol Secure against Dictionary Attacks and Password File Compromise. In: ACM CCS 1993, pp. 244–250. ACM (1993)
9. Boyko, V., MacKenzie, P.D., Patel, S.: Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 156–171. Springer, Heidelberg (2000)
10. Bresson, E., Chevassut, O., Pointcheval, D., Quisquater, J.-J.: Provably Authenticated Group Diffie-Hellman Key Exchange. In: ACM CCS 2001, pp. 255–264. ACM Press (2001)
11. Bresson, E., Manulis, M.: Contributory Group Key Exchange in the Presence of Malicious Participants. IET Information Security 2(3), 85–93 (2008)
12. Bresson, E., Manulis, M.: Securing Group Key Exchange against Strong Corruptions. In: ACM ASIACCS 2008, pp. 249–260. ACM Press (2008); full version in Intl. J. Applied Cryptography in 2008
13. Bresson, E., Manulis, M., Schwenk, J.: On Security Models and Compilers for Group Key Exchange Protocols. In: Miyaji, A., Kikuchi, H., Rannenberg, K. (eds.) IWSEC 2007. LNCS, vol. 4752, pp. 292–307. Springer, Heidelberg (2007)
14. Canetti, R., Krawczyk, H.: Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001)
15. Choo, K.-K.R.: Secure Key Establishment. Advances in Information Security, vol. 41. Springer (2009)
16. Cremers, C.J.F.: Examining Indistinguishability-Based Security Models for Key Exchange Protocols: The case of CK, CK-HMQV, and eCK. In: ASIACCS 2011, pp. 80–91. ACM, New York (2011)
17. Diffie, W., Hellman, M.E.: New Directions in Cryptography. IEEE Transactions on Information Theory IT-22(6), 644–654 (1976)
18. Fujioka, A., Suzuki, K.: Designing Efficient Authenticated Key Exchange Resilient to Leakage of Ephemeral Secret Keys. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 121–141. Springer, Heidelberg (2011)
19. Gorantla, M.C., Boyd, C., González-Nieto, J.M., Manulis, M.: Modeling key compromise impersonation attacks on group key exchange protocols. ACM Trans. Inf. Syst. Secur. 14(4), 28 (2011)

20. Joux, A.: A one round protocol for tripartite Diffie–Hellman. Journal of Cryptology 17(4), 263–276 (2004)
21. Katz, J., Shin, J.S.: Modeling Insider Attacks on Group Key-Exchange Protocols. In: ACM CCS 2005, pp. 180–189. ACM Press (2005)
22. Katz, J., Yung, M.: Scalable Protocols for Authenticated Group Key Exchange. J. Cryptology 20(1), 85–113 (2007)
23. Kim, M., Fujioka, A., Ustaoğlu, B.: Strongly Secure Authenticated Key Exchange without NAXOS' Approach. In: Takagi, T., Mambo, M. (eds.) IWSEC 2009. LNCS, vol. 5824, pp. 174–191. Springer, Heidelberg (2009)
24. Krawczyk, H.: HMQV: A High-Performance Secure Diffie-Hellman Protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546–566. Springer, Heidelberg (2005)
25. LaMacchia, B.A., Lauter, K., Mityagin, A.: Stronger Security of Authenticated Key Exchange. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 1–16. Springer, Heidelberg (2007)
26. Lim, M.-H., Lee, S., Lee, H.: Cryptanalysis on improved one-round Lin-Li's tripartite key agreement protocol. Cryptology ePrint Archive, Report 2007/411
27. Lim, M.-H., Lee, S., Park, Y., Lee, H.: An Enhanced One-Round Pairing-Based Tripartite Authenticated Key Agreement Protocol. In: Gervasi, O., Gavrilova, M.L. (eds.) ICCSA 2007, Part II. LNCS, vol. 4706, pp. 503–513. Springer, Heidelberg (2007)
28. Lin, C.-H., Lin, H.-H.: Secure one-round tripartite authenticated key agreement protocol from Weil pairing. In: AINA 2005, vol. 2, pp. 135–138. IEEE (2005)
29. Manulis, M.: Security-Focused Survey on Group Key Exchange Protocols. Cryptology ePrint Archive, Report 2006/395 (2006), `http://eprint.iacr.org/2006/395`
30. Manulis, M.: Group Key Exchange Enabling On-Demand Derivation of Peer-to-Peer Keys. In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536, pp. 1–19. Springer, Heidelberg (2009)
31. Manulis, M., Suzuki, K., Ustaoglu, B.: Modeling Leakage of Ephemeral Secrets in Tripartite/Group Key Exchange. In: Lee, D., Hong, S. (eds.) ICISC 2009. LNCS, vol. 5984, pp. 16–33. Springer, Heidelberg (2010)
32. Menezes, A., Ustaoglu, B.: Comparing the Pre- and Post-specified Peer Models for Key Agreement. In: Mu, Y., Susilo, W., Seberry, J. (eds.) ACISP 2008. LNCS, vol. 5107, pp. 53–68. Springer, Heidelberg (2008)
33. Moriyama, D., Okamoto, T.: An eCK-Secure Authenticated Key Exchange Protocol without Random Oracles. In: Pieprzyk, J., Zhang, F. (eds.) ProvSec 2009. LNCS, vol. 5848, pp. 154–167. Springer, Heidelberg (2009)
34. Okamoto, T.: Authenticated Key Exchange and Key Encapsulation in the Standard Model. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 474–484. Springer, Heidelberg (2007)
35. Okamoto, T., Pointcheval, D.: The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes. In: Kim, K.-C. (ed.) PKC 2001. LNCS, vol. 1992, pp. 104–118. Springer, Heidelberg (2001)
36. Shim, K.: Efficient one round tripartite authenticated key agreement protocol from Weil pairing. IET Electronics Letters 39(2), 208–209 (2003)
37. Ustaoglu, B.: Comparing *SessionStateReveal* and *EphemeralKeyReveal* for Diffie-Hellman Protocols. In: Pieprzyk, J., Zhang, F. (eds.) ProvSec 2009. LNCS, vol. 5848, pp. 183–197. Springer, Heidelberg (2009)
38. Zhao, J., Gu, D., Gorantla, M.C.: Stronger security model of group key agreement. In: ASIACCS 2011, pp. 435–440. ACM (2011)

A Game-Theoretic Perspective on Oblivious Transfer*

Haruna Higo[1], Keisuke Tanaka[1], Akihiro Yamada[1], and Kenji Yasunaga[2]

[1] Tokyo Institute of Technology, Japan
{higo9,keisuke,yamada9}@is.titech.ac.jp
[2] Institute of Systems, Information Technologies and Nanotechnologies, Japan
yasunaga@isit.or.jp

Abstract. Asharov, Canetti, and Hazay (Eurocrypt 2011) studied how game-theoretic concepts can be used to capture the cryptographic properties of correctness, privacy, and fairness in two-party protocols in the presence of fail-stop adversaries. Based on their work, we characterize the properties of "two-message" oblivious transfer protocols by using a game-theoretic concept. Specifically, we present a single two-player game for two-message oblivious transfer in the game-theoretic framework, where it captures the cryptographic properties of correctness and privacy in the presence of *malicious* adversaries.

Keywords: cryptography, game theory, oblivious transfer.

1 Introduction

1.1 Background

Cryptographic protocols are designed for parties who follow them to guarantee some properties such as correctness and privacy. In many cases, such properties are discussed in a way that if some player honestly follow the description of the protocol, she can achieve some desirable properties even if some of other participants of the protocol are controlled by an adversary. Game theory studies the behavior of "rational" parties interacting with each other. One of the interplay between cryptography and game theory is to design cryptographic protocols in the presence of rational parties, who are neither honest nor malicious. A line of work on *rational secret sharing* [10,14,1,7,12,13,16,15,3,6] is in this direction.

Recently, Asharov, Canetti, and Hazay [2] studied how game-theoretic concepts can be used to capture the cryptographic properties such as correctness, privacy, and fairness. In particular, they characterized these properties by using a game-theoretic concept, Nash equilibrium, in the setting of secure two-party protocols in the presence of fail-stop adversaries. Cryptographic properties of two-party protocols are characterized in the following way. A protocol satisfies a

* This research was supported in part by a grant of I-System Co. Ltd., NTT Information Sharing Platform Laboratories, and Ministry of Education, Culture, Sports, Science and Technology.

"certain" cryptographic property if and only if the strategy of honestly following the description of the protocol is a Nash equilibrium in a "certain" game defined with "certain" utility functions. Regarding the cryptographic properties of correctness and privacy, they showed games together with utility functions that are equivalent to these properties. Regarding fairness, they introduced a new cryptographic fairness that has an equivalent game-theoretic characterization.

1.2 This Work

Based on the work of Asharov et al. [2], we further explore how the cryptographic properties can be captured by game-theoretic concepts. In particular, we characterize the properties of *oblivious transfer* (OT), which is one of the well-studied two-party protocols, in terms of game-theoretic concepts. OT is a protocol between a sender and a receiver. The sender has two secrets x_0 and x_1, and the receiver has a choice bit $c \in \{0, 1\}$. After running the protocol, the receiver obtains x_c, while the sender obtains nothing. Privacy is considered both for the sender and the receiver. The sender's privacy requires that the receiver learns nothing about x_{1-c}. The receiver's privacy requires that the sender learns nothing about the choice bit c. In this work, we present a game for *two-message* OT together with the utility functions of the sender and the receiver such that a protocol satisfies the cryptographic properties of correctness and privacy in the presence of *malicious* adversaries *if and only if* the strategy of honestly following the description of the protocol is a Nash equilibrium in the game.

Our characterization of two-message OT has several advantages compared to the work of [2].

First, the game defined in our work is played between two rational players, while every game defined in [2] is played by a single rational player. For example, in [2], the privacy of a protocol is characterized by two games, one for the privacy of the player 1 and the other the player 2. In each game, the utility function of only one player is essentially considered. Since game-theoretic concepts are of significant meaning in the presence of multiple rational players, it is preferable to characterize a single game which is essentially played between two rational players.

Second, we characterize both correctness and privacy by a single game, while each property is characterized by different games in [2].

Third, we consider the setting in the presence of malicious adversaries, who can take any malicious action in the protocol and are stronger than fail-stop adversaries, who are allowed to take only two actions, "continue" and "stop", in each round.

The reason for focusing on "two-message" OT is that there exists an definition of privacy based on indistinguishability for two-message OT in the presence of malicious adversaries [11,8]. Although the ideal/real simulation paradigm provides strong and desirable security in some cryptographic contexts, the definition based on indistinguishability fits for a game-theoretic framework. In the indistinguishability-based privacy, a player is asked to predict which of the two values is used as the input of the other player in the protocol. Thus, the utility of the player can be explicitly defined in a way such that she obtains higher utility if the prediction is correct, and lower utility otherwise.

We present a single game between two players that characterize the cryptographic properties of two-message OT. This characterization is the first step toward understanding how OT protocols work for rational players. It is an interesting challenge to characterize other variations of games with other utility functions and other solution concepts than Nash equilibrium.

2 Models and Definitions

We review some basic definitions as well as the solution concepts from game theory.

A function $\mu : \mathbb{N} \to \mathbb{R}$ is called *negligible* if for any polynomial p, there exists a value $N \in \mathbb{N}$ such that for all $n > N$ it holds that $\mu(n) < 1/p(n)$. We describe a negligible function on n as $negl(n)$. Let $X = \{X(n,a)\}_{n\in\mathbb{N},a\in\{0,1\}^*}$ and $Y = \{Y(n,a)\}_{n\in\mathbb{N},a\in\{0,1\}^*}$ be distribution ensembles. Then, we say that X and Y are *computationally indistinguishable*, denoted by $X \overset{c}{\equiv} Y$, if for every probabilistic polynomial-time distinguisher D, it holds that

$$|\Pr[D(X(n,a)) = 1] - \Pr[D(Y(n,a)) = 1]| \leq negl(n).$$

All players are assumed to run in time which is polynomial on the security parameter n. Formally, each player has a security parameter tape that the value 1^n is written as a part of the input.

2.1 Cryptographic Security

We now turn to define OT and its cryptographic security. We focus only on two-message OT. The receiver sends a message to the sender, then the sender sends a message to the receiver. After receiving the message, the receiver obtains the resulting output. Namely, an OT protocol consists of two probabilistic polynomial-time algorithms S^π and R^π. When a protocol $\pi = (S^\pi, R^\pi)$ is executed on the input pair $((x_0, x_1), c)$ where $x_0, x_1 \in \{0,1\}^*$, $|x_0| = |x_1|$ and $c \in \{0,1\}$, the two algorithms run as follows. First, R^π runs on input c and outputs a message $m_R \in \{0,1\}^*$, then S^π runs on $((x_0, x_1), m_R)$ and it outputs a message $m_S \in \{0,1\}^*$. After that, R^π computes the resulting output. As we define the privacy of OT in malicious model, one of the players can be malicious. A malicious adversary use any polynomial-time algorithm. Note that we restrict a malicious receiver to use a *deterministic* polynomial-time algorithm, while a malicious sender can use a probabilistic polynomial-time algorithm. As we will mention later, this restriction is necessary when we define the privacy based on indistinguishability, which is used widely [11].

Definition 1 (View). *Let $\pi = (S^\pi, R^\pi)$ be an OT protocol. The* view *of the sender during the execution of π on input pair (x_S, x_R), when the sender and the receiver use S and R as their algorithms respectively, is denoted by*

$\mathsf{view}_{\pi,S}(S(x_S), R(x_R))$ *and equals* (x_S, r_S, m_R)*, where* r_S *is its random tape, and* m_R *represents the message which the sender received from the receiver. The view of the receiver is defined as* $\mathsf{view}_{\pi,R}(S(x_S), R(x_R)) = (x_R, r_R, m_S)$*, where* r_R *is its random/advice tape, and* m_S *represents the message which the receiver received from the sender.*

Definition 2 (Output). *Let* $\pi = (S^\pi, R^\pi)$ *be an OT protocol. The* output *of the receiver after the execution of* π *on input pair* (x_S, x_R) *when the sender and the receiver use* S *and* R *as their algorithms is denoted by* $\mathsf{output}_{\pi,R}(S(x_S), R(x_R))$*. The output of the receiver is equal to* $\perp$ *if and only if the receiver does not receive any message from the sender.*

Next, we define the cryptographic security of OT. Our definition of secure OT is similar to the ones considered in previous works [8,4].

Definition 3 (Cryptographic security). *An OT protocol* $\pi = (S^\pi, R^\pi)$ *is said to be* cryptographically secure *if all of the following properties are satisfied.*

Sender's privacy: π *is said to be* cryptographically private for the sender *if the following two properties holds.*
- *It holds that*

$$\begin{aligned}&\{\mathsf{view}_{\pi,R}(S^\pi(X^0), R^\pi(c))\}_{x_0,x_1,x\in\{0,1\}^*,|x_0|=|x_1|=|x|,c\in\{0,1\}}\\&\quad\overset{c}{\equiv}\{\mathsf{view}_{\pi,R}(S^\pi(X^1), R^\pi(c))\}_{x_0,x_1,x\in\{0,1\}^*,|x_0|=|x_1|=|x|,c\in\{0,1\}},\end{aligned} \tag{1}$$

where $X^0 = (x_0, x_1)$*, and* $X^1 = (x_0, x)$ *if* $c = 0$ *and* $X^1 = (x, x_1)$ *if* $c = 1$*.*
- *There exists a function Choice such that for every deterministic polynomial-time algorithm* R^* *and advice tape* $z \in \{0,1\}^*$*, it holds that*

$$\begin{aligned}&\{\mathsf{view}_{\pi,R}(S^\pi(X^0), R^*(c,z))\}_{x_0,x_1,x\in\{0,1\}^*,|x_0|=|x_1|=|x|,c\in\{0,1\}}\\&\quad\overset{c}{\equiv}\{\mathsf{view}_{\pi,R}(S^\pi(X^1), R^*(c,z))\}_{x_0,x_1,x\in\{0,1\}^*,|x_0|=|x_1|=|x|,c\in\{0,1\}},\end{aligned} \tag{2}$$

where $X^0 = (x_0, x_1)$*, and* $X^1 = (x_0, x)$ *if* $Choice(R^*, c, z) = 0$ *and* $X^1 = (x, x_1)$ *if* $Choice(R^*, c, z) = 1$*.*

Receiver's privacy: π *is said to be* cryptographically private for the receiver *if for every probabilistic polynomial-time algorithm* S^**, it holds that*

$$\begin{aligned}&\{\mathsf{view}_{\pi,S}(S^*(x_0,x_1), R^\pi(0))\}_{x_0,x_1,z\in\{0,1\}^*,|x_0|=|x_1|}\\&\qquad\qquad\overset{c}{\equiv}\{\mathsf{view}_{\pi,S}(S^*(x_0,x_1), R^\pi(1))\}_{x_0,x_1,z\in\{0,1\}^*,|x_0|=|x_1|}.\end{aligned}$$

Correctness: π *is said to be* cryptographically correct *if for every sender's two input strings* $x_0, x_1 \in \{0,1\}^*$ *such that* $|x_0| = |x_1|$*, it holds that*

$$\Pr[\mathsf{output}_{\pi,R}(S^\pi(x_0,x_1), R^\pi(c)) = x_c] \geq 1 - negl(n).$$

Since we focus on two-message OT and the malicious receiver's algorithm is deterministic, the message which the receiver sends to the sender is fully determined by its input. This means that the receiver's algorithm and its input fully determines which secret he should receive after interacting with the sender. The function *Choice* outputs the index of the received secret. As mentioned in [11], restricting the receiver's algorithm to be deterministic does not weaken its maliciousness, but allows us to define the sender's privacy based on indistinguishability.

2.2 Game-Theoretic Concepts

To capture the security of OT in the field of game theory, we define the concepts of games, utility functions, and a solution concept called Nash equilibrium. Our definitions are similar to the ones in previous works [5,9].

First, we define non-cooperative two-player games with incomplete information. Since the players of OT do not know any information about the other player's input, and they independently decide how to behave, the implementation of OT can be defined in terms of non-cooperative two-player games with incomplete information. Formally, we define such games as follows.

Definition 4 (Non-cooperative two-player games with incomplete information). *A non-cooperative two-player game with incomplete information is described as $\Gamma = (N, \{A_i, T_i, \mathcal{U}_i\}_{i \in N}, \mathcal{D})$, where*

- *$N = \{0, 1\}$ is a set of players.*
- *A_i is a set of actions for player $i \in N$. Let $A = A_0 \times A_1$.*
- *T_i is a set of types for player $i \in N$. Let $T = T_0 \times T_1$.*
- *$\mathcal{U}_i : A \times T \to \mathbb{R}$ is the utility function for player $i \in N$.*
- *$\mathcal{D}$ is the probability distribution over T. The tuples of the types for each player $(t_0, t_1) \in T$ happens with probability $p_{\mathcal{D}}(t_0, t_1)$ which is defined by $\mathcal{D}$. Each player $i \in N$ with the type t_i believes that the type t_{1-i} of the other player occur with probability $p_{\mathcal{D}}(t_{1-i}|t_i)$.*

A function $\sigma_i : T_i \to A_i$ is called a strategy for player i. Each player's action is determined by its type and its strategy.

If the player i knows the types of the both players, it can calculate its own utility with respect to any pair of their strategies. For example, when the pair of their types is (t_0, t_1) and each player has strategy σ_0 and σ_1 respectively, the utility of the player i after the completion of the game is $\mathcal{U}_i(\sigma_0(t_0), \sigma_1(t_1), t_0, t_1)$. We write it as $\mathcal{U}_i(\sigma_0(t_0), \sigma_1(t_1))$ for simplicity.

The values of utility functions are used as the "indicator" when the players select their strategies. Each players rationally select their strategies based on their utility functions, that is, they select the strategies with which they can get the highest utility.

However, in games with incomplete information, not knowing the other's type even after the execution of the protocol, each player cannot calculate its own utility. Thus, they use the expectation value of the utility to decide what strategy they take. Formally, we use the following concept of the expected utility.

Definition 5 (Expected utility for non-cooperative two-player games with incomplete information). *Let $\Gamma = (N, \{A_i, T_i, \mathcal{U}_i\}_{i \in N}, \mathcal{D})$ be a non-cooperative two-player game with incomplete information, and let $N = \{0, 1\}$. The* expected utility *of the player* 0 *with type t_0 for a pair of their strategies (σ_0, σ_1) on the game Γ is*

$$U_0(\sigma_0, \sigma_1) = \sum_{t_1 \in T_1} \mathcal{U}_0(\sigma_0(t_0), \sigma_1(t_1))\, p_{\mathcal{D}}(t_1|t_0).$$

The expected utility of the player 1 *is defined analogously.*

The players of incomplete information games seek to maximize their expected utility instead of the exact value of utility with respect to a certain strategy and a certain type. With such games, we use the following definition of Nash equilibrium as a solution concept. To compensate for the small inevitable imperfections of cryptographic constructs, we take no account of negligible differences of values.

Definition 6 (Computational Nash equilibrium for non-cooperative two-player games with incomplete information). *Let $\Gamma = (N, \{A_i, T_i, \mathcal{U}_i\}_{i \in N}, \mathcal{D})$ be a non-cooperative two-player game with incomplete information, and let $N = \{0, 1\}$. A pair of probabilistic polynomial-time strategies (σ_0, σ_1) is a* computational Nash equilibrium *if for every player $i \in N$ and every strategy σ'_i it can take, it holds that*

$$U_i(\sigma_0, \sigma_1) \geq U_i(\sigma''_0, \sigma''_1) - negl(n)$$

where $\sigma''_i = \sigma'_i$ and $\sigma''_{1-i} = \sigma_{1-i}$.

3 Game-Theoretic Perspective

3.1 Game-Theoretic Security

We formally define a game to capture the security of OT. Our game can be roughly divided into two phases. In the first phase, the sender and the receiver execute an OT protocol, and in the second one, they are given two values and guess the other's input from them.

Definition 7 (A game to capture the security). *Let $\pi = (S^\pi, R^\pi)$ be an OT protocol. On input $((S, G_S), (R, G_R), Choice, (x_0, x_1, x, c, z))$ where*

- *S is a probabilistic polynomial-time algorithm of the sender to execute the protocol π,*
- *R is R^π or a deterministic polynomial-time algorithm of the receiver to execute the protocol π,*
- *G_S and G_R are probabilistic polynomial-time guessing algorithms which outputs a binary value,*
- *Choice is a function which outputs* 0 *or* 1*,*
- *$x_0, x_1, x, z \in \{0,1\}^*$, $|x_0| = |x_1| = |x|$, $c \in \{0, 1\}$,*

the game Game^{π} *runs as follows.*

1. *Let* $X^0 = (x_0, x_1)$*. If* $Choice(R, c, z) = 0$ *then let* $X^1 = (x_0, x)$*, and if* $Choice(R, c, z) = 1$ *then let* $X^1 = (x, x_1)$*. If* $R = R^{\pi}$ *then set* z *be an empty string.*
2. *Choose a bit* b *from* $\{0, 1\}$ *uniformly at random.*
3. *Execute* π *with algorithms* S *and* R *on the pair of input* $(X^b, (c, z))$*. The receiver outputs* $\mathsf{output}_{\pi,R}(S(X^b), R(c, z))$*. Let* $\mathsf{fin}_{\pi}(S(X^b), R(c, z)) = 1$ *if the protocol finishes to the end or in the middle, without stopped by the players during the protocol execution.*
4. *Compute* $G_S(\mathsf{history}_{\pi,S}(S(X^b), R(c, z)))$ *and* $G_R(\mathsf{history}_{\pi,R}(S(X^b), R(c, z)), X^0, X^1)$*. Here we describe the local histories of each player by* $\mathsf{history}_{\pi,S}$ *and* $\mathsf{history}_{\pi,R}$*. They consist of their own inputs, their random/advice tapes, and the message received from the other player.*
5. *Let* $\mathsf{guess}_S = 1$ *if* $G_S(\mathsf{history}_{\pi,S}(S(X^b), R(c, z))) = c$ *holds and* $\mathsf{guess}_S = 0$ *otherwise,* $\mathsf{guess}_R = 1$ *if* $G_R(\mathsf{history}_{\pi,R}(S(X^b), R(c, z)), X^0, X^1) = b$ *holds and* $\mathsf{guess}_R = 0$ *otherwise , and* $\mathsf{correct} = 1$ *if* $(\mathsf{fin}_{\pi}(S(X^b), R(c, z)) = 0$ *or* $\mathsf{output}_{\pi,R}(S(X^b), R(c, z)) = x_c$ *holds and* $\mathsf{correct} = 1$ *otherwise.*
6. *Output* $(\mathsf{guess}_S, \mathsf{guess}_R, \mathsf{correct})$*.*

As well as the cryptographic security, we consider the three properties such that sender's privacy, receiver's privacy and correctness in a game-theoretic manner. More specifically, we define the utility function of each player with three terms representing the properties. The functions describe the motivation of each player to execute an OT protocol which protects his/her privacy, delivers the proper secret and may tell him/her some information about the other's input.

Definition 8 (A pair of expected utility functions for the security). *Let* $\pi = (S^{\pi}, R^{\pi})$ *be an OT protocol, and* α_S*,* β_S*,* γ_S*,* α_R*,* β_R *and* γ_R *be positive constants. The pair of utility functions for security is defined by* $\mathcal{U} = (\mathcal{U}_S, \mathcal{U}_R)$*. For a pair of strategies and guessing algorithms* $((S, G_S), (R, G_R))$*, four strings* x_0, x_1, x, z *such that* $|x_0| = |x_1| = |x|$*, a bit* c *and a function Choice, let* $X^0 = (x_0, x_1)$*,* $X^1 = (x_0, x)$ *if* $Choice(R, c, z) = 0$ *and* $X^1 = (x, x_1)$ *if* $Choice(R, c, z) = 1$*. The functions on* $((S, G_S), (R, G_R), Choice, (x_0, x_1, x, c, z))$ *are defined as follows.*

$$\begin{aligned}
&\mathcal{U}_S((S, G_S), (R, G_R), Choice, (x_0, x_1, x, c, z)) \\
&\qquad = -\alpha_S\,(\mathsf{guess}_R - 1/2) + \beta_S\,(\mathsf{correct} - 1) + \gamma_S\,(\mathsf{guess}_S - 1/2) \\
&\mathcal{U}_R((S, G_S), (R, G_R), Choice, (x_0, x_1, x, c, z)) \\
&\qquad = -\alpha_R\,(\mathsf{guess}_S - 1/2) + \beta_R\,(\mathsf{correct} - 1) + \gamma_R\,(\mathsf{guess}_R - 1/2)
\end{aligned}$$

Here, guess_S*,* guess_R *and* $\mathsf{correct}$ *is the outcome of the* Game^{π} *on input* $((S, G_S), (R, G_R), Choice, (x_0, x_1, x, c, z))$*. We omit to write the input of the function as* $((S, G_S), (R, G_R))$ *if the function Choice and the input tuple* (x_0, x_1, x, c, z) *are obvious from the context.*

Utility functions capture the preference of the players. Since each of our function has three parameters, changing the parameters will make it represents a player with another incentive. For example, if we set γ_S larger than α_S and β_S, it captures the sender who have a strong desire to learn the receiver's choice bit.

Definition 9 (Computational Nash equilibrium for the security). *For a pair of utility functions $\mathcal{U} = (\mathcal{U}_S, \mathcal{U}_R)$, we say a pair of the strategies (S, R) is a* Nash equilibrium *for the game* Game^π*, if there exist a function Choice such that for every pair of probabilistic polynomial-time guessing algorithms (G_S, G_R), pair of polynomial-time strategies (S', R') and tuple (x_0, x_1, x, c, z) where $x_0, x_1, x, z \in \{0,1\}^*$, $|x_0| = |x_1| = |x|$, $c \in \{0,1\}$, it holds that*

$$\mathcal{U}_S((S, G_S), (R, G_R)) \geq \mathcal{U}_S((S', G_S), (R, G_R)) - negl(n),$$

and

$$\mathcal{U}_R((S, G_S), (R, G_R)) \geq \mathcal{U}_R((S, G_S), (R', G_R)) - negl(n).$$

Using the game Game^π and the pair of utility functions $\mathcal{U} = (\mathcal{U}_S, \mathcal{U}_R)$, we define game-theoretic security of OT. Intuitively, if both players can get the highest utility when both of them act "honestly" then the protocol is said to be game-theoretically secure. The definition of game-theoretic security is as follows.

Definition 10 (Game-theoretic security of oblivious transfer). *An OT protocol $\pi = (S^\pi, R^\pi)$ is said to be* game-theoretically secure *if the pair of strategies (S^π, R^π) is a Nash equilibrium for the game* Game^π *with the pair of utility functions $\mathcal{U} = (\mathcal{U}_S, \mathcal{U}_R)$.*

3.2 Equivalence of the Two Security Definitions

We show the equivalence between the two security notions. That is, cryptographic security introduced in Section 2.1 and our game-theoretic security defined in the Section 3.1 are equal.

Theorem 1. *An OT protocol π is cryptographically secure if and only if π is game-theoretically secure.*

We prove this theorem in the next two sections.

3.3 Cryptographic Security Implies Game-Theoretic Security

In this section, we prove the "only if" part of Theorem 1, that is, we prove the following lemma.

Lemma 1. *An OT protocol π is game-theoretically secure if π is cryptographically secure.*

Proof. To prove the lemma, we assume that a protocol is not game-theoretically secure and show that nor is it cryptographically secure. When a protocol π is not game-theoretically secure, at least one of the next two cases holds.

Case1: For every function *Choice*, there exist probabilistic polynomial-time algorithms S^*, G_S^* and G_R^*, a tuple (x_0, x_1, x, c, z) where $x_0, x_1, x, z \in \{0,1\}^*$, $|x_0| = |x_1| = |x|$, $c \in \{0,1\}$, and a non-negligible function ϵ_S such that

$$\begin{aligned}&\mathcal{U}_S((S^*, G_S^*), (R^\pi, G_R^*), Choice, (x_0, x_1, x, c, z))\\ &\qquad > \mathcal{U}_S((S^\pi, G_S^*), (R^\pi, G_R^*), Choice, (x_0, x_1, x, c, z)) + \epsilon_S(n).\end{aligned}$$

Case2: For every function *Choice*, there exist a deterministic polynomial-time algorithm R^*, probabilistic polynomial-time algorithms G_S^* and G_R^*, a tuple (x_0, x_1, x, c, z) where $x_0, x_1, x, z \in \{0,1\}^*$, $|x_0| = |x_1| = |x|$, $c \in \{0,1\}$, and a non-negligible function ϵ_R such that

$$\begin{aligned}&\mathcal{U}_R((S^\pi, G_S^*), (R^*, G_R^*), Choice, (x_0, x_1, x, c, z))\\ &\qquad > \mathcal{U}_R((S^\pi, G_S^*), (R^\pi, G_R^*), Choice, (x_0, x_1, x, c, z)) + \epsilon_R(n).\end{aligned}$$

When Case 1 holds, then for at least one of the three terms of the utility function $\mathcal{U}_S$, the expectation on $((S^*, G_S^*), (R^\pi, G_R^*))$ is greater than that on $((S^\pi, G_S^*), (R^\pi, G_R^*))$. That is, at least one of the next formulae holds, where ϵ_1, ϵ_2 and ϵ_3 are non-negligible functions.

$$\begin{aligned}&\Pr[G_R^*(\mathsf{history}_{\pi,R}(S^*(X^b), R^\pi(c)), X^0, X^1) = b]\\ &\ < \Pr[G_R^*(\mathsf{history}_{\pi,R}(S^\pi(X^b), R^\pi(c)), X^0, X^1) = b] - \epsilon_1(n) \qquad (3)\\ &\Pr[\mathsf{fin}_\pi(S^*(X^b), R^\pi(c)) = 0 \vee \mathsf{output}_{\pi,R}(S^*(X^b), R^\pi(c)) = x_c]\\ &\ > \Pr[\mathsf{fin}_\pi(S^\pi(X^b), R^\pi(c)) = 0 \vee \mathsf{output}_{\pi,R}(S^\pi(X^b), R^\pi(c)) = x_c] + \epsilon_2(n) \qquad (4)\\ &\Pr[G_S^*(\mathsf{history}_{\pi,S}(S^*(X^b), R^\pi(c))) = c]\\ &\ > \Pr[G_S^*(\mathsf{history}_{\pi,S}(S^\pi(X^b), R^\pi(c))) = c] + \epsilon_3(n) \qquad (5)\end{aligned}$$

When formula (3) holds, we have

$$\begin{aligned}&\Pr[G_R^*(\mathsf{history}_{\pi,R}(S^\pi(X^b), R^\pi(c)), X^0, X^1) = b]\\ &\ > \Pr[G_R^*(\mathsf{history}_{\pi,R}(S^*(X^b), R^\pi(c)), X^0, X^1) = b] + \epsilon_1(n)\\ &\ \geq 1/2 + \epsilon_1(n), \qquad (6)\end{aligned}$$

since $b \in \{0,1\}$ is chosen uniformly at random and any valid algorithm succeed in guessing b with probability at least $1/2$.

Let D_R be an algorithm that runs G_R^* on its own input after the execution of Game^π, and outputs what G_R^* outputs. Obviously D_R runs in probabilistic polynomial-time. Since $\mathsf{history}_{\pi,R}(S^\pi(X^b), R^\pi(c)) = \mathsf{view}_{\pi,R}(S^\pi(X^b), R^\pi(c))$, it holds that

$$\begin{aligned}&\Pr[D_R(\mathsf{view}_{\pi,R}(S^\pi(X^1), R^\pi(c))) = 1]\\ &\qquad - \Pr[D_R(\mathsf{view}_{\pi,R}(S^\pi(X^0), R^\pi(c))) = 1] > 2\epsilon_1(n)\end{aligned}$$

by formula (6). It means that π is not cryptographically private for the sender.

When formula (4) holds, we have

$$\begin{aligned}
&\Pr[\mathsf{output}_{\pi,R}(S^\pi(X^b), R^\pi(c,z)) = x_c] \\
&\leq \Pr[\mathsf{fin}_\pi(S^\pi(X^b), R^\pi(c,z)) = 0 \vee \mathsf{output}_{\pi,R}(S^\pi(X^b), R^\pi(c,z)) = x_c] \\
&\leq \Pr[\mathsf{fin}_\pi(S^*(X^b), R^\pi(c,z)) = 0 \vee \mathsf{output}_{\pi,R}(S^*(X^b), R^\pi(c,z)) = x_c] - \epsilon_2(n) \\
&\leq 1 - \epsilon_2(n).
\end{aligned}$$

This means that π is not cryptographically correct.

And when formula (5) holds, we have

$$\begin{aligned}
&\Pr[G_S^*(\mathsf{history}_{\pi,S}(S^*(X^b), R^\pi(c))) = c] \\
&> \Pr[G_S^*(\mathsf{history}_{\pi,S}(S^\pi(X^b), R^\pi(c))) = c] + \epsilon_3(n) \\
&\geq 1/2 + \epsilon_3(n). \qquad (7)
\end{aligned}$$

Let D_S be an algorithm that runs G_S^* on its own input after the execution of Game^π, and outputs what G_S^* outputs. Obviously D_S runs in probabilistic polynomial-time, and it holds that

$$\Pr[D_S(\mathsf{view}_{\pi,S}(S^*(X^b), R^\pi(1))) = 1] - \Pr[D_S(\mathsf{view}_{\pi,S}(S^*(X^b), R^\pi(0))) = 1] > 2\epsilon_3(n)$$

by formula (7). It means that π is not cryptographically private for the receiver.

From a similar consideration, Case 2 also implies that π is not cryptographically secure.

Therefore, π is not cryptographically secure if it is not game-theoretically secure. □

3.4 Game-Theoretic Security Implies Cryptographic Security

In this section, we prove the "if" part of Theorem 1. That is, we prove the following lemma.

Lemma 2. *An OT protocol π is cryptographically secure if π is game-theoretically secure.*

To prove Lemma 2, we assume that a protocol is not cryptographically secure and show that nor is it game-theoretically secure. Namely, we show that the pair (S^π, R^π) is not a Nash equilibrium. For the sake of convenience, we divide the lemma into two lemmas depending on whether the protocol is cryptographically correct or not, and prove them one by one.

Lemma 3. *If an OT protocol π is not cryptographically correct, then π is not game-theoretically secure.*

Proof. To prove this lemma, we assume that an OT protocol is not cryptographically correct, and show that the pair of algorithms (S^π, R^π) is not a Nash equilibrium. Namely, we show that the existence of an alternative pair of algorithms which achieves a higher utility than the pair (S^π, R^π).

Let $\pi = (S^\pi, R^\pi)$ be an OT protocol which is not cryptographically correct. Then, there exist $x_0, x_1 \in \{0,1\}^*$ such that $|x_0| = |x_1|$, $c \in \{0,1\}$ and a non-negligible function ϵ such that

$$\Pr[\mathsf{output}_{\pi,R}(S^\pi(x_0,x_1), R^\pi(c)) = x_c] < 1 - \epsilon(n).$$

Let G be an algorithm that outputs 0 or 1 uniformly at random, and S^{def} be an algorithm of the sender which stops the protocol before it starts. Then, for every function *Choice* the expected utilities are as follows.

$$\begin{aligned}
\mathcal{U}_S((S^\pi, G), (R^\pi, G)) &< -\alpha_S(1/2 - 1/2) + \beta_S(1 - \epsilon(n) - 1) + \gamma_S(1/2 - 1/2) \\
&= -\beta_S \epsilon(n) \\
\mathcal{U}_S((S^{\mathsf{def}}, G), (R^\pi, G)) &= -\alpha_S(1/2 - 1/2) + \beta_S(1 - 1) + \gamma_S(1/2 - 1/2) \\
&= 0
\end{aligned}$$

Then we have

$$\mathcal{U}_S((S^{\mathsf{def}}, G), (R^\pi, G)) - \mathcal{U}_S((S^\pi, G), (R^\pi, G)) > \beta_S \epsilon(n)$$

which is a non-negligible value. Namely, the pair (S^π, R^π) is not a Nash equilibrium, which concludes that π is not game-theoretically secure. □

Before proving the next lemma, we briefly look at the receiver's privacy. If there is an OT protocol π that is not cryptographically private for the receiver, then there exist a sender's algorithm S^* and a guessing algorithm G_S^* and the sender can guess the receiver's input that the algorithm R^π used. Recall that π is a two-message OT protocol and the input of G_S^* is the history of the sender $\mathsf{history}_{\pi,S}(S^*(X^b), R^\pi(c))$. The output of G_S^* relies on whether the sender have received a message from the receiver, and not on how the sender behaves after she receives a message from him. Namely, if π is not private for the receiver, then it is not private even against the semi-honest receiver. Therefore the next proposition holds.

Proposition 1. *If $\pi = (S^\pi, R^\pi)$ is not cryptographically private for the receiver, then there exists a probabilistic polynomial-time distinguisher D, a non-negligible function ϵ and a tuple (x_0, x_1) where $|x_0| = |x_1|$ such that,*

$$\begin{aligned}
&\Pr[D(\mathsf{view}_{\pi,S}(S^\pi(x_0,x_1), R^\pi(1))) = 1] \\
&\qquad\qquad - \Pr[D(\mathsf{view}_{\pi,S}(S^\pi(x_0,x_1), R^\pi(0))) = 1] > \epsilon(n).
\end{aligned}$$

Using this proposition, we prove the next lemma.

Lemma 4. *If an OT protocol π is not cryptographically secure but cryptographically correct, then π is not game-theoretically secure.*

Proof. Similar to the proof of Lemma 3, we show that the pair (S^π, R^π) is not a Nash equilibrium.

Let $\pi = (S^\pi, R^\pi)$ be an OT protocol which is not cryptographically secure but cryptographically correct. Recall that π is not private for the receiver means that it is not private against the honest sender by Proposition 1. To prove this lemma, we divide such π into three groups according to the privacy of the players.

First, we assume that the formula (1) in Definition 3 does not hold. Then there exist a probabilistic polynomial-time algorithm D_1, a non-negligible function μ_R, and a tuple (x_0, x_1, x, c) where $|x_0| = |x_1| = |x|$, $c \in \{0,1\}$ such that

$$\begin{aligned}&\Pr[D_1(\mathsf{view}_{\pi,R}(S^\pi(X^1), R^\pi(c))) = 1]\\ &\qquad\qquad - \Pr[D_1(\mathsf{view}_{\pi,R}(S^\pi(X^0), R^\pi(c))) = 1] > 2\mu_R(n),\end{aligned}$$

where $X^0 = (x_0, x_1)$, and $X^1 = (x_0, x)$ if $c = 0$ and $X^1 = (x, x_1)$ otherwise. Let G_R be an algorithm that runs D_1 on its own input and outputs what D_1 outputs. Then, for every function *Choice* the expected utilities are as follows.

$$\begin{aligned}&\mathcal{U}_S((S^\pi, G), (R^\pi, G_R))\\ &\;< -\alpha_S(1/2 + \mu_R(n) - 1/2) + \beta_S(1 - \mathit{negl}(n) - 1) + \gamma_S(1/2 - 1/2)\\ &\;< -\alpha_S \mu_R(n)\\ &\mathcal{U}_S((S^{\mathsf{stop}}, G), (R^\pi, G_R))\\ &\;= -\alpha_S(1/2 - 1/2) + \beta_S(1 - \mathit{negl}(n) - 1) + \gamma_S(1/2 - 1/2)\\ &\;> -\mathit{negl}(n)\end{aligned}$$

Here, S^{stop} is an algorithm of the sender which stops the protocol when she receives a message from the receiver. Then we have

$$\mathcal{U}_S((S^{\mathsf{stop}}, G), (R^\pi, G_R)) - \mathcal{U}_S((S^\pi, G), (R^\pi, G_R)) > \alpha_S \mu_R(n) - \mathit{negl}(n),$$

which is non-negligible.

Secondly, we assume that π is not cryptographically private for the receiver and formula (1) holds. Then there exist a probabilistic polynomial-time algorithm D_2, a non-negligible function μ_S and a tuple (x_0, x_1) where $|x_0| = |x_1|$ such that

$$\begin{aligned}&\Pr[D_2(\mathsf{view}_{\pi,S}(S^\pi(x_0, x_1), R^\pi(1))) = 1]\\ &\qquad\qquad - \Pr[D_2(\mathsf{view}_{\pi,S}(S^\pi(x_0, x_1), R^\pi(0))) = 1] > 2\mu_S(n).\end{aligned}$$

Let G_S be an algorithm that runs D_2 on its own input and outputs what D_2 outputs. Then, the expected utilities are as follows.

$$\begin{aligned}&\mathcal{U}_R((S^\pi, G_S), (R^\pi, G))\\ &\;< -\alpha_R(1/2 + \mu_S(n) - 1/2) + \beta_R(1 - \mathit{negl}(n) - 1) + \gamma_R(1/2 - 1/2)\\ &\;< -\alpha_R \mu_S(n)\\ &\mathcal{U}_R((S^\pi, G_S), (R^{\mathsf{def}}, G)) = 0\end{aligned}$$

Here, R^{def} is an algorithm of the receiver which stops the protocol before it starts. Therefore, for every function *Choice* we have

$$\mathcal{U}_R((S^\pi, G_S), (R^{\mathsf{def}}, G)) - \mathcal{U}_R((S^\pi, G_S), (R^\pi, G)) > \alpha_R \mu_S(n)$$

which is non-negligible.

Finally, we assume that π is cryptographically private for the receiver and formula (1) holds but formula (2) does not holds. Then there exist a deterministic polynomial-time algorithm R^* a probabilistic polynomial-time algorithm D_3, a non-negligible function μ_R, a function *Choice* and a tuple (x_0, x_1, x, c, z) where $|x_0| = |x_1| = |x|$ and $c \in \{0, 1\}$, such that

$$\Pr[D_3(\mathsf{view}_{\pi,R}(S^\pi(X^1), R^*(c, z))) = 1] - \Pr[D_3(\mathsf{view}_{\pi,R}(S^\pi(X^0), R^*(c, z))) = 1] > 2\mu_R(n),$$

where $X^0 = (x_0, x_1)$, and $X^1 = (x_0, x)$ if $Choice(R^*, c, z) = 0$ and $X^1 = (x, x_1)$ otherwise.

Let G_R be an algorithm that runs D_3 on its own input and outputs what D_3 outputs. Then, the expected utilities are as follows.

$$\begin{aligned}
&\mathcal{U}_R((S^\pi, G), (R^\pi, G_R)) \\
&= -\alpha_R(1/2 - 1/2) + \beta_R(1 - negl(n) + 1) + \gamma_R(1/2 + negl(n) - 1/2) \\
&< negl(n) \\
&\mathcal{U}_R((S^\pi, G), (R^*, G_R)) \\
&> -\alpha_R(1/2 - 1/2) + \beta_R(1 - negl(n) - 1) + \gamma_R(1/2 + \mu_R(n) - 1/2) \\
&> \gamma_R\mu_R(n) - negl(n)
\end{aligned}$$

Therefore, for every function *Choice* we have

$$\mathcal{U}_R((S^\pi, G), (R^*, G_R)) - \mathcal{U}_R((S^\pi, G), (R^\pi, G_R)) > \gamma_R\mu_R(n) - negl(n)$$

which is non-negligible.

With all cases, we prove that the pair (S^π, R^π) is not a Nash equilibrium, which concludes that π is not game-theoretically secure. □

References

1. Abraham, I., Dolev, D., Gonen, R., Halpern, J.Y.: Distributed Computing Meets Game Theory: Robust Mechanisms for Rational Secret Sharing and Multiparty Computation. In: Ruppert, E., Malkhi, D. (eds.) PODC, pp. 53–62. ACM (2006)
2. Asharov, G., Canetti, R., Hazay, C.: Towards a Game Theoretic View of Secure Computation. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 426–445. Springer, Heidelberg (2011)
3. Asharov, G., Lindell, Y.: Utility Dependence in Correct and Fair Rational Secret Sharing. J. Cryptology 24(1), 157–202 (2011)
4. Ding, Y.Z., Harnik, D., Rosen, A., Shaltiel, R., Shaltiel, R.: Constant-Round Oblivious Transfer in the Bounded Storage Model. J. Cryptology, 165–202 (2007)
5. Dodis, Y., Rabin, T.: Cryptography and Game Theory. In: Nisan, N., Roughgarden, T., Tardos, É., Vazirani, V.V. (eds.) Algorithmic Game Theory, ch. 8, pp. 181–205. Cambridge University Press (2007)
6. Fuchsbauer, G., Katz, J., Naccache, D.: Efficient Rational Secret Sharing in Standard Communication Networks. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 419–436. Springer, Heidelberg (2010)

7. Gordon, S.D., Katz, J.: Rational Secret Sharing, Revisited. In: De Prisco, R., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 229–241. Springer, Heidelberg (2006)
8. Halevi, S., Kalai, Y.T.: Smooth Projective Hashing and Two-Message Oblivious Transfer. J. Cryptology 25(1), 158–193 (2012)
9. Halpern, J.Y., Pass, R., Pass, R.: Game Theory with Costly Computation: Formulation and Application to Protocol Security. In: ICS, pp. 120–142 (2010)
10. Halpern, J.Y., Teague, V.: Rational Secret Sharing and Multiparty Computation: Extended Abstract. In: Babai, L. (ed.) STOC, pp. 623–632. ACM (2004)
11. Hazay, C., Lindell, Y.: Efficient Secure Two-Party Protocols: Techniques and Constructions. Information Security and Cryptography Series. Springer, Heidelberg (2010)
12. Kol, G., Naor, M.: Cryptography and Game Theory: Designing Protocols for Exchanging Information. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 320–339. Springer, Heidelberg (2008)
13. Kol, G., Naor, M.: Games for Exchanging Information. In: Dwork, C. (ed.) STOC, pp. 423–432. ACM (2008)
14. Lysyanskaya, A., Triandopoulos, N.: Rationality and Adversarial Behavior in Multi-party Computation. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 180–197. Springer, Heidelberg (2006)
15. Micali, S., Shelat, A.: Purely Rational Secret Sharing (Extended Abstract). In: Reingold [17], pp. 54–71
16. Ong, S.J., Parkes, D.C., Rosen, A., Vadhan, S.P.: Fairness with an Honest Minority and a Rational Majority. In: Reingold [17], pp. 36–53
17. Reingold, O. (ed.): TCC 2009. LNCS, vol. 5444. Springer, Heidelberg (2009)

Faster Algorithm for Solving Hard Knapsacks for Moderate Message Length

Yuji Nagashima and Noboru Kunihiro

The University of Tokyo
{y_nagashima@it.,kunihiro@}k.u-tokyo.ac.jp

Abstract. At Eurocrypt2011, Becker, Coron and Joux proposed an algorithm for solving hard knapsacks, i.e., knapsacks with a density close to 1. Their algorithm solves hard knapsacks in time $\tilde{O}(2^{0.2909n})$. In this paper, we evaluate their algorithm by O notation and prove that the running time is $O(n^{3.5} \cdot 2^{0.2909n})$. Furthermore, we extend their algorithm and propose the algorithm of which running time is $O(n^3 \cdot 2^{0.2919n})$. Asymptotic running time of our algorithm is not faster. However, when $n < 6312$, our algorithm can solve subset sum problem faster than algorithm of Becker, Coron and Joux.

Keywords: knapsack cryptosystem, subset sum problem, hard knapsacks.

1 Introduction

Knapsack cryptosystem is based on the difficulty of subset sum problem. When a set of positive integers $(a_1, \ldots, a_n)$ and a positive integer S is given, finding a vector $(\epsilon_1, \ldots, \epsilon_n) \in \{0, 1\}^n$ satisfying

$$S = \sum_{i=1}^{n} \epsilon_i a_i$$

is subset sum problem. In this paper, we assume that $\sum_{i=1}^{n} \epsilon_i = n/2$ and we can determine $(\epsilon_1, \ldots, \epsilon_n)$ uniquely from S and $(a_1, \ldots, a_n)$. This problem is NP-hard [3], and polynomial time algorithms are not known. Subset sum problem was introduced in cryptography by Merkle and Hellman [6] in 1978.

We define the density of a knapsack as:

$$d := \frac{n}{\log_2 \max_i a_i}.$$

When the density d of a knapsack is low, some attacks are proposed. If d is lower than 0.64, Lagarias and Odlyzko [5] proved that knapsack schemes are broken by using lattice. Furthermore, Coster et al. [2] proposed an improved algorithm which can solve the schemes with higher density $d < 0.94$. However, there is no effective lattice-based approach to solve subset sum problem for knapsacks with density close to 1. Such knapsacks are called *hard knapsacks* in [1], [4].

For hard knapsacks, the best algorithm was due to Schroeppel and Shamir [7] with time complexity $\tilde{O}(2^{0.5n})$ until 2009 (see Appendix 2.1 about the definition of $\tilde{O}$ notation). At Eurocrypt2010, Howgrave-Graham and Joux [4] proposed the algorithm with time complexity $\tilde{O}(2^{0.337n})$. Furthermore, Becker et al. [1] proposed an improved algorithm of Howgrave-Graham and Joux which can solve subset sum problem in time $\tilde{O}(2^{0.2909n})$ for any density. We call this algorithm *BCJ algorithm*. By decomposing a knapsack into eight subknapsacks, they succeed in reducing running time. In [1], they estimated the running time by $\tilde{O}$ notation. When n is not so large, estimation by $\tilde{O}$ notation is inadequate, so we estimate detail running time by O notation.

In this paper, first we analyze BCJ algorithm in detail and we prove that the running time of BCJ algorithm is $O(n^{3.5} \cdot 2^{0.2909n})$. Second, we extend BCJ algorithm and prove that our algorithm can solve subset sum problem in $O(n^3 \cdot 2^{0.2919n})$. Specifically, our algorithm decomposes a knapsack into 16 subknapsacks. Our algorithm can solve subset sum problem more efficiently when n is not so large, and $n < 6312$ in particular.

This paper is organized as follows: In Sect. 2 we introduce some preliminaries. In Sect. 3 we introduce the basic principle of BCJ algorithm. In Sect. 4 we analyze BCJ algorithm in detail. In Sect. 5 we extend BCJ algorithm and analyze our algorithm, finally in Sect. 6 we introduce extensions.

2 Preliminaries

2.1 $\tilde{O}$ Notation

Let $f(n)$ and $g(n)$ be function of n and i be a fixed value. We write

$$f(n) = \tilde{O}(g(n)),$$

if there is constants i such that

$$g(n) = O(f(n) \cdot (\log f(n))^i).$$

In particular, we write $g(n) = \tilde{O}(2^{cn})$ if $g(n)$ equal to $O(n^i \cdot 2^{cn})$. When n is not so large, O notation allows us to analyze time cost in detail.

2.2 Asymptotic Values of Binomials and Multinomials

When we analyze running time of algorithm, we need to obtain asymptotic approximation for binomials and multinomials. In this section, we introduce approximation method.

Let n be a positive number. When $n \gg 1$, we approximate $n!$ by using Stirling's formula as:

$$n! = (1 + o(1)) \sqrt{2\pi n} \left(\frac{n}{e}\right)^n .$$

By using this formula, we find

$$\binom{n}{\alpha n} = \frac{n!}{(\alpha n)! \cdot ((1-\alpha)n)!} \approx \frac{1}{\sqrt{n}} \cdot 2^{h(\alpha)n},$$

for fixed value $0 < \alpha < 1$, where $h(\alpha) := -\alpha \log_2 \alpha - (1-\alpha)\log_2(1-\alpha)$. Furthermore, we approximate as:

$$\binom{an}{bn} \approx \frac{1}{\sqrt{n}} \cdot 2^{a \cdot h(\frac{b}{a})n}.$$

We approximate multinomials with $0 < \alpha, \beta < 10 < \alpha + \beta < 1$ as:

$$\binom{n}{\alpha n, \beta n, (1-\alpha-\beta)n} = \frac{n!}{(\alpha n)! \cdot (\beta n)! \cdot ((1-\alpha-\beta)\,n)!} \approx \frac{1}{n} \cdot 2^{h(\alpha,\beta)n},$$

where $h(\alpha, \beta) := -\alpha \log_2 \alpha - \beta \log_2 \beta - (1-\alpha-\beta)\log_2(1-\alpha-\beta)$. Furthermore, we find

$$\binom{an}{bn, cn, dn} = \binom{an}{bn, cn, *} \approx \frac{1}{n} \cdot 2^{a \cdot h(\frac{b}{a}, \frac{c}{a})n},$$

where $d = a - b - c$. The symbol $*$ in the above multinomial denotes the number of remaining elements.

3 Basic Principle of BCJ Algorithm [1]

In this section, we introduce the basic principle of BCJ algorithm. This algorithm decomposes a knapsack into two subknapsacks three times, that is to say, it finally decomposes a knapsack into eight subknapsacks. By solving modular knapsack problems for eight subknapsacks, we obtain the solution of original subset sum problem.

3.1 Decomposing Knapsack

In this section, we introduce the idea of splitting a knapsack to reduce the overall running time. Given a list of n positive integers $(a_1, \ldots, a_n)$ and a non-negative integer σ such that:

$$\sigma = \sum_{i=1}^{n} x_i a_i,$$

where $x_i \in \{0, 1\}$, we split the knapsack as:

$$\underbrace{\sum_{i=1}^{n} x_i a_i}_{\sigma} = \underbrace{\sum_{i=1}^{n} y_i a_i}_{\sigma_1} + \underbrace{\sum_{i=1}^{n} z_i a_i}_{\sigma_2} \tag{1}$$

where $y_i, z_i \in \{-1, 0, 1\}$. By this split, we reduce the overall running time of recovering the coefficients x_i. We assume that $\sum_{i=1}^{n} x_i = n/2$ and we can determine x from σ and $(a_1, \ldots, a_n)$ uniquely.

By this decomposition, we decompose a single solution of the original subset sum problem into many different representations. In other words, we can represent x_i as:

$$\begin{cases} x_i = 1 \Rightarrow (y_i, z_i) = (1, 0), (0, 1) \\ x_i = 0 \Rightarrow (y_i, z_i) = (0, 0), (-1, 1), (1, -1) \end{cases}$$

by a tuple (y_i, z_i). When x contains $n/2$ 1s and $n/2$ 0s, we decompose a knapsack so that y and z contain $(1/4 + \alpha)n$ 1s and αn -1s, where $0 < \alpha < 3/8$. The number of such decompositions is

$$M = \binom{n/2}{(1/4 + \alpha)n} \cdot \binom{n/2}{\alpha n, \alpha n, *}.$$

In other words, when we split the knapsack as Eq. (1), there exist M pairs (σ_1, σ_2) which satisfy $\sigma = \sigma_1 + \sigma_2$. We use this to reduce the overall running time. We introduce a modulus M and a random element $R \in \mathbb{Z}_M$ and we only use σ_1 and σ_2 which satisfy

$$\sigma_1 \left(= \sum_{i=1}^{n} y_i a_i \right) \equiv R \pmod{M}, \ \sigma_2 \left(= \sum_{i=1}^{n} z_i a_i \right) \equiv \sigma - R \pmod{M}$$

to solve the original subset sum problem.

3.2 Decomposing Knapsack into Eight Subknapsacks

We extend the idea in Sect. 3.1, that is to say, we decompose the original knapsack into eight subknapsacks. More specifically, we first decompose the knapsack as:

$$\underbrace{\sum_{i=1}^{n} \epsilon_i a_i}_{\sigma_\epsilon = S} = \underbrace{\sum_{i=1}^{n} \nu_i^{(1)} a_i}_{\sigma_\nu^{(1)}} + \underbrace{\sum_{i=1}^{n} \nu_i^{(2)} a_i}_{\sigma_\nu^{(2)}}.$$

Second, we decompose the knapsack as:

$$\underbrace{\sum_{i=1}^{n} \nu_i^{(j)} a_i}_{\sigma_\nu^{(j)}} = \underbrace{\sum_{i=1}^{n} \kappa_i^{(2j-1)} a_i}_{\sigma_\kappa^{(2j-1)}} + \underbrace{\sum_{i=1}^{n} \kappa_i^{(2j)} a_i}_{\sigma_\kappa^{(2j)}} \ (j = 1, 2).$$

Last, we decompose the knapsack as:

$$\underbrace{\sum_{i=1}^{n} \kappa_i^{(j)} a_i}_{\sigma_\kappa^{(j)}} = \underbrace{\sum_{i=1}^{n} \omega_i^{(2j-1)} a_i}_{\sigma_\omega^{(2j-1)}} + \underbrace{\sum_{i=1}^{n} \omega_i^{(2j)} a_i}_{\sigma_\omega^{(2j)}} \ (j = 1, \ldots, 4).$$

We can denote the partial sum of level χ as:

$$\sigma_\chi^{(j)} = \sum_{i=1}^{n} \chi_i^{(j)} a_i,$$

where $\chi \in \{\epsilon, \nu, \kappa, \omega\}$.

We denote the ratio of occurrences of $x \in \{-1, 0, 1\}$ in the coefficient vector χ by $N_\chi(x)$. For a knapsack element, we have

$$\epsilon \begin{cases} N_\epsilon(1) = 1/2 \\ N_\epsilon(-1) = 0 \\ N_\epsilon(0) = 1/2, \end{cases} \tag{2}$$

$$\nu \begin{cases} N_\nu(1) = 1/4 + \alpha \\ N_\nu(-1) = \alpha \\ N_\nu(0) = 3/4 - 2\alpha, \end{cases} \tag{3}$$

$$\kappa \begin{cases} N_\kappa(1) = 1/8 + \alpha/2 + \beta \\ N_\kappa(-1) = \alpha/2 + \beta \\ N_\kappa(0) = 7/8 - \alpha - 2\beta, \end{cases} \tag{4}$$

$$\omega \begin{cases} N_\omega(1) = 1/16 + \alpha/4 + \beta/2 + \gamma \\ N_\omega(-1) = \alpha/4 + \beta/2 + \gamma \\ N_\omega(0) = 15/16 - \alpha/2 - \beta - 2\gamma. \end{cases} \tag{5}$$

We always have $N_\chi(-1) + N_\chi(0) + N_\chi(1) = 1$. Parameters α, β and γ denote the proportion of extra -1s in the subknapsacks. We control these parameters to minimize the overall running time. Parameters α, β and γ satisfy $0 < \alpha, \beta, \gamma < 1$ and $0 < N_\chi(x) < 1$.

We introduce a modulus and a target value for each subknapsack to reduce overall running time. First, we consider the eight subknapsacks at the level ω. We denote the modulus corresponding to the level ω by M_ω. We choose random values $R_\omega^{(j)}$ $(j = 1, \ldots, 7)$ and let $R_\omega^{(8)} = S - \sum_{j=1}^{7} R_\omega^{(j)}$. We solve the eight modular subknapsacks:

$$\sigma_\omega^{(j)} \equiv R_\omega^{(j)} \pmod{M_\omega}. \tag{6}$$

We denote a list of values $\sigma_\omega^{(j)}$ satisfying Eq. (6) by $\mathbb{L}_\omega^{(j)}$.

Next, we consider the four modular subknapsacks at the level κ. To build list $\mathbb{L}_\kappa^{(j)}$, we use the lists $\mathbb{L}_\omega^{(2j-1)}$ and $\mathbb{L}_\omega^{(2j)}$. We choose a modulus M_κ which is coprime to M_ω and random values $R_\kappa^{(j)}$ $(j = 1, 2, 3)$ and let $R_\kappa^{(4)} = S - \sum_{j=1}^{3} R_\kappa^{(j)}$. We solve the four modular subknapsacks:

$$\sigma_\kappa^{(j)} (= \sigma_\omega^{(2j-1)} + \sigma_\omega^{(2j)}) \equiv R_\kappa^{(j)} \pmod{M_\kappa}. \tag{7}$$

We denote a list of values $\sigma_\kappa^{(j)}$ satisfying Eq. (7) by $\mathbb{L}_\kappa^{(j)}$.

At the level ν, we choose a modulus M_ν and a random value $R_\nu^{(1)}$ and let $R_\nu^{(2)} = S - R_\nu^{(1)}$. We build lists $\mathbb{L}_\nu^{(1)}$ and $\mathbb{L}_\nu^{(2)}$ in the same way at the level κ.

Last, we search for the solution of the original knapsack by searching for a collision of the form $\sigma_\nu^{(1)} + \sigma_\nu^{(2)} = S$ with $\sigma_\nu^{(1)} \in \mathbb{L}_\nu^{(1)}$ and $\sigma_\nu^{(2)} \in \mathbb{L}_\nu^{(2)}$.

4 Analyzing Detailed Running Time of BCJ Algorithm

In this section, we analyze the running time of BCJ algorithm in detail. While [1] analyzed their algorithm by $\tilde{O}$ notation and proved that the running time was $\tilde{O}(2^{0.2909n})$, we analyze the running time in detail by O notation.

4.1 Details of Every Step

In this section, we explain the detail steps of BCJ algorithm and analyze the running time in detail.

Steps at the Level $\boldsymbol{\omega}$. We explain the steps at the level ω. In particular, we explain how to construct list $\mathbb{L}_\omega^{(j)}$.

First, we assume that an ω vector has $N_\omega(1)n/2$ 1s, $N_\omega(-1)n/2$ -1s and $N_\omega(0)n/2$ 0s among first half $n/2$ elements. This assumption holds with probability $2/(n\pi\sqrt{N_\omega(1)N_\omega(-1)N_\omega(0)})$ when we randomly choose an ω vector satisfying Eq. (5). We do not know whether this assumption holds in advance, so we need to run algorithm to the end. Therefore, we need to repeat the overall algorithm $O(n)$ times in average. We construct the lists $\mathcal{L}_{\omega/2}^{(1)}$ and $\mathcal{L}_{\omega/2}^{(2)}$. The list $\mathcal{L}_{\omega/2}^{(1)}$ is list of values $\sum_{i=1}^{n/2} \omega_i a_i$ and the list $\mathcal{L}_{\omega/2}^{(2)}$ is list of values $\sum_{i=n/2+1}^{n} \omega_i a_i$, where the ω vector satisfies the assumption. The average size of lists $\mathcal{L}_{\omega/2}^{(1)}$ and $\mathcal{L}_{\omega/2}^{(2)}$ is

$$\mathcal{L}_{\omega/2} = \binom{n/2}{N_\omega(1)n/2, N_\omega(-1)n/2, N_\omega(0)n/2} \approx \frac{1}{n} \cdot 2^{\left(\frac{1}{2}h(N_\omega(1), N_\omega(-1))\right)n}.$$

Next, we let M_ω the modulus at the level ω and choose random integers $R_\omega^{(j)}$ $(j = 1, \ldots, 7)$ and let $R_\omega^{(8)} = S - \sum_{j=1}^{7} R_\omega^{(j)}$. From the lists $\mathcal{L}_{\omega/2}^{(1)}$ and $\mathcal{L}_{\omega/2}^{(2)}$, the target sum $R_\omega^{(j)}$ and the modulus M_ω, we construct the list $\mathbb{L}_\omega^{(j)}$ by list algorithm(see Append A about list algorithm). The list $\mathbb{L}_\omega^{(j)}$ is list of values $\sigma_\omega^{(j)}$ satisfying $\sigma_\omega^{(j)} \equiv R_\omega^{(j)} \pmod{M_\omega}$. The average size of the list $\mathbb{L}_\omega^{(j)}$ is $L_\omega = \mathcal{L}_{\omega/2}^2 / M_\omega$.

When the assumption holds, the average running time of these steps is $O(\max(n \cdot \mathcal{L}_{\omega/2}, n \cdot L_\omega))$.

Steps at the Level $\boldsymbol{\kappa}$. We explain the steps at the level κ. In particular, we present how to construct list $\mathbb{K}_\kappa^{(j)}$ from the lists $\mathbb{L}_\omega^{(2j-1)}$ and $\mathbb{L}_\omega^{(2j)}$ and the list $\mathbb{L}_\kappa^{(j)}$ from $\mathbb{K}_\kappa^{(j)}$.

First, we let M_κ the modulus at the level κ and choose random integers $R_\kappa^{(j)} (j = 1, 2, 3)$ and let $R_\kappa^{(4)} = S - \sum_{j=1}^{3} R_\kappa^{(j)}$. Here, we choose M_κ so that it is coprime to M_ω. From the lists $\mathbb{L}_\omega^{(2j-1)}$ and $\mathbb{L}_\omega^{(2j)}$, the target sum $R_\kappa^{(j)}$ and the modulus M_κ, we construct the list $\mathbb{K}_\kappa^{(j)}$ by list algorithm. The list $\mathbb{K}_\kappa^{(j)}$ is the list of values $\sigma_\kappa^{(j)}$ satisfying $\sigma_\kappa^{(j)} \equiv R_\kappa^{(j)} \pmod{M_\kappa}$. The average size of the list $\mathbb{K}_\kappa^{(j)}$ is $K_\kappa = L_\omega^2 / M_\kappa$.

The list $\mathbb{K}_\kappa^{(j)}$ includes some inconsistent κ vectors which do not satisfy Eq. (4) or contain 2s or -2s. So we need to remove some inconsistent solutions from $\mathbb{K}_\kappa^{(j)}$ in order to produce list $\mathbb{L}_\kappa^{(j)}$. To complete this step, we must check all κ vector's elements, so cost of this step is $O(n \cdot K_\kappa^{(j)})$. The average size of the list $\mathbb{L}_\kappa^{(j)}$ is $L_\kappa = \mathcal{L}_\kappa / (M_\kappa \cdot M_\omega)$, where

$$\mathcal{L}_\kappa = \binom{n}{N_\kappa(1)n, N_\kappa(-1)n, N_\kappa(0)n} \approx \frac{1}{n} \cdot 2^{h(N_\kappa(1), N_\kappa(-1))n}.$$

The average running time of these steps is $O(\max(n \cdot L_\omega, n \cdot K_\kappa))$.

Steps at the Level $\boldsymbol{\nu}$. We explain the steps at the level ν. In particular, we present how to construct the list $\mathbb{K}_\nu^{(j)}$ from lists $\mathbb{L}_\kappa^{(2j-1)}$ and $\mathbb{L}_\kappa^{(2j)}$ and the list $\mathbb{L}_\nu^{(j)}$ from the list $\mathbb{K}_\nu^{(j)}$.

First, let M_ν the modulus at the level ν and choose a random integer $R_\nu^{(1)}$ and let $R_\nu^{(2)} = S - R_\nu^{(1)}$. Here, we choose M_ν so that it is coprime to M_κ and M_ω.

Steps at the level ν are the same as the steps at the level κ. The average size of the list $\mathbb{K}_\nu^{(j)}$ is $K_\nu = L_\kappa^2 / M_\nu$. The average size of the list $\mathbb{L}_\nu^{(j)}$ is $L_\nu = \mathcal{L}_\nu / (M_\nu \cdot M_\kappa \cdot M_\omega)$, where

$$\mathcal{L}_\nu = \binom{n}{N_\nu(1)n, N_\nu(-1)n, N_\nu(0)n} \approx \frac{1}{n} \cdot 2^{h(N_\nu(1), N_\nu(-1))n}.$$

The average running time of these steps is $O(\max(n \cdot L_\kappa, n \cdot K_\nu))$.

Steps at the Level $\boldsymbol{\epsilon}$. We explain the steps at the level ϵ. In particular, we present how to construct list $\mathbb{K}_\epsilon$ from the lists $\mathbb{L}_\nu^{(1)}$ and $\mathbb{L}_\nu^{(2)}$ and obtain the solution of the original subset sum problem.

From the lists $\mathbb{L}_\nu^{(1)}$ and $\mathbb{L}_\nu^{(2)}$ and the target sum $\sigma_\epsilon (= S)$, we construct the list $\mathbb{K}_\epsilon$ by list algorithm. Here, we do not use a modulus. The upper size of the list $\mathbb{K}_\epsilon^{(j)}$ is $K_\epsilon = L_\nu^2 \cdot M_\nu \cdot M_\kappa \cdot M_\omega / 2^n$. This is because it is easier to find a solution modulo the value than a solution over the integers and a modulus should be smaller than 2^n.

The list $\mathbb{K}_\epsilon$ includes some inconsistent ϵ vectors which do not satisfy Eq. (2) or contain 2s, -2s or -1s. So we need to remove some inconsistent solutions from the list $\mathbb{K}_\epsilon$ in order to obtain the original solution.

The average running time of these steps is $O(\max(n \cdot L_\nu, n \cdot K_\epsilon))$.

Specifying the Values of Moduli. We specify the values of moduli M_ν, M_κ and M_ω. We specify moduli as we present in Sect. 3.1. In particular, we specify M_ω so that M_ω is the combination of decomposing a κ vector into an ω vector as:

$$\begin{aligned} M_\omega &= \binom{N_\kappa(1)n}{N_\kappa(1)n/2} \cdot \binom{N_\kappa(-1)n}{N_\kappa(-1)n/2} \\ &\quad \cdot \binom{N_\kappa(0)n}{(N_\omega(1) - N_\kappa(1)/2)n, (N_\omega(-1) - N_\kappa(-1)/2)n, *} \\ &\approx \frac{1}{n^2} \cdot 2^{\left(N_\kappa(1)+N_\kappa(-1)+N_\kappa(0)h\left(\frac{\gamma}{N_\kappa(0)}, \frac{\gamma}{N_\kappa(0)}\right)\right)n}. \end{aligned}$$

Next, we specify $M_\kappa \cdot M_\omega$ so that $M_\kappa \cdot M_\omega$ is the combination of decomposing a ν vector into a κ vector as:

$$\begin{aligned} M_\kappa \cdot M_\omega &= \binom{N_\nu(1)n}{N_\nu(1)n/2} \cdot \binom{N_\nu(-1)n}{N_\nu(-1)n/2} \\ &\quad \cdot \binom{N_\nu(0)n}{(N_\kappa(1) - N_\nu(1)/2)n, (N_\kappa(-1) - N_\nu(-1)/2)n, *} \\ &\approx \frac{1}{n^2} \cdot 2^{\left(N_\nu(1)+N_\nu(-1)+N_\nu(0)h\left(\frac{\beta}{N_\nu(0)}, \frac{\beta}{N_\nu(0)}\right)\right)n}. \end{aligned}$$

Lastly, we specify $M_\nu \cdot M_\kappa \cdot M_\omega$ so that $M_\nu \cdot M_\kappa \cdot M_\omega$ is the combination of decomposing an ϵ vector into a ν vector as:

$$\begin{aligned} M_\nu \cdot M_\kappa \cdot M_\omega &= \binom{N_\epsilon(1)n}{N_\epsilon(1)n/2} \cdot \binom{N_\epsilon(0)n}{N_\nu(-1)n, N_\nu(-1)n, *} \\ &\approx \frac{1}{\sqrt{n^3}} \cdot 2^{\left(N_\epsilon(1)+N_\epsilon(0)h\left(\frac{\alpha}{N_\epsilon(0)}, \frac{\alpha}{N_\epsilon(0)}\right)\right)n}. \end{aligned}$$

4.2 Running Time of BCJ Algorithm

When the assumption in Sect. 4.1 holds, the overall running time of BCJ algorithm is denoted by the function of n, α, β and γ as:

$$\begin{aligned} &T_0(n; \alpha, \beta, \gamma) \\ &= O(\max(n \cdot \mathcal{L}_{\omega/2}, n \cdot L_\omega, n \cdot K_\kappa, n \cdot L_\kappa, n \cdot K_\nu, n \cdot L_\nu, n \cdot K_\epsilon)). \end{aligned}$$

Without this assumption, we need to repeat overall algorithm $O(n)$ times in average. So the overall running time is denoted as:

$$T_1(n; \alpha, \beta, \gamma) = O(n \cdot T_o(n; \alpha, \beta, \gamma)) \tag{8}$$

Next, we adjust parameters to minimize the running time. When n is large enough, Eq. (8) is minimized with the values as:

$$\alpha = 0.026727, \beta = 0.016857, \gamma = 0.002901.$$

Thus, T_1 is denoted as:

$$\begin{aligned}
&T_1(n; 0.026727, 0.016857, 0.002901)\\
&= O(n \cdot (\max(2^{0.2660n}, n \cdot 2^{0.2909}, n \cdot 2^{0.2909n}, n^2 \cdot 2^{0.2789n},\\
&\quad n^{2.5} \cdot 2^{0.2909n}, n^{1.5} \cdot 2^{0.2173n}, n^{0.5} \cdot 2^{0.2331n})))\\
&= O(n^{3.5} \cdot 2^{0.2909n}).
\end{aligned}$$

So, without the assumption in Sect. 4.1, the overall running time is $O(n^{3.5} \cdot 2^{0.2909n})$. While the running time was estimated $\tilde{O}(2^{0.2909n})$ in [1], we estimate the running time $O(n^{3.5} \cdot 2^{0.2909n})$ in detail.

Note that since the list $\mathbb{K}_\chi$ need not to be stored, memory cost is:

$$O(\max(n \cdot \mathcal{L}_{\omega/2}, n \cdot L_\omega, n \cdot L_\kappa, n \cdot L_\nu)).$$

5 Our Algorithm

While BCJ algorithm requires the assumption in Sect. 4.1, our algorithm does not require the assumption and $O(n)$ repetitions. Our algorithm extends BCJ algorithm by decomposing subknapsack at the level ω into two subknapsacks. That is to say, we decompose the subknapsack as:

$$\underbrace{\sum_{i=1}^{n} \omega_i^{(j)} a_i}_{\sigma_\omega^{(j)}} = \underbrace{\sum_{i=1}^{n} \phi_i^{(2j-1)} a_i}_{\sigma_\phi^{(2j-1)}} + \underbrace{\sum_{i=1}^{n} \phi_i^{(2j)} a_i}_{\sigma_\phi^{(2j)}} \; (j = 1, \ldots, 8).$$

Let us call the fourth level *level* ϕ, and we introduce a parameter δ which controls the proportion of -1s at the level ϕ. The proportions of 1s, -1s and 0s are denoted as:

$$\phi \begin{cases} N_\phi(1) = 1/32 + \alpha/8 + \beta/4 + \gamma/2 + \delta \\ N_\phi(-1) = \alpha/8 + \beta/4 + \gamma/2 + \delta \\ N_\phi(0) = 31/32 - \alpha/4 - \beta/2 - \gamma - 2\delta. \end{cases}$$

5.1 Details of Every Step

In this section, we introduce the steps of our algorithm. However, the steps after the level κ are the same as BCJ algorithm's steps, so we do not explain them in detail.

Steps at the Level $\boldsymbol{\phi}$. First, we explain about steps at the level ϕ. In particular, we explain how to construct lists $\mathbb{L}_\phi^{(j)}$. First, we let M_ϕ the modulus at the level

ϕ and choose random integers $R_\phi^{(j)} (j = 1, \ldots, 15)$ and let $R_\phi^{(16)} = S - \sum_{j=1}^{15} R_\phi^{(j)}$. We specify the modulus M_ϕ in the same way as BCJ algorithm. Thus, M_ϕ is

$$\begin{aligned} M_\phi = &\binom{N_\omega(1)n}{N_\omega(1)n/2} \cdot \binom{N_\omega(-1)n}{N_\omega(-1)n/2} \\ &\cdot \binom{N_\omega(0)n}{(N_\phi(1) - N_\omega(1)/2)n, (N_\phi(-1) - N_\omega(-1)/2)n, *} \\ \approx &\frac{1}{n^2} \cdot 2^{\left(N_\omega(1)+N_\omega(-1)+N_\omega(0)h\left(\frac{\delta}{N_\omega(0)}, \frac{\delta}{N_\omega(0)}\right)\right)n}. \end{aligned}$$

We construct the list $\mathcal{L}_\phi$ of values $\sum_{i=1}^n \phi_i a_i$ from ϕ vector, where ϕ vector has $N_\phi(1)n$ 1s, $N_\phi(-1)n$ -1s and $N_\phi(0)n$ 0s. Here, we are not required the assumption in the Sect. 4.1 and $O(n)$ repetitions. The average size of the list $\mathcal{L}_\phi$ is

$$\mathcal{L}_\phi = \binom{n}{N_\phi(1)n, N_\phi(-1)n, N_\phi(0)n} \approx \frac{1}{n} \cdot 2^{h(N_\phi(1), N_\phi(-1))n}.$$

Next, from the list $\mathcal{L}_\phi$, the target sum $R_\phi^{(j)}$ and the modulus M_ϕ, we construct the list $\mathbb{L}_\phi^{(j)}$ of values $\sigma_\phi^{(j)}$ satisfying $\sigma_\phi^{(j)} \equiv R_\phi^{(j)} \pmod{M_\phi}$. The average size of the list $\mathbb{L}_\phi^{(j)}$ is $L_\phi = \mathcal{L}_\phi / M_\phi$.

The average running time of these steps is $O(n \cdot \mathcal{L}_\phi)$.

Steps at the Level $\boldsymbol{\omega}$. We explain steps at the level ω. In particular, we present how we construct list $\mathbb{K}_\omega^{(j)}$ from the lists $\mathbb{L}_\phi^{(2j-1)}$ and $\mathbb{L}_\phi^{(2j)}$ and list $\mathbb{L}_\omega^{(j)}$ from the list $\mathbb{K}_\omega^{(j)}$.

First, we denote the modulus at the level ω by M_ω which is coprime to M_ϕ and choose random integers $R_\omega^{(j)} (j = 1, \ldots, 7)$ and let $R_\omega^{(8)} = S - \sum_{j=1}^{7} R_\omega^{(j)}$. From the lists $\mathbb{L}_\phi^{(2j-1)}$ and $\mathbb{L}_\phi^{(2j)}$, the target sum $R_\omega^{(j)}$ and the modulus M_ω, we construct the list $\mathbb{K}_\omega^{(j)}$ by list algorithm. The list $\mathbb{K}_\omega^{(j)}$ is the list of values $\sigma_\omega^{(j)}$ satisfying $\sigma_\omega^{(j)} \equiv R_\omega^{(j)} \pmod{M_\omega}$. The average size of list $\mathbb{K}_\omega^{(j)}$ is $K_\omega = L_\phi^2 / M_\omega$.

Next, we obtain list $\mathbb{L}_\omega^{(j)}$ by removing some inconsistent solutions from the list $\mathbb{K}_\omega^{(j)}$. The average size of the list $\mathbb{L}_\omega^{(j)}$ is $L_\omega = \mathcal{L}_\omega / (M_\omega \cdot M_\phi)$ where

$$\mathcal{L}_\omega = \binom{n}{N_\omega(1)n, N_\omega(-1)n, N_\omega(0)n} \approx \frac{1}{n} \cdot 2^{h(N_\omega(1), N_\omega(-1))n}.$$

The average running time of these steps is $O(\max(n \cdot L_\phi, n \cdot K_\omega))$.

Steps after Level $\boldsymbol{\kappa}$. Remain steps are the same as BCJ algorithm's steps, but we need to transpose from M_ω to $M_\omega \cdot M_\phi$. Furthermore, our algorithm's L_ω is different from BCJ algorithm's one, so we need to reestimate L_ω^2 / M_κ. In our algorithm, L_ω^2 / M_κ is

$$\frac{L_\omega^2}{M_\kappa} = \frac{\mathcal{L}_\omega^2}{(M_\kappa \cdot M_\omega \cdot M_\phi) \cdot (M_\omega \cdot M_\phi)}.$$

5.2 Running Time of Our Algorithm

Overall running time of our algorithm is denoted by the function of n, α, β, γ and δ as:

$$\begin{aligned} T(n;\alpha,\beta,\gamma,\delta) = O((&\max(n\cdot\mathcal{L}_\phi, n\cdot L_\phi, n\cdot K_\omega, n\cdot L_\omega,\\ &n\cdot L_\kappa, n\cdot L_\kappa, n\cdot K_\nu, n\cdot L_\nu, n\cdot K_\epsilon))). \end{aligned} \tag{9}$$

When n is large enough, Eq. (9) is minimized with the values as:

$$\alpha = 0.022267, \beta = 0.013603, \gamma = 0.001285, \delta = 0.$$

With these parameters, the running time is

$$\begin{aligned} &T(n; 0.022267, 0.013603, 0.001285, 0)\\ &= O(\max(2^{0.2919n}, n^2\cdot 2^{0.2021n}, n^3\cdot 2^{0.2919n}, n^2\cdot 2^{0.2889n},\\ &\quad n^3\cdot 2^{0.2919n}, n^2\cdot 2^{0.2826n}, n^{2.5}\cdot 2^{0.2919n}, n^{1.5}\cdot 2^{0.2271n}, n^{0.5}\cdot 2^{0.2155n}))\\ &= O(n^3\cdot 2^{0.2919n}). \end{aligned}$$

Thus, the running time of our algorithm is $O(n^3\cdot 2^{0.2919n})$.

Note that since the list $\mathbb{K}_\chi$ need not to be stored, memory cost is:

$$O((\max(n\cdot\mathcal{L}_\phi, n\cdot L_\phi, n\cdot L_\omega, n\cdot L_\kappa, n\cdot L_\nu))).$$

5.3 Comparison between BCJ Algorithm and Our Algorithm

The discussions in Sect. 4 and Sect. 5 show that the running time of BCJ algorithm is $O(n^{3.5}\cdot 2^{0.2909n})$ and the running time of our algorithm is $O(n^3\cdot 2^{0.2919n})$. Instead of comparing $O(n^{3.5}\cdot 2^{0.2909n})$ with $O(n^3\cdot 2^{0.2919n})$, we compare $n^{0.5}$ with $2^{0.001n}$ as Fig. 1. Here we assume that constant terms of both functions are equal. When $n < 6312$, it holds $n^{0.5} > 2^{0.001n}$. So our algorithm can solve subset sum problem faster than BCJ algorithm when $n < 6312$.

6 Extensions in More Decompositions

In this paper, while we decompose a knapsack four times, we can decompose a knapsack more times and it may reduce the running time. When we decompose a knapsack k times, the evaluation function is denoted as:

$$T(n; a_1, a_2, \ldots, a_k) = O(n\cdot\max(\mathcal{L}_k, L_k, K_{k-1}, L_{k-1}, K_{k-2}, \ldots, L_1, K_0)).$$

First, we introduce the ration of occurrences of $x \in \{-1, 0, 1\}$ at the knapsack of i-th($i = 1, \ldots, k$) level as:

$$N_i(1) = \frac{1}{2^{i+1}} + \sum_{j=1}^{i}\frac{a_j}{2^{i-j}},\ N_i(-1) = \sum_{j=1}^{i}\frac{a_j}{2^{i-j}},\ N_i(0) = 1 - \frac{1}{2^{i+1}} - 2\sum_{j=1}^{i}\frac{a_j}{2^{i-j}},$$

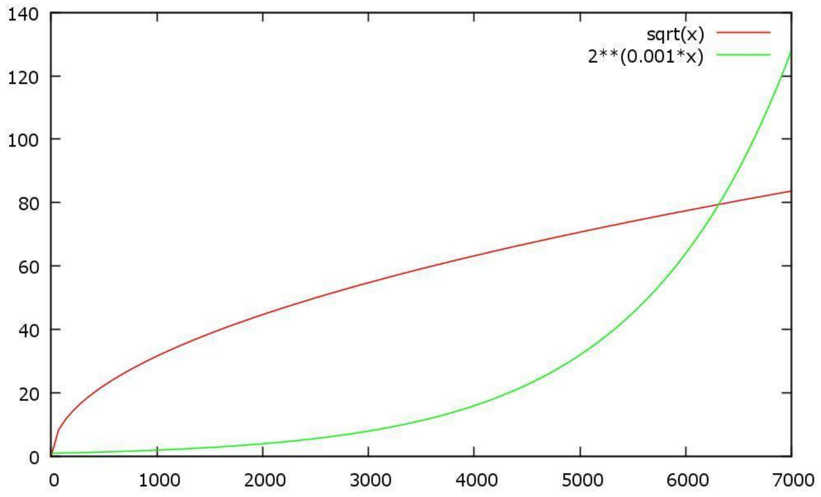


Fig. 1. Relation between $n^{0.5}$ and $2^{0.001n}$

where a_i is the parameter introduced at the i-th level and

$$N_0(1) = 1/2,\ N_0(-1) = 0,\ N_0(0) = 1/2.$$

Next, we denote modulus $M_i^{(k)}$ as:

$$M_i^{(k)} = \binom{N_{i-1}(1)n}{N_{i-1}(1)n/2} \cdot \binom{N_{i-1}(-1)n}{N_{i-1}(-1)n/2} \cdot \binom{N_{i-1}(0)n}{(N_i(1) - N_{i-1}(1)/2)n, (N_i(-1) - N_{i-1}(-1)/2)n, *},$$

when $i = 2, \ldots, k$ and define modulus $M_1^{(k)}$ as:

$$M_1^{(k)} = \binom{N_0(1)n}{N_0(1)n/2} \cdot \binom{N_0(0)n}{N_1(-1)n, N_1(-1)n, *}.$$

By using them, each function included in the evaluation function is denoted as:

$$\mathcal{L}_i = \binom{n}{N_i(1)n, N_i(-1)n, N_i(0)n},\ L_i = \frac{\mathcal{L}_i}{M_i^{(k)}},$$

$$K_i = \frac{\mathcal{L}_{i+1}^2}{M_i^{(k)} \cdot M_{i+1}^{(k)}}\ (i \neq k),\ K_0 = L_1^2 \cdot \frac{M_1^{(k)}}{2^n},$$

where $i = 1, \ldots, k$. However, an evaluation function has more than five parameters, so we have not optimized the evaluation function yet.

7 Concluding Remarks

In this paper, we analyze the running time of BCJ algorithm [1] by O notation and extend it. Our algorithm can solve subset sum problem faster than BCJ algorithm when $n < 6312$. We have not yet settled how to determine the best parameters of estimation function analytically. However, we believe that our parameters setting is optimal.

Acknowledgement. The second author was supported by KAKENHI 22700006.

References

1. Becker, A., Coron, J.S., Joux, A.: Improved Generic Algorithms for Hard Knapsacks. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 364–385. Springer, Heidelberg (2011)
2. Coster, M.J., Joux, A., LaMacchia, B.A., Odlyzko, A.M., Schnorr, C.P., Stern, J.: Improved Low-Density Subset Algorithms. Computational Complexity 2, 111–128 (1992)
3. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman (1979)
4. Howgrave-Graham, N., Joux, A.: New Generic Algorithms for Hard Knapsacks. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 235–256. Springer, Heidelberg (2010)
5. Lagarias, J.C., Odlyzko, A.M.: Solving Low-Density Subset Sum Problems. Journal of the Association for Computing Machineary 32(1), 229–246 (1985)
6. Merkle, R.C., Hellman, M.E.: Hiding Information and Signatures in Trapdoor Knapsacks. IEEE Transactions on Information Theory 24, 525–530 (1978)
7. Schroeppel, R., Shamir, A.: A $T = O(2^{n/2})$, $S = O(2^{n/4})$ Algorithm for Certain NP-Complete Problems. SIAM Journal on Computing 10(3), 456–464 (1981)

A List Algorithm

We introduce the algorithm which computes integers list $\mathbb{L}_R$. We denote the size of $\mathbb{L}_R$ by L_R.

Given two list of integers $\mathbb{L}_a$ and $\mathbb{L}_b$ of respective size L_a and L_b, together with a positive integer M and a non-negative integer R, the algorithm computes the list $\mathbb{L}_R$ such that:

$$\mathbb{L}_R = \{x + y \mid x \in \mathbb{L}_a, y \in \mathbb{L}_b \text{ s.t. } x + y \equiv R \pmod{M}\}.$$

In this paper, we call this algorithm *list algorithm*. This algorithm is introduced in [1] and its pseudo-code is given by Algorithm 1. The running time of Algorithm 1 is

$$O(\max(L_a \log L_a, L_b \log L_b, L_R)).$$

Moreover, assuming that the values of $\mathbb{L}_a$ and $\mathbb{L}_b$ modulo M are randomly distributed, the expected value of L_R is $L_a \cdot L_b/M$. In this paper's analysis, we assume that this assumption is guaranteed.

Using a slight variation of Algorithm 1, it is possible given $\mathbb{L}_a$ and $\mathbb{L}_b$ together with a target integer R to construct the set:

$$\mathbb{L}_R = \{x + y \mid x \in \mathbb{L}_a, y \in \mathbb{L}_b \text{ s.t. } x + y = R\}.$$

Algorithm 1. List Algorithm(Compute List $\mathbb{L}_R$)

```
Sort list $\mathbb{L}_a$ and $\mathbb{L}_b$(by increasing order of the values modulo $M$)
Let Target $\leftarrow R$
Let $i \leftarrow 0$ and $j \leftarrow L_b - 1$
while $i < L_a$ and $j \geq 0$ do
  Let Sum $\leftarrow (\mathbb{L}_a[i] \pmod{M}) + (\mathbb{L}_b[j] \pmod{M})$
  if Sum $<$ Target then
    Increment $i$
  end if
  if Sum $>$ Target then
    Decrement $j$
  end if
  if Sum $=$ Target then
    Let $i_0, i_1 \leftarrow i$
    while $i_1 < L_a$ and $\mathbb{L}_a[i_1] \equiv \mathbb{L}_a[i_0] \pmod{M}$ do
      Increment $i_1$
    end while
    Let $j_0, j_1 \leftarrow j$
    while $j_1 \geq 0$ and $\mathbb{L}_b[j_1] \equiv \mathbb{L}_b[j_0] \pmod{M}$ do
      Decrement $j_1$
    end while
    for $i \leftarrow i_0$ to $i_1 - 1$ do
      for $j \leftarrow j_1 + 1$ to $j_0$ do
        Append $\mathbb{L}_a[i] + \mathbb{L}_b[j]$ to $\mathbb{L}_R$
      end for
    end for
    Let $i \leftarrow i_1$ and $j \leftarrow j_1$
  end if
end while
Let Target $\leftarrow R + M$
Let $i \leftarrow 0$ and $j \leftarrow L_b - 1$
Repeat the above loop with the new target
```

Accelerating the Secure Distributed Computation of the Mean by a Chebyshev Expansion

Peter Lory and Manuel Liedel [⋆]

University of Regensburg, D-93040 Regensburg, Germany
{peter.lory,manuel.liedel}@wiwi.uni-regensburg.de

Abstract. Lindell and Pinkas (2002) have proposed the idea of using the techniques of secure multi-party computations to generate efficient algorithms for *privacy preserving data-mining*. In this context Kiltz, Leander, and Malone-Lee (2005) have presented a protocol for the secure distributed computation of the mean and related statistics in a two-party setting. Their protocol achieves constant round complexity. As a novel suggestion we use a Chebyshev expansion to accelerate this protocol. This approach considerably reduces the overhead of the protocol in terms of both computation and communication. The proposed technique can be applied to other protocols in the field of privacy preserving data-mining as well.

Keywords: Privacy preserving data mining, secure two-party computations, oblivious polynomial evaluation, Chebyshev expansion.

1 Introduction

Kiltz, Leander, and Malone-Lee [6] have presented a protocol for the secure computation of the mean and related statistics in a two-party setting. Each party owns a database with an attribute present in both databases. The protocol enables the two parties to evaluate the mean of these attribute values for the union of their databases. At the end of the protocol, the two parties learn the mean and nothing else without requiring a trusted party. The crucial point of the protocol is the computation of the quotient. For this purpose it employs a Taylor polynomial as an approximation of the geometric power series. This approximation can be made arbitrarily accurate by increasing the degree of this polynomial. However, under efficiency considerations such a simple truncation of the power series is not optimal. As a novel approach the present paper takes as point of departure a Chebyhev series $\sum_{\nu=0}^{\infty} T_\nu(x)$ where $T_\nu(x)$ denotes the Chebyshev polynomial of the first kind of degree ν. The paper heavily uses the fact that these expansions converge very rapidly under reasonable conditions (see Bulirsch and Stoer [2] or Mason and Handscomb [8]). Indeed, in the present case the coefficients of

[⋆] The authors are supported by the European Regional Development Fund – Europäischer Fonds für regionale Entwicklung (EFRE).

the Chebyshev expansion decrease much faster than the coefficients of the corresponding Taylor expansion. Consequently, we use a truncated Chebyshev series as the approximating polynomial. In this way an approximating polynomial can be constructed, which at equal accuracy requirements has a much lower degree than the truncated Taylor expansion. In fact, the degree can be reduced by a factor of approximately 0.43. Additionally, a novel approach is presented that merges three protocol steps of [6] into two. By these modifications considerable reductions in both the computation overhead and the communication overhead can be achieved. The security properties of the protocol are not affected because only publicly known parameters are used for the modifications.

The protocol of [6] repeatedly uses oblivious polynomial evaluation (OPE). Protocols for OPE are powerful cryptographic primitives for privacy preserving computations, which were first considered by Naor and Pinkas [9] and further elaborated by the same authors in [12]. As with oblivious transfer, OPE involves a sender and a receiver. The sender's input is a polynomial P of degree d over some finite field $\mathcal{F}$ and the receiver's input is an element $\alpha \in \mathcal{F}$. The degree d of P is public. The protocol is such that the receiver obtains $P(\alpha)$ without learning anything else about the polynomial P, and the sender learns nothing.

Notation. Let a be a real number. Then we denote by $\lceil a \rceil$ ($\lfloor a \rfloor$) the smallest (largest) integer b with $b \geq a$ ($b \leq a$), and by $\lceil a \rfloor$ the largest integer $b \leq a + 1/2$. The operator $\texttt{trunc}(a)$ rounds a towards 0, that is $\texttt{trunc}(a) = \lceil a \rceil$ if $a < 0$ and $\texttt{trunc}(a) = \lfloor a \rfloor$ if $a \geq 0$. Let p be a positive integer. All arithmetic modulo p is done centered around 0; that is $c \bmod p = c - \lceil c/p \rfloor p$. Much of the computation is going to occur in a suitably large finite field, named $\mathbb{F}_p$. The meaning of "suitably large" will be discussed in Section 5. Throughout the paper $\log a$ is the binary logarithm.

Outline. The paper is organized as follows: Section 2 presents the protocol of [6] for the privacy preserving computation of the mean. It is accelerated in Section 3 by using a Chebyshev expansion. This approach results in a new accelerated protocol (Section 4). Its overhead in terms of both computation and communication is studied in Section 5. The conclusion in Section 6 refers to other protocols in the field of privacy preserving data mining that benefit from the presented technique.

2 Privacy Preserving Computation of the Mean

Kiltz, Leander, and Malone-Lee [6] have presented a protocol for the secure computation of the mean and related statistics where the entries are shared by two players. Their protocol achieves constant round complexity (cf. Damgård et al. [4]) and enjoys a formal proof of security in the scenario of a semi-honest adversary. A semi-honest (or passive) adversary is an adversary that follows the instructions of the protocol but may try to use the information that he/she obtains during the execution of the protocol to learn something about the input of the other party. Using standard techniques the protocol can be made secure against an active adversary that attempts to deviate from the protocol.

For the reader's convenience, the protocol in [6] is briefly outlined here: Suppose that player P1 has n_1 entries in its database and P2 has n_2, which are denoted by $\{x_{1,1}, x_{1,2}, \ldots, x_{1,n_1}\}$ and $\{x_{2,1}, x_{2,2}, \ldots, x_{2,n_2}\}$ respectively. Let the sums x_1 and x_2 be defined as $x_1 = \sum_{i=1}^{n_1} x_{1,i}$ and $x_2 = \sum_{i=1}^{n_2} x_{2,i}$. Without loss of generality it can be assumed that x_1 and x_2 are integers. The task is the computation of the mean $M = (x_1 + x_2)/(n_1 + n_2)$ where P1 knows (x_1, n_1) and P2 knows (x_2, n_2). It is assumed that there are publicly known values N_1 and N_2 such that

$$-2^{N_1} \leq x_1 + x_2 \leq 2^{N_1} \tag{1}$$

and

$$0 < n_1 + n_2 < 2^{N_2 - 1}\,.^{1} \tag{2}$$

In the following a protocol is described for privately computing an approximation $\hat{M}$ of M. Let m_1 be the closest power of 2 to n_1. That is

$$2^{m_1 - 1} + 2^{m_1 - 2} \leq n_1 < 2^{m_1} + 2^{m_1 - 1}$$

and let m_2 be defined analogously. Let

$$k = \max\{m_1, m_2\} + 1\,. \tag{3}$$

As a consequence of (2), $0 \leq m_1, m_2 \leq N_2 - 1$ and

$$1 \leq k \leq N_2. \tag{4}$$

With the definition

$$\epsilon := 1 - (n_1 + n_2)/2^k \tag{5}$$

we have

$$n_1 + n_2 = 2^k(1 - \epsilon) \quad \text{where} \quad -\frac{1}{2} < \epsilon \leq \frac{5}{8}\,. \tag{6}$$

Consequently,

$$\frac{1}{n_1 + n_2} = \frac{1}{2^k(1 - \epsilon)} = \frac{1}{2^k}\left(\sum_{i=0}^{\infty} \epsilon^i\right) = \frac{1}{2^k}\left(\sum_{i=0}^{d} \epsilon^i\right) + \frac{1}{2^k} R_d \tag{7}$$

where

$$|R_d| \leq \frac{8}{3}\left(\frac{5}{8}\right)^{d+1}\,. \tag{8}$$

It follows that $2^{N_2 d}/(1 - \epsilon) = 2^{N_2 d + k}/(n_1 + n_2) = Z_d + 2^{N_2 d} R_d$ where Z_d is defined by

$$Z_d = \sum_{i=0}^{d} (2^{N_2}\epsilon)^i 2^{N_2(d-i)}\,. \tag{9}$$

Finally, some polynomials are defined. For $i = 1, 2, \ldots, N_2$ let $P_i(X)$ be the polynomial of degree $N_2 - 1$ such that for $X \in \{1, 2, \ldots, N_2\}$ $P_i(X) = 1$ for $X = i$ and $P_i(X) = 0$ otherwise.

The protocol is given in Fig. 1. It involves a series of sub-protocols from [6], which are repeated in Appendix A for the reader's convenience.

[1] The assumption $0 < n_1 + n_2 < 2^{N_2}$ in [6] does not guarantee that $2^{N_2}\epsilon$ in the definition of Z_d in (9) is an integer.

Set $d = \lceil \frac{3}{2}(t + N_1 + 2) \rceil$.

1. P1 and P2 input n_1 and n_2 respectively. Using Protocols 5 and 7 they compute additive shares of $2^{N_2}\epsilon$. With these shares as inputs P1 and P2 use Protocol 6 and calculate additive shares a_1^F, a_2^F of Z_d over $\mathbb{F}_p$.
2. P1 and P2 input a_1^F and a_2^F respectively. Using Protocol 4 they compute additive shares a_1^I, a_2^I of Z_d over $\mathbb{Z}$.
3. For $j = 1, \ldots, N_2$: [a] P1 computes $b_{1,j} = \lfloor a_1^I / 2^j \rfloor$; P2 computes $b_{2,j} = \lfloor a_2^I / 2^j \rfloor$.
4. Parties locally convert their shares back into additive $\mathbb{F}_p$-shares $c_{i,j}$, where share $c_{i,j}$ is the $\mathbb{F}_p$-equivalent of integer share $b_{i,j}$.
5. P1 and P2 input m_1 and m_2 respectively. Using Protocol 7 they compute additive shares d_1, d_2 of k over $\mathbb{F}_p$.
6. P1 chooses e_1 at random from $\mathbb{F}_p$ and defines the polynomial
$$R_1(X) = \sum_{i=1}^{N_2} c_{1,i} P_i(d_1 + X) - e_1 \,.$$
P1 runs an OPE protocol with P2 so that P2 learns $e_2 = R_1(d_2)$ and P1 learns nothing.
7. P2 chooses f_2 at random from $\mathbb{F}_p$ and defines the polynomial
$$R_2(X) = \sum_{i=1}^{N_2} c_{2,i} P_i(d_2 + X) - f_2 \,.$$
P2 runs an OPE protocol with P1 so that P1 learns $f_1 = R_2(d_1)$ and P2 learns nothing.
8. P1 and P2 input $e_1 + f_1$ and $e_2 + f_2$ respectively. Using Protocol 2 they compute multiplicative shares g_1, g_2 of $(e_1 + f_1) + (e_2 + f_2)$ over $\mathbb{F}_p$.
9. P1 inputs x_1 and P2 inputs x_2. Using Protocol 2 they compute multiplicative shares h_1, h_2 of $x_1 + x_2$ over $\mathbb{F}_p$.
10. P1 computes $\hat{M}_1 = g_1 h_1 \cdot 2^{-N_2 d}$ and sends it to P2.
11. P2 computes $\hat{M}_2 = g_2 h_2$ and sends it to P1.
12. P1 computes and outputs $\hat{M} = \hat{M}_1 \hat{M}_2$.
13. P2 computes and outputs $\hat{M} = \hat{M}_1 \hat{M}_2$.

[a] The range for j is determined by (4), cf. steps 6 and 7.

Fig. 1. The protocol of [6] to compute a private 2^{-t}-approximation $\hat{M}$ of $M = \frac{x_1+x_2}{n_1+n_2}$. For Protocols 2–7 see Appendix A. The size of the finite field $\mathbb{F}_p$ is a sufficiently large prime p (see Section 5).

3 Acceleration of the Computation of the Mean by a Chebyshev Expansion

The protocol of Fig. 1 relies on the fact that the integer Z_d with $d+1$ summands as defined in (9) approximates the quotient $2^{N_2 d+k}/(n_1+n_2)$. The present section applies a Chebyshev expansion in order to reduce the number of terms in the definition of Z_d. Let ϵ be defined as in Section 2 with the interval given in (6). By the substitution $\epsilon = (9x+1)/16$ this interval is mapped to the standard interval $-1 < x \leq 1$ and $1/(1-\epsilon)$ is transformed to

$$f(x) := \frac{16}{15} \cdot \frac{1}{1 - \frac{3x}{5}}, \tag{10}$$

which allows an analytic continuation to the function $f(z)$ with $z = x+iy$ on the ellipse with foci at $z = \pm 1$, major semi-axis $a = 5/3$ and eccentricity $e = 1/a$.

Theorem 1. *Let the function $f(z)$ be defined as above. Then its Chebyshev expansion is given by*

$$f(z) = \frac{8}{3}\left[\frac{1}{2} + \sum_{\nu=1}^{\infty}\left(\frac{1}{3}\right)^{\nu} T_{\nu}(z)\right]. \tag{11}$$

Proof. By the transformation

$$z = \frac{1}{2}\left(\zeta + \frac{1}{\zeta}\right) \tag{12}$$

the annulus $1/\rho < |\zeta| < \rho$ with $\rho := a + \sqrt{a^2-1} = 3$ in the ζ-plane is mapped to the above mentioned (doubly covered) ellipse in the z-plane and

$$f\left(\frac{1}{2}\left(\zeta + \frac{1}{\zeta}\right)\right) = \frac{16}{15} \cdot \frac{-10\zeta}{3\zeta^2 - 10\zeta + 3} = \frac{4}{3} \cdot \frac{1}{1 - \frac{\zeta}{3}} + \frac{4}{9} \cdot \frac{1}{\zeta} \cdot \frac{1}{1 - \frac{1}{3\zeta}}.$$

This yields the Laurent expansion

$$f\left(\frac{1}{2}\left(\zeta + \frac{1}{\zeta}\right)\right) = \frac{8}{3}\left[\frac{1}{2}\sum_{\nu=0}^{\infty}\left(\frac{1}{3}\right)^{\nu}\zeta^{\nu} + \frac{1}{6}\sum_{\nu=1}^{\infty}\left(\frac{1}{3}\right)^{\nu-1}\frac{1}{\zeta^{\nu}}\right]$$

in the above mentioned annulus, which proves (11) (see e.g. Bulirsch and Stoer [2] or Mason and Handscomb [8]). □

Using the result of Theorem 1 and the first equation in (7) we get

$$\frac{2^k}{n_1+n_2} = \frac{1}{1-\epsilon} = \frac{8}{3}\left[\frac{1}{2} + \sum_{\nu=1}^{\hat{d}}\left(\frac{1}{3}\right)^{\nu} T_{\nu}(x)\right] + \frac{8}{3}\hat{R}_{\hat{d}} \tag{13}$$

where

$$x = \frac{16\epsilon - 1}{9} \tag{14}$$

and

$$|\hat{R}_{\hat{d}}| \leq \frac{1}{2}\left(\frac{1}{3}\right)^{\hat{d}}. \tag{15}$$

The Chebyshev polynomial $T_\nu(x)$ is a polynomial of degree ν in the independent variable x (see again [2] or [8]):

$$T_\nu(x) = \sum_{j=0}^{\nu} t_{\nu,j}\, x^j. \tag{16}$$

The polynomials $T_\nu(x)$ are even or odd functions involving only even or odd powers of x, according as ν is even or odd. For $0 \leq \kappa \leq \lfloor \nu/2 \rfloor$ the coefficient $t_{\nu,\nu-2\kappa}$ of $x^{\nu-2\kappa}$ in (16) is given by

$$\begin{aligned} t_{\nu,\nu-2\kappa} &= (-1)^\kappa 2^{\nu-2\kappa-1}\left[2\binom{\nu-\kappa}{\kappa} - \binom{\nu-\kappa-1}{\kappa}\right] && (17)\\ &= (-1)^\kappa 2^{\nu-2\kappa-1}\frac{\nu}{\nu-\kappa}\binom{\nu-\kappa}{\kappa}, && (18)\end{aligned}$$

whereas for the coefficient $t_{\nu,\nu-2\kappa-1}$ of $x^{\nu-2\kappa-1}$

$$t_{\nu,\nu-2\kappa-1} = 0. \tag{19}$$

Obviously, the coefficients $t_{\nu,j}$ are integers. A straightforward application of formula (16) gives

$$\frac{1}{2} + \sum_{\nu=1}^{\hat{d}}\left(\frac{1}{3}\right)^\nu T_\nu(x) = \frac{1}{2} + \sum_{\nu=1}^{\hat{d}}\left(\frac{1}{3}\right)^\nu t_{\nu,0} + \sum_{j=1}^{\hat{d}}\left[\sum_{\nu=j}^{\hat{d}}\left(\frac{1}{3}\right)^\nu t_{\nu,j}\right]\cdot x^j. \tag{20}$$

In order to keep the computations in the integers, we multiply this equation by $2\cdot 3^{\hat{d}}$ and get

$$\left[\frac{1}{2} + \sum_{\nu=1}^{\hat{d}}\left(\frac{1}{3}\right)^\nu T_\nu(x)\right]\cdot 2\cdot 3^{\hat{d}} = \sum_{j=0}^{\hat{d}}\beta_j x^j \tag{21}$$

where the coefficients β_j are defined by

$$\beta_0 := 3^{\hat{d}} + 2\sum_{\nu=1}^{\hat{d}} 3^{\hat{d}-\nu} t_{\nu,0}, \tag{22}$$

$$\beta_j := 2\sum_{\nu=j}^{\hat{d}} 3^{\hat{d}-\nu} t_{\nu,j} \qquad (j = 1, 2, \ldots, \hat{d}). \tag{23}$$

For the same reason (14) is multiplied by $9\cdot 2^{N_2}$,

$$9\cdot 2^{N_2}x = 16\cdot 2^{N_2}\epsilon - 2^{N_2}. \tag{24}$$

It follows from (4) and the definition of ϵ in (5) that this quantity is an integer. Multiplying (21) by $9^{\hat{d}} \cdot 2^{N_2\hat{d}}$,

$$\left[\frac{1}{2}+\sum_{\nu=1}^{\hat{d}}\left(\frac{1}{3}\right)^{\nu} T_\nu(x)\right] \cdot 2 \cdot 3^{\hat{d}} \cdot 9^{\hat{d}} \cdot 2^{N_2\hat{d}} = \hat{Z}_{\hat{d}} \tag{25}$$

where

$$\hat{Z}_{\hat{d}} := \sum_{j=0}^{\hat{d}} \gamma_j \theta^j 2^{N_2(\hat{d}-j)} \tag{26}$$

with

$$\theta := 9 \cdot 2^{N_2} x \quad \text{and} \quad \gamma_j := \beta_j \cdot 9^{\hat{d}-j} \qquad (j = 0, 1, \ldots, \hat{d}). \tag{27}$$

Now the following theorem is an immediate consequence of (13) and (15).

Theorem 2. *Let n_1 and n_2 be integers that obey (2) and let k be defined by (3). Further, let $\hat{Z}_{\hat{d}}$ be defined by (26) and (27). Then the quotient $\hat{Z}_{\hat{d}}/(3^{3\hat{d}+1}2^{N_2\hat{d}-2})$ approximates $2^k/(n_1+n_2)$ such that*

$$\frac{2^k}{n_1+n_2} = \frac{\hat{Z}_{\hat{d}}}{3^{3\hat{d}+1}2^{N_2\hat{d}-2}} + \frac{8}{3}\hat{R}_{\hat{d}}, \quad \text{where} \quad |\hat{R}_{\hat{d}}| \le \frac{1}{2}\left(\frac{1}{3}\right)^{\hat{d}}. \tag{28}$$

Please note that only integers are involved in the computation of $\hat{Z}_{\hat{d}}$. The coefficients γ_j are defined in (27); for the β_j see (22) and (23); the $t_{\nu,j}$ are given in (17) – (19).

4 The New Protocol

A closer look at steps 5–7 of the of the protocol of Fig. 1 reveals that a further gain in efficiency is possible: Protocol 7 in step 5 returns additive shares of k. This involves a case distinction owing to the definition (3) of k, which motivates the definition of the polynomials Q_a and Q_b in [6]. This case distinction can be merged into steps 6 and 7. Please remember that $m_1, m_2 \in \{0, \ldots, N_2-1\}$, $m_1 - m_2 \in S := \{-(N_2-1), \ldots, (N_2-1)\}$ and $|S| = 2N_2 - 1$. Let the polynomials $\hat{Q}_a$ and $\hat{Q}_b$ of degree $|S|-1$ be defined by

$$\hat{Q}_a(s) = \begin{cases} \text{random values} & \text{for} \quad s \in S \setminus \{-(N_2-1)+m_1, \ldots, N_2-1\} \\ c_{1,1+m_1-s} & \text{for} \quad s \in \{-(N_2-1)+m_1, \ldots, -1\} \\ c_{1,m_1+1} & \text{for} \quad s \in \{0, 1, \ldots, N_2-1\} \end{cases}$$

and

$$\hat{Q}_b(s) = \begin{cases} \text{random values} & \text{for} \quad s \in S \setminus \{-(N_2-1)+m_2, \ldots, N_2-1\} \\ c_{2,1+m_2-s} & \text{for} \quad s \in \{-(N_2-1)+m_2, \ldots, -1, 0\} \\ c_{2,m_2+1} & \text{for} \quad s \in \{1, \ldots, N_2-1\} \end{cases}$$

Parameter $\hat{d}$, see Theorem 3; coefficients $\gamma_j, j = 0, 1, \ldots, \hat{d}$, see (27).

1. P1 and P2 input n_1 and n_2 respectively. Using Protocols 5 and 7 they compute additive shares of $2^{N_2}\epsilon$. Then each party locally computes its additive share of $\theta = 16 \cdot 2^{N_2}\epsilon - 2^{N_2}$. With these shares as inputs P1 and P2 use Protocol 6' and calculate additive shares a_1^F, a_2^F of $\hat{Z}_{\hat{d}}$ over $\mathbb{F}_p$.
2. P1 and P2 input a_1^F and a_2^F respectively. Using Protocol 4 they compute additive shares a_1^I, a_2^I of $\hat{Z}_{\hat{d}}$ over $\mathbb{Z}$.
3. Step 3 of Fig. 1.
4. Step 4 of Fig. 1.
5. This step is eliminated.
6. P1 chooses e_1 at random from $\mathbb{F}_p$ and defines the polynomial
$$R_a(X) = \hat{Q}_a(m_1 - X) - e_1 \,.$$
P1 runs an OPE protocol with P2 so that P2 learns $e_2 = R_a(m_2)$ and P1 learns nothing.
7. P2 chooses f_2 at random from $\mathbb{F}_p$ and defines the polynomial
$$R_b(X) = \hat{Q}_b(m_2 - X) - f_2 \,.$$
P2 runs an OPE protocol with P1 so that P1 learns $f_1 = R_b(m_1)$ and P2 learns nothing.
8. Step 8 of Fig. 1.
9. Step 9 of Fig. 1.
10. P1 computes $\hat{M}_1 = g_1 h_1 \cdot 2^{-N_2\hat{d}+2}$ and sends it to P2.
11. Step 11 of Fig. 1.
12. Step 12 of Fig. 1.
13. Step 13 of Fig. 1.

Fig. 2. The new protocol. For Protocols 2–7 see Appendix A .

It is easily verifiable that $\hat{Q}_a(m_1 - m_2) = c_{1,k}$ and $\hat{Q}_b(m_2 - m_1) = c_{2,k}$. This and the results of Section 3 are the basis of a new protocol, which is presented in Fig. 2.

As in [6] it is assumed, that $\mathbb{F}_p$ is sufficiently large that whenever a conversion from the integers to $\mathbb{F}_p$ is necessary it can be done without wrap-around (no modular reduction). The consequences for the selection of the value p are discussed in Section 5. It follows from Theorem 2 that after steps 1 and 2 of the new protocol players P1 and P2 hold a_1^I and a_2^I respectively such that

$$\left| \frac{2^{N_2\hat{d}-2+k}}{n_1 + n_2} - \frac{a_1^I + a_2^I}{3^{3\hat{d}+1}} \right| \le \frac{4}{3} \cdot 2^{N_2\hat{d}-2} \left(\frac{1}{3}\right)^{\hat{d}} .$$

The definition of k implies $k \ge 1$ (see (4)). Consequently, $b_{1,k}$ and $b_{2,k}$ after step 3 obey the inequality

$$\left| \frac{2^{N_2\hat{d}-2}}{n_1 + n_2} - \frac{b_{1,k} + b_{2,k}}{3^{3\hat{d}+1}} \right| < \frac{2}{3} \cdot 2^{N_2\hat{d}-2} \left(\frac{1}{3}\right)^{\hat{d}} + \frac{2}{3^{3\hat{d}+1}} .$$

When e_1, e_2, f_1 and f_2 are considered as integers it follows that

$$\left| \frac{2^{N_2\hat{d}-2}}{n_1+n_2} - \frac{(e_1+f_1)+(e_2+f_2)}{3^{3\hat{d}+1}} \right| < \frac{2}{3} \cdot 2^{N_2\hat{d}-2} \left(\frac{1}{3}\right)^{\hat{d}} + \frac{2}{3^{3\hat{d}+1}} .$$

Treating $\hat{M}$ as an integer we have because of (1)

$$\left| \frac{x_1+x_2}{n_1+n_2} \cdot 2^{N_2\hat{d}-2} - \frac{2^{N_2\hat{d}-2}}{3^{3\hat{d}+1}} \cdot \hat{M} \right| < \left[\frac{2}{3} \cdot 2^{N_2\hat{d}-2} \left(\frac{1}{3}\right)^{\hat{d}} + \frac{2}{3^{3\hat{d}+1}} \right] \cdot 2^{N_1} .$$

A division by $2^{N_2\hat{d}-2}$ gives

$$\left| \frac{x_1+x_2}{n_1+n_2} - \frac{\hat{M}}{3^{3\hat{d}+1}} \right| < \left[\frac{2}{3} \left(\frac{1}{3}\right)^{\hat{d}} + \frac{2}{3^{3\hat{d}+1} \cdot 2^{N_2\hat{d}-2}} \right] \cdot 2^{N_1} .$$

Inequality (2) implies $N_2 \geq 2$. Thus, the following theorem is proven:

Theorem 3. *Let $\hat{M}$ be the output from the protocol of Fig. 2 and let p be sufficiently large. Further, let $\hat{d}$ be chosen such that*

$$\frac{2}{3} \left(\frac{1}{3}\right)^{\hat{d}} + \frac{8}{3^{3\hat{d}+1} \cdot 2^{2\hat{d}}} \leq 2^{-(t+N_1)} , \tag{29}$$

then $\hat{M}/3^{3\hat{d}+1}$ is a 2^{-t}-approximation of the mean $M = \frac{x_1+x_2}{n_1+n_2}$.

Table 1 shows the degrees $\hat{d}$ of the polynomial (26) that satisfy (29) for a series of values of the accuracy parameter $t + N_1$. The entries in the table are computed with the help of the *GNU Multiple Precision Arithmetic Library* [5]. Table 1 also gives the corresponding values for the degree $d_{[6]}$. This is the degree of the Taylor polynomial (9) according to [6] that guarantees the same accuracy. The corresponding formula is given in the first line of Fig. 1. However, it would be unfair to compare this degree with $\hat{d}$ because in the proof of Theorem 3 sharper bounds are used than in [6]. Proceeding analogously to the proof of Theorem 3 results in the accuracy requirement

$$\frac{5}{6} \left(\frac{5}{8}\right)^{d} + 2^{1-2d} \leq 2^{-(t+N_1)} , \tag{30}$$

for the degree d of the Taylor polynomial (9). The fair degrees $d_{[6]}^{(\text{fair})}$ in Table 1 are calculated in this way.

The left hand sides of (29) and (30) are clearly dominated by $(2/3)(1/3)^{\hat{d}}$ and by $(5/6)(5/8)^d$ respectively. Equating these terms gives $\hat{d} \approx 0.43 \cdot d_{[6]}^{(\text{fair})} - 0.2$. Further neglecting the small additive term,

$$\boxed{\hat{d} \approx 0.43 \cdot d_{[6]}^{(\text{fair})} .} \tag{31}$$

Table 1. Degree $\hat{d}$ of (26) satisfying (29), degree $d_{[6]}$ according to [6] and fair degree $d_{[6]}^{(\mathrm{fair})}$ for various values of the accuracy parameter $t + N_1$ (cf. text in Section 4); $\log B$ with $B := \max_i |\beta_i|$. For the last two rows see text in Section 5.

$t + N_1$	8	16	24	32	40	48	64	80	104	128
Degree $\hat{d}$ of (26)	5	10	15	20	25	30	41	51	66	81
Degree $d_{[6]}$	15	27	39	51	63	75	99	123	159	195
Fair degree $d_{[6]}^{(\mathrm{fair})}$	12	24	36	47	59	71	94	118	153	189
$\hat{d}/d_{[6]}^{(\mathrm{fair})}$	0.42	0.42	0.42	0.43	0.42	0.42	0.44	0.43	0.43	0.43
$\log B$	7.61	15.53	23.45	31.38	39.30	47.23	64.66	80.51	104.29	128.06
$N_2^{(\mathrm{equal})}$ of (33)	3.43	3.48	3.48	3.60	3.57	3.54	3.74	3.67	3.65	3.60

This result is confirmed by the entries in the fifth row of Table 1 and demonstrates that considerable reductions in the computational efforts are possible by the use of Chebyshev expansions as presented in Section 3.

Security. The above described construction of the polynomial defining $\hat{Z}_{\hat{d}}$ uses only publicly known parameters. Therefore, the new protocol has the same security properties as the protocol in [6]. In particular, it enjoys a formal proof of privacy (for details and definitions see [6]).

5 Computation and Communication Overhead

The protocol of Fig. 2 clearly runs in a constant number of communication rounds between the two parties. The computation overhead of the protocol depends on the accuracy of the result. Naturally the degree $\hat{d}$ grows with increasing accuracy requirements (see Table 1).

Let us now consider the size of the field $\mathbb{F}_p$. For the protocol of Fig. 2 we have to choose the prime p sufficiently large so that no unwanted wrap-around (no modulo p reduction) can occur. It follows from (28) and (29) that the value $\hat{Z}_{\hat{d}}$ that is computed in the first step according to (26) is positive. Since $|x| \leq 1$, the following bound can be proven from the definitions in (26) and (27):

$$0 < \hat{Z}_{\hat{d}} \leq B \cdot \sum_{j=0}^{\hat{d}} 9^{\hat{d}-j} (9 \cdot 2^{N_2})^j 2^{N_2(\hat{d}-j)} = B \cdot (\hat{d}+1) \cdot 9^{\hat{d}} \cdot 2^{N_2 \hat{d}} \qquad (32)$$

where $B := \max_j |\beta_j|$. Consequently Protocol 4 in Appendix A (FTI protocol that converts additive shares over $\mathbb{F}_p$ into additive shares over the integers) requires a prime p with $\log p > N_2\hat{d} + \log B + \log(\hat{d}+1) + 2\hat{d}\log 3 + \rho + 6$ where ρ is a security parameter, typically chosen as $\rho = 80$. In steps 10 to 13 it has to be ensured that the value $g_1 h_1 g_2 h_2 = 2^{N_2\hat{d}-2}\hat{M}$ does not exceed p. Since $\hat{M} \approx 3^{3\hat{d}+1}(x_1+x_2)/(n_1+n_2)$, it can be concluded that $|g_1 h_1 g_2 h_2| \leq 2^{N_2\hat{d}-2} 3^{3\hat{d}+1} 2^{N_1}$ and the requirement.

$$\log p > N_2\hat{d} + 2\hat{d}\log 3 + \max\{\log B + \log(\hat{d}+1) + \rho + 6, (\hat{d}+1)\log 3 + N_1 - 2\}$$

is sufficient. By an improved FTI protocol (see [6]) this requirement can be weakened to $\log p > N_2\hat{d}+N_1-2+2\hat{d}\log 3+\max\{\log B+\log(\hat{d}+1), (\hat{d}+1)\log 3\}$. This has to be compared with the requirement $\log p_{[6]} > N_2 d + N_1$ in [6]. Because of (31) the bound for $\log p_{[6]}$ grows faster with increasing N_2 than our bound. Both bounds coincide for

$$N_2^{(\mathrm{equal})} = \frac{2\hat{d}\log 3 - 2 + \max\{\log B + \log(\hat{d}+1), (\hat{d}+1)\log 3\}}{d - \hat{d}} \tag{33}$$

These values are presented in the last row of Table 1 for $d = d_{[6]}^{(\mathrm{fair})}$. Please note that $N_2 = \lceil N_2^{(\mathrm{equal})} \rceil$ is the bound (2) and will exceed these values in almost all practical cases. Then the approach of the present paper allows smaller field sizes than [6].

The complexities of both the protocol of [6] (see Fig. 1) and the new protocol of Fig. 2 are clearly dominated by two operations: (1) Step 1 to compute shares of Z_d and $\hat{Z}_{\hat{d}}$ respectively. This is implemented by Protocol 7 (see Appendix A) using two invocations of an OPE with polynomials of degree $2N_2 - 2$ and Protocols 6 and 6' respectively (see again Appendix A) using an OPE with polynomials of degree d and $\hat{d}$ respectively. (2) Steps 5–7 in Fig. 1 and steps 6, 7 in Fig. 2 respectively.

Let us first consider (1), i.e.the effect of the reduction of the degree in step 1. The comparison (31) shows that at equal accuracy requirements the degree $\hat{d}$ of $\hat{Z}_{\hat{d}}$ in the new protocol is reduced by a factor of approximately 0.43. The consequences of such a reduction on the computation overhead of the OPE depend on the selected OPE protocol. Following the arguments in [12] we measure the computation overhead in terms of the number of invocations of a 1-out-of-2 oblivious transfer protocol (OT_1^2). This is justified because the overhead of this operation is greater by orders of magnitude than the overhead of the other operations. The OPE Protocol 3.2 in [12] runs a single invocation of a $(kd+1)$-out-of-$[(kd+1)\cdot m]$ oblivious transfer where k and m are security parameters and d is the degree of the polynomial. The OPE Protocol 3.3 in [12] runs $(kd + 1)$ invocations of a 1-out-of-m oblivious transfer. The oblivious transfer protocols of Naor and Pinkas [11] are optimized with respect to efficiency. If these protocols are used as subprotocols in the OPE Protocols 3.2 and 3.3 of [12], the number of invocations of OT_1^2 grows linearly with the degree d. Taking into account that an efficient OT_1^2 protocol (see [10]) requires a constant number of exponentiations we conclude that the computation complexity of Protocols 6 and 6' respectively is $O(d\log p)$ multiplications in the finite field $\mathbb{F}_p$. Thus, the computational overhead for the calculation of additive shares of $\hat{Z}_{\hat{d}}$ in the new protocol is approximately 43% of the overhead the corresponding operation in the protocol of [6]. Taking into account that the new protocol allows smaller field sizes (see above) the acceleration is even more pronounced. Using Horner's rule the OPE can be reduced to the parallel execution of an OPE of d linear polynomials where all the linear polynomials are evaluated at the same point. In this sense, the number of invocations of OT_1^2 required by Protocol 3.4 in [12] is independent of the degree. In any case the communication complexity is $O(d \log p)$.

Finally, let us study (2) i.e. the effects of merging steps 5–7 of the protocol of [6] (see Fig.1) into steps 6 and 7 of the new protocol of Fig. 2. The former protocol uses in its step 5 Protocol 7 (see Appendix A), which requires two invocations of an OPE with polynomials of degree $2N_2 - 2$. Each step 6 and 7 requires an OPE with a polynomial of degree $N_2 - 1$. In the new protocol step 5 is eliminated and each step 6 and 7 now requires an OPE with a polynomial of degree $2N_2 - 2$. Let us assume that Protocols 3.2 or 3.3 of [12] are used as OPE protocols. Then the number of invocations of OT_1^2 grows linearly with the degree (see above). Consequently, it is proportional to $6N_2 - 6$ for the protocol of [6], whereas it is proportional to $4N_2 - 4$ for the new protocol. This is a gain of approximately 30%. Protocol 3.4 of [12] requires the same number of invocations of OT_1^2 regardless of the degree of the polynomial. Therefore, in this case steps 5–7 of the new protocol require only 50% of the number of invocations of OT_1^2 that is needed for steps 5–7 of the protocol of [6]. As above the communication complexity is $O(d \log p)$ in any case, where d is again the degree of the polynomial and p the size of the finite field $\mathbb{F}_p$.

6 Conclusion

The two-party protocol of Kiltz, Leander, and Malone-Lee [6] for the secure computation of the mean has been accelerated in the present paper by replacing a Taylor approximation of $1/(1 - \epsilon)$ by a corresponding truncated Chebyshev series expansion. It has been demonstrated that this technique allows much lower degrees of the approximating polynomial. In this way it forms a sound basis for practical improvements. The method is useful not only for the protocol of [6] but for all protocols, where a power series is approximated by a Taylor polynomial. Obviously, the new technique can be applied to the distributed calculation of an additive sharing of the variance as described in [6] as well. Another example is the protocol for the private computation of the ID3 data mining algorithm of Lindell and Pinkas [7], which employs the power series for the logarithm function. A third application is oblivious neural learning (see Chang and Lu [3]), where the activation function $\phi(\epsilon) = a \tanh(b\epsilon)$ has to be approximated by low degree polynomials in a piecewise manner.

References

1. Algesheimer, J., Camenisch, J., Shoup, V.: Efficient Computation Modulo a Shared Secret with Application to the Generation of Shared Safe-Prime Products. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 417–432. Springer, Heidelberg (2002)

2. Bulirsch, R., Stoer, J.: Darstellung von Funktionen in Rechenautomaten. In: Sauer, R., Szabó, I. (eds.) Mathematische Hilfsmittel des Ingenieurs. Grundlehren der mathematischen Wissenschaften, vol. 141, pp. 352–446. Springer, Berlin (1968)
3. Chang, Y.-C., Lu, C.-J.: Oblivious Polynomial Evaluation and Oblivious Neural Learning. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 369–384. Springer, Heidelberg (2001)
4. Damgård, I., Fitzi, M., Kiltz, E., Nielsen, J.B., Toft, T.: Unconditionally Secure Constant-Rounds Multi-party Computation for Equality, Comparison, Bits and Exponentiation. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 285–304. Springer, Heidelberg (2006)
5. The GNU Multiple Precision Arithmetic Library, Edition 5.0.3 (2012), `http://gmplib.org`
6. Kiltz, E., Leander, G., Malone-Lee, J.: Secure Computation of the Mean and Related Statistics. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 283–302. Springer, Heidelberg (2005)
7. Lindell, Y., Pinkas, B.: Privacy preserving data mining. Journal of Cryptology 15, 177–206 (2002)
8. Mason, J.C., Handscomb, D.C.: Chebyshev Polynomials. Chapman & Hall/CRC, Boca Raton (2003)
9. Naor, M., Pinkas, B.: Oblivious transfer and polynomial evaluation. In: Vitter, J.S., Larmore, L., Leighton, T. (eds.) Proceedings of the 31st ACM Symposium on Theory of Computing (STOC 1999), pp. 245–254. ACM Press (1999)
10. Naor, M., Pinkas, B.: Efficient oblivious transfer protocols. In: Proceedings of the 12th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA 2001), pp. 448–457. Society for Industrial and Applied Mathematics (2001)
11. Naor, M., Pinkas, B.: Computationally secure oblivious transfer. Journal of Cryptology 18, 1–35 (2005)
12. Naor, M., Pinkas, B.: Oblivious polynomial evaluation. SIAM Journal on Computing 35(5), 1254–1281 (2006)

A Sub-protocols

The main protocol of [6] is presented in Fig. 1, the corresponding new main protocol in Fig. 2. Both use Protocols 2–7 of [6] as sub-protocols. This appendix summarizes them for the reader's convenience and gives the (slight) modifications that are necessary for the new protocol. Most of these protocols employ an oblivious polynomial evaluation protocol OPE (see e.g. Naor and Pinkas [12]).

The protocols of Figs. 1 and 2 use a series of **conversion protocols**: Protocols 2 (ATM) and 3 (MTA) of [6] can be used for converting additive to multiplicative shares over $\mathbb{F}_p$ and vice versa. Protocol 4 (FTI) of [6] converts additive shares from $\mathbb{F}_p$ to additive shares from the integers. It is taken from [1] (where it is called Protocol SQ2SI) and specialised to the two-party case. It assumes that the parties have shares (z_1^F, z_2^F) over $\mathbb{F}_p$ where

$$-2^{n-1} < z = (z_1^F + z_2^F) \mod p < 2^{n-1}$$

for some n and requires $p > 2^{\rho+n+5}$ where ρ is a security parameter.

By Protocol 5 of [6] the parties obtain additive shares of

$$2^{N_2}\epsilon = 2^{N_2} - (n_1 + n_2)2^{N_2-k}$$

over $\mathbb{F}_p$. Protocol 6 computes additive shares of Z_d over $\mathbb{F}_p$. It is used as a sub-protocol in step 1 of Fig. 1. The parties have already obtained additive shares a_1, a_2 of $2^{N_2}\epsilon$ by using Protocols 5 and 7.

Protocol 6. Sharing of Z_d.

---

1. $\mathsf{P1}$ chooses b_1 at random and defines the polynomial $P(X) = \sum_{i=0}^{d}(a_1 + X)^i 2^{N_2(d-i)} - b_1$.
2. $\mathsf{P1}$ runs OPE with $\mathsf{P2}$ so that $\mathsf{P2}$ learns $b_2 = P(a_2)$ and $\mathsf{P1}$ learns nothing.

At the end the parties hold additive shares b_1, b_2 of Z_d.

In step 1 of Fig. 2 a slightly different sub-protocol is used, which is presented here as Protocol 6'. The parties have already obtained additive shares of $2^{N_2}\epsilon$ by using Protocols 5 and 7 and each party has already locally computed its additive share a_1 and a_2 respectively of $\theta = 16 \cdot 2^{N_2}\epsilon - 2^{N_2}$. With these shares as inputs $\mathsf{P1}$ and $\mathsf{P2}$ use Protocol 6' and calculate additive shares of $\hat{Z}_{\hat{d}}$ over $\mathbb{F}_p$.

Protocol 6'. Sharing of $\hat{Z}_{\hat{d}}$.

---

1. $\mathsf{P1}$ chooses b_1 at random and defines the polynomial $P(X) = \sum_{i=0}^{\hat{d}} \gamma_i (a_1 + X)^i 2^{N_2(\hat{d}-i)} - b_1$.
2. $\mathsf{P1}$ runs OPE with $\mathsf{P2}$ so that $\mathsf{P2}$ learns $b_2 = P(a_2)$ and $\mathsf{P1}$ learns nothing.

At the end the parties hold additive shares b_1, b_2 of $\hat{Z}_{\hat{d}}$.

Protocol 5 requires a protocol for obtaining an additive sharing of 2^k over $\mathbb{F}_p$. Protocol 7 in [6] does this. At the end of this protocol the parties hold additive shares of 2^k. By a slight modification (see [6]) the same technique can be used for sharing k. The latter version is used in step 5 of Fig. 1, but it is not used in the new protocol of Fig. 2.

Security Analysis of the Lightweight Block Ciphers XTEA, LED and Piccolo

Takanori Isobe and Kyoji Shibutani

Sony Corporation
1-7-1 Konan, Minato-ku, Tokyo 108-0075, Japan
{Takanori.Isobe,Kyoji.Shibutani}@jp.sony.com

Abstract. In this paper, we investigate the security of the lightweight block ciphers against the meet-in-the-middle (MITM) attack. Since the MITM attack mainly exploits low key-dependency in a key expanding function, the block ciphers having a simple key expanding function are likely to be vulnerable to the MITM attack. On the other hand, such a simple key expanding function leads compact implementation, and thus is utilized in several lightweight block ciphers. However, the security of such lightweight block ciphers against the MITM attack has not been studied well so far. We apply the MITM attack to the ciphers, then give more accurate security analysis for them. Specifically, combining thorough analysis with new techniques, we present the MITM attacks on 29, 8, 16, 14 and 21 rounds of XTEA, LED-64, LED-128, Piccolo-80 and Piccolo-128, respectively. Consequently, it is demonstrated that the MITM attack is the most powerful attack in the single-key setting on those ciphers with respect to the number of attacked rounds. Moreover, we consider the possibility of applying the recent speed-up keysearch based on MITM attack to those ciphers.

Keywords: block cipher, lightweight, meet-in-the-middle attack, speed-up keysearch.

1 Introduction

Over the past years the demand for security in low resource devices such as RFID tags and sensor nodes has been dramatically increased. This motivates cryptographers to design a new cryptographic primitives aimed to such constrained devices, and thus several lightweight block ciphers have been proposed such as PRESENT [6], LED [13] and Piccolo [22]. In such ciphers, several newly developed design tricks are utilized in order to reduce the required gates. For example, a serially computable MDS matrix used in LED provides good diffusion but requires small gate areas. The key expanding function of KTANTAN [8] does not need to update the user provided secret key itself to generate sub-keys. This allows us to significantly reduce the required gates especially in the fixed-key mode, i.e., a secret key is hard-wired. This technique is utilized in several block ciphers such as XTEA [18], LED and Piccolo. We refer such design trick for a key expanding function as a direct-key expansion throughout this paper. More precisely, the direct-key expansion referred in this paper is defined that each subkey k_i can be represented by a secret key K and a round constant c_i as $k_i = KP_i(K) \oplus c_i$ or $k_i = KP_i(K) \boxplus c_i$, where KP_i denotes an arbitrary bit permutation.

While the direct-key expansion leads compact implementation peculiarly in the fixed-key mode, it sometimes causes the security problems due to its slow diffusion regarding sub-keys. For instance, it has been known that KASUMI [1], which consists of the direct-key expansion, is vulnerable to the related-key attacks [3,11]. Though the practical impact on a related-key attack depends on an application and required conditions for related-keys, the security for most of the recently proposed block ciphers having the direct-key expansion against the related-key attack is well analyzed by the designers such as KTANTAN, LED and Piccolo. Therefore these ciphers are expected to be secure against the related-key attack. In fact, for a well-designed direct-key expansion, it is relatively simple to show that there is no unexpected flaw regarding the related-key attack in the key expanding, in contrast to a complicated key expanding such as AES [12]. It is considered as one of the desirable design features of the direct-key expansion.

Meet-in-the-middle (MITM) attack was first introduced in [10]. Since the MITM attack usually exploits the slowness of the diffusion in a key expanding function, the ciphers having the direct-key expansion are likely to be vulnerable to it. In fact, KTANTAN, which is believed to be secure against the related-key attack, has been theoretically broken by the MITM attack [7]. However, the security of the other block ciphers consisting of the direct-key expansion against the MITM attack have not been well studied. Thus, while the direct-key expansion has several desirable features especially in implementation, it is less reliable compared to a complicated key expanding function due to lack of the thorough security analysis.

In this paper, we reconsider the security of lightweight block ciphers against the MITM attacks. Our target ciphers are XTEA, LED and Piccolo, which have the direct-key expansion. Combining recent advanced techniques with new techniques such as an equivalent transformation technique for SPN ciphers, we succeed in improving the numbers of attacked rounds for those ciphers by the MITM attack. Specifically, we present the MITM attacks on 29 (out of 64), 8 (out of 32), 16 (out of 48), 14 (out of 25) and 21 (out of 31) rounds of XTEA, LED-64, LED-128, Piccolo-80 and Piccolo-128, respectively. Note that all of our attacks are the best with respect to the number of attacked rounds[1] in the single-key setting in literature. This implies that the MITM attack is actually the most effective attack for block ciphers having the direct-key expansion, and its security analysis is necessary. Moreover, we consider the possibility of applying the recent speed-up keysearch based on the MITM attack, which are applied to the full AES [5] and the full IDEA [15], to our target ciphers.

[1] The MITM attacks on the 14- and 21-round Piccolo-80 and Piccolo-128 were introduced in [22]. However, the details of those attacks have not been published.

The rest of this paper is organized as follows. Brief introductions of our target ciphers and the MITM attack are given in Sections 2 and 3, respectively. Section 4 shows our MITM attacks on the target ciphers. We give discussions on the speed-up keysearch based on the MITM attack and other attacks in Section 5. Finally, we conclude in Section 6.

2 Target Ciphers

This section gives brief descriptions of our target ciphers including XTEA [18], LED [13] and Piccolo [22]. Note that all of our target ciphers consist of the direct-key expansion. In the descriptions, for each algorithm, we use the same notations used in their original papers.

2.1 Description of XTEA

XTEA is a 64-round Feistel cipher with 64-bit block and 128-bit key proposed by Needham and Wheeler in 1997 [18]. Let 64-bit state of the round i be $S^i = L^i|R^i$, where $L^i, R^i \in \{0,1\}^{32}$. Each round updates a state by using a 32-bit round key RK^i as

$$L^{i+1} = R^i, \quad R^{i+1} = L^i \boxplus_{(32)} ((\delta^i \boxplus_{(32)} RK^i) \oplus F(R^i)),$$

where $\boxplus_{(32)}$ denotes an addition modulo 2^{32} and δ^i represents the i-th round constant. The function F is defined as $F(x) = ((x \lll 4) \oplus (x \ggg 5)) \boxplus_{(32)} x$. Each round key RK^i is derived from a secret key $K = K_0|K_1|K_2|K_3$, $K_i \in \{0,1\}^{32}$ according to the predefined rule (see Table 1).

2.2 Description of LED

LED is an SPN-type block cipher supporting a 64-bit block and 64-bit to 128-bit keys proposed by Guo et al. in 2011 [13]. For the 64-bit key mode called LED-64, a 64-bit secret key K is XORed to the 64-bit state every 4 rounds. For the 128-bit key mode called LED-128, a 128-bit secret key K is divided into two 64-bit sub-keys K_1 and K_2, then K_1 and K_2 are alternatively XORed to the 64-bit state every 4 rounds. Each round function consists of `AddConstants`, `SubCells`, `ShiftRows` and `MixColumnsSerial` similar to the round function of AES. The numbers of rounds for LED-64 and LED-128 are 32 and 48, respectively.

2.3 Description of Piccolo

Piccolo is a 64-bit block cipher supporting 80 and 128-bit keys proposed by Shibutani et al. in 2011 [22]. Piccolo-80 and Piccolo-128 consist of 25 and 31 rounds of a variant of a generalized Feistel network, respectively. Let 64-bit input of the i-th round be $S^i = X_0^i|X_1^i|X_2^i|X_3^i$, $X_j^i \in \{0,1\}^{16}$. Then the $(i+1)$-th round input S^{i+1} is derived as follows:

$$S^{i+1} = RP(X_0^i|(X_1^i \oplus F(X_0^i) \oplus rk_{2i-2})|X_2^i|(X_3^i \oplus F(X_2^i) \oplus rk_{2i-1})),$$

Table 1. Key expanding functions of XTEA, Piccolo-80 and Piccolo-128

Key expanding function of XTEA

Round i (1 to 32)	1 to 8								9 to 16								17 to 24								25 to 32							
j of K_j for RK^i	0	3	1	2	2	1	3	0	0	0	1	3	2	2	3	1	0	0	1	0	2	3	3	2	0	1	1	1	2	0	3	3
Round i (33 to 64)	33 to 40								41 to 48								49 to 56								57 to 64							
j of K_j for RK^i	0	2	1	1	2	1	3	0	0	3	1	2	2	1	3	1	0	0	1	3	2	2	3	2	0	1	1	0	2	3	3	2

Key expanding function of Piccolo-80

Round i (1 to 16)	1		2		3		4		5		6		7		8		9		10		11		12		13		14		15		16	
j of k_j for $(rk_{2i-2} \vert rk_{2i-1})$	2	3	0	1	2	3	4	4	0	1	2	3	0	1	2	3	4	4	0	1	2	3	0	1	2	3	4	4	0	1	2	3
Round i (17 to 25)	17		18		19		20		21		22		23		24		25															
j of k_j for $(rk_{2i-2} \vert rk_{2i-1})$	0	1	2	3	4	4	0	1	2	3	0	1	2	3	4	4	0	1														

Key expanding function of Piccolo-128

Round i (1 to 16)	1		2		3		4		5		6		7		8		9		10		11		12		13		14		15		16	
j of k_j for $(rk_{2i-2} \vert rk_{2i-1})$	2	3	4	5	6	7	2	1	6	7	0	3	4	5	6	1	4	5	2	7	0	3	4	1	0	3	6	5	2	7	0	1
Round i (17 to 31)	17		18		19		20		21		22		23		24		25		26		27		28		29		30		31			
j of k_j for $(rk_{2i-2} \vert rk_{2i-1})$	2	7	4	3	6	5	2	1	6	5	0	7	4	3	6	1	4	3	2	5	0	7	4	1	0	7	6	3	2	5		

where F is a 16-bit F-function, RP is a 64-bit permutation, and rk_{2i-2} and rk_{2i-1} are round keys introduced into the i-th round. Each 16-bit round key rk_i is derived from an 80-bit secret key $K = k_0|k_1|...|k_4$ or a 128-bit secret key $K = k_0|k_1|...|k_7$, where $k_j \in \{0, 1\}^{16}$ according to the predefined manner (see Table 1).

3 Meet-in-the-Middle Attacks

In this section, we briefly review the MITM attack presented in [7] and several advanced techniques introduced in [23,2,19,5,15].

The MITM attack consists of the MITM stage and the key testing stage. The attacker first filters out some wrong keys and reduces the key space in the MITM stage, then exhaustively searches a correct key from the surviving key candidates in the key testing stage.

The MITM stage first divides an n-bit block cipher E with an ℓ-bit secret key K into two functions $\mathcal{F}_{(1)}$ and $\mathcal{F}_{(2)}$. K is grouped into three sets $\mathcal{K}_{(1)}$, $\mathcal{K}_{(2)}$ and $\mathcal{K}_{(3)}$, where $\mathcal{K}_{(1)}$ and $\mathcal{K}_{(2)}$ are used only in $\mathcal{F}_{(1)}$ and $\mathcal{F}_{(2)}$, respectively, and $\mathcal{K}_{(3)}$ denotes the other bits of K. Such $\mathcal{K}_{(1)}$ and $\mathcal{K}_{(2)}$ are called *neutral key bits* of $\mathcal{F}_{(2)}$ and $\mathcal{F}_{(1)}$, respectively. Then, we can compute $\mathcal{F}_{(1)}(P)$ and $\mathcal{F}^{-1}_{(2)}(C)$ independently by guessing each neutral key bit. If the guessed key value is correct, the equation $\mathcal{F}_{(1)}(P) = \mathcal{F}^{-1}_{(2)}(C)$ holds. Due to parallel guesses of $\mathcal{K}_{(1)}$ and $\mathcal{K}_{(2)}$, we can efficiently check whether the guessed key is the correct one. After this stage, we have $2^{\ell-n}(= 2^{|\mathcal{K}_{(1)}|+|\mathcal{K}_{(2)}|}/2^n \times 2^{|\mathcal{K}_{(3)}|})$ key candidates[2]. In the key

[2] The number of key candidates can be reduced by repeatedly performing the MITM stage with additional plaintext/ciphertext pairs.

testing stage, the attacker exhaustively searches a correct key from the surviving key candidates by using additional plaintext/ciphertext pairs. The required computations of the attack in total C_{comp} is estimated as

$$C_{comp} = \underbrace{2^{|\mathcal{K}_{(3)}|}(2^{|\mathcal{K}_{(1)}|} + 2^{|\mathcal{K}_{(2)}|})}_{\text{MITM stage}} + \underbrace{(2^{\ell-n} + 2^{\ell-2n} + 2^{\ell-3n} + ...)}_{\text{Key testing stage}}.$$

The number of the required plaintext/ciphertext pairs is $\lceil \ell/n \rceil$. The required memory is $\min(2^{|\mathcal{K}_{(1)}|}, 2^{|\mathcal{K}_{(2)}|})$ blocks, which is the cost of the table used in the MITM stage.

Here, we review several advanced techniques that enhance the MITM attack.

– **Partial Matching [2] :** When checking $\mathcal{F}_{(1)}(P) = \mathcal{F}_{(2)}^{-1}(C)$, it is not necessary to match all values of the state. By using only part of the state value for the matching, some key bits around the matching point can be omitted.
– **Splice and Cut [23,2] :** It regards the last and the first rounds of the block cipher as consecutive rounds assuming the chosen plaintext attack or the chosen ciphertext attack. This allows the attacker to freely choose two functions $\mathcal{F}_{(1)}$ and $\mathcal{F}_{(2)}$, though it generally requires more plaintext/ciphertext pairs than the attack without the splice and cut.
– **Initial Structure [19,5] :** It is used to skip some rounds around the starting point of the MITM attack, by exchanging two neutral key bits in those skipped round. The formal concept of the initial structure is called *biclique* [16,5].

Since the MITM attack exploits low key-dependency that large parts of the cipher are independent from some key bits, it is considered as one of the most powerful attacks for the direct-key expansion. Moreover, since the MITM attack does not require related-keys, i.e., it works in the single-key setting, its security analysis is considered to be more important than related-key attacks. These are the reasons why we focus on the MITM attack in this work.

4 Meet-in-the-Middle Attacks on Lightweight Block Ciphers

In this section, we apply the MITM attack to lightweight block ciphers XTEA, LED and Piccolo. The results on our MITM attacks and known single-key attacks are summarized in Table 2.

In the followings, $F[i,j]$ denotes the $(j-i+1)$ consecutive rounds starting from the i-th round, e.g., $F[3,6]$ represents 4 consecutive rounds starting from the 3rd round. $X_{i[j]}$ denotes the j-th bit of X_i, where $X_{i[0]}$ is the most significant bit. Also, $X_{i[0-4]}$ denotes the bits $X_{i[0]}$ to $X_{i[4]}$.

Table 2. Summary of the attacks in the single-key setting

algorithm	#rounds	attack/bound	#attacked rounds	time	data	memory [bytes]	reference
XTEA	64	MITM	23	2^{117}	18 KP	negligible	[21]
		IDA	23	$2^{105.6}$	2^{63} CP	2^{104}	[9]
		MITM	**29**	2^{124}	2^{45} CP	2^{4}	This paper
LED-64	32	diff./linear	8^{*1}	-	-	-	[13]
		MITM	**8**	2^{56}	2^{8} CP	2^{11}	This paper
LED-128	48	diff./linear	8^{*1}	-	-	-	[13]
		MITM	**16**	2^{112}	2^{16} CP	2^{19}	This paper
Piccolo-80	25	MITM	14	-	-	-	[22]
		MITM	**14**	2^{73}	2^{64} CP	2^{8}	This paper
Piccolo-128	31	MITM	21	-	-	-	[22]
		MITM	**21**	2^{121}	2^{64} CP	2^{9}	This paper

IDA: impossible differential attack, KP: known plaintext, CP: chosen plaintext
$*1$: there is no useful differentials or linear hulls over 8 rounds.

4.1 Security of XTEA against MITM Attack

The MITM attack on the 23-round reduced XTEA was proposed in [21]. In order to improve this attack, we develop the 12-round partial matching by thoroughly analyzing the diffusion property of the round functions of XTEA, especially the addition property regarding key differences. Moreover, we carefully choose the value of the starting point in each inner loop to make the splice and cut technique more effective. It enables us to save the amount of the required data in spite of the use of the splice and cut technique. By combining these techniques, we succeed in constructing a MITM attack on the 29-round reduced XTEA. Figure 1 shows an overview of the attack. In the figure, PM and IS denote *partial matching* and *initial structure*, respectively.

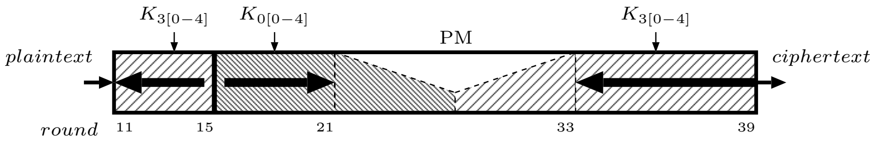


Fig. 1. Overview of the 29-round attack on XTEA

MITM Attack on 29-round XTEA. For the 29-round variant of XTEA starting from the round 11, using neutral key bits $\mathcal{K}_{(1)}(=K_{0[0-4]})$ and $\mathcal{K}_{(2)}$ $(=K_{3[0-4]})$, two chunks, $F[16,21]$ and $F^{-1}[34,39] \circ F^{-1}[11,15]$, can be computed independently. The detailed explanations of the partial matching for this attack and the attack complexity are given as follows.

– **Partial Matching.** Due to the differential property of the addition, we can compute $L^{27}_{[30-31]}$ and $R^{27}_{[30-31]}$ from S^{21} without using the value of $K_{3[0-4]}$ in

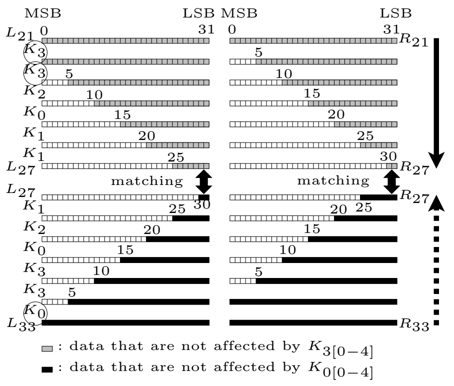

MSB
LSB
MSB
LSB
matching
matching
▭ : data that are not affected by $K_{3[0-4]}$
■ : data that are not affected by $K_{0[0-4]}$

Fig. 2. 12-round partial matching of XTEA

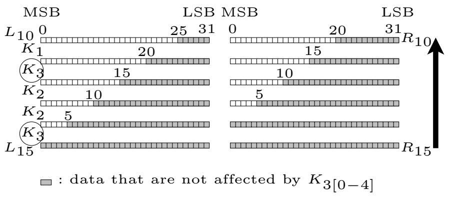


Fig. 3. $F^{-1}[11, 15]$ of XTEA

the forward process. On the other hand, $L^{27}_{[30-31]}$ and $R^{27}_{[30-31]}$ are calculated from S^{33} without using the value of $K_{0[0-4]}$ in the backward process as well. Thus, the 12-round partial matching works (see Fig. 2).

– **Evaluation.** The time complexity C_{comp} is estimated as

$$C_{comp} = \underbrace{2^{118}(2^5 + 2^5)}_{\text{MITM}} + \underbrace{2^{124}}_{\text{Key testing}} \approx 2^{124}.$$

The required memory is 2^4 byte ($= 2^5 \cdot 4$ bits). The number of the required plaintext/ciphertext pairs depends on the range of the output of $F^{-1}[11, 15]$. As mentioned in [25], the entire code book seems to be necessary, because $F^{-1}[11, 15]$ contains more key bits than the block size. However, by using the fact that neutral key bits $K_{3[0-4]}$ do not affect $L^{11}_{[25-31]}$ and $R^{11}_{[20-31]}$ as shown in Fig. 3, we can fix $L^{11}_{[25-31]}$ and $R^{11}_{[20-31]}$ in the MITM stage. To put it more precisely, we choose the start value as $S_{16} = F[11, 15](S_{11})$ in each value of $\mathcal{K}_{(3)}$, where $L^{11}_{[25-31]}$ and $R^{11}_{[20-31]}$ are set as arbitrary constant, and the other bits are free. Then, during the MITM stage, $L^{11}_{[25-31]}$ and $R^{11}_{[20-31]}$ are always fixed. Thus, the required data is estimated as $2^{45}(= 2^{64-19})$ chosen plaintext/ciphertext pairs.

4.2 Security of LED against MITM Attack

For SPN structures with the direct-key expansion, it seems hard to apply a MITM attack, since all of secret key bits are used in the small number of rounds. In fact, all of secret key bits of LED-64 are introduced in every 4 rounds. However, using *equivalent transformation technique*, we show MITM attacks on the 16-round and the 8-round reduced LED-128 and LED-64, respectively. Figures 4 and 5 show attack overviews for LED-128 and LED-64, respectively.

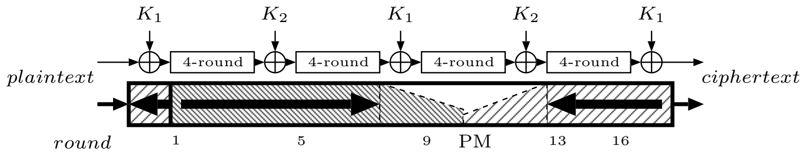


Fig. 4. Overview of the 16-round attack on LED-128

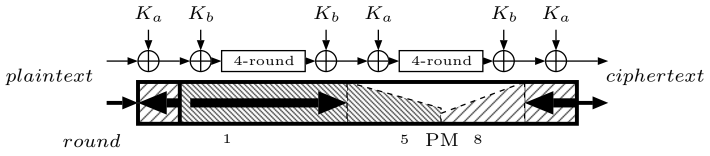


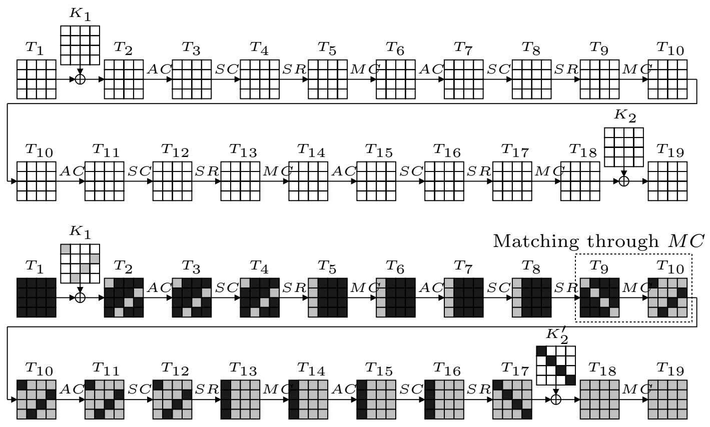
Fig. 5. Overview of the 8-round attack on LED-64

MITM Attack on 16-round LED-128. We consider the 16-round variant of LED-128 starting from the 1st round (i.e., $F[1, 16]$ of LED-128). Let FF and FB be 8 and 4 consecutive round functions starting from the round 1 and 13, respectively, i.e., $FF = F[1, 8]$ and $FB = F[13, 16]$ of LED-128. Also, FB contains both the final and the initial key additions by K_1 as shown in Fig. 4. Using neutral key bits K_1 and K_2, two chunks FB and FF can be computed independently. The partial matching used in this attack and the estimated attack complexity are explained as follows.

– **Partial Matching.** The 4-round partial matching is done at rounds 9 to 12 including the key additions before the 9-th round by K_1 and after the 12th round by K_2. $T_1, ..., T_{19}$ denote each state in this part as shown in Fig. 6, where each state consists of the 16 4-bit words arranged in a 4×4 square array. $T_i[j]$ denotes the j-th 4-bit word of T_i and $T_i[j, k]$ denotes $T_i[j]$ and $T_i[k]$, e,g., $T_i[3, 7, 11, 15]$ is four 4-bit words in the right most column of T_i. K_1 and K_2 are denoted in a similar way.

  In the straightforward method, it seems hard to construct the 4-round partial matching, since the 2 rounds of LED achieve the full diffusion. We introduce an *equivalent transformation technique* regarding the key addition by K_2. Since MC is a linear operation, T_{19} can be expressed as $T_{19} = MC(T_{17}) \oplus K_2 = MC(T_{17} \oplus K_2')$, where MC is an operation of `MixColumnsSerial` and $K_2' = MC^{-1}(K_2)$. This equation means that the operation of `MixColumnsSerial` and the key addition of K_2 are exchangeable by exploiting the linearity of `MixColumnsSerial`. In that case, the diffusion effect of the `MixColumnsSerial` regarding K_2 can be omitted in the backward computation. Since MC^{-1} is an invertible function, even if we directly control the value of K_2' instead of K_2, the MITM attack surely works in a similar manner.

  Besides, we utilize the match through MixColumns technique [20]. Suppose that $\mathcal{K}_{(1)}$ is $K_2'[0, 5, 10, 15]$ and $\mathcal{K}_{(2)}$ is $K_1[0, 7, 10, 13]$, then we can compute the values of $T_9[1, 2, 3, 4, 6, 7, 8, 9, 11, 12, 13, 14]$ without using $\mathcal{K}_{(2)}$ in the


K_1
T_1 T_2 T_3 T_4 T_5 T_6 T_7 T_8 T_9 T_{10}
AC SC SR MC
K_2
T_{10} T_{11} T_{12} T_{13} T_{14} T_{15} T_{16} T_{17} T_{18} T_{19}
Matching through MC
K'_2

▨ : data computed without using the neutral key bits of K'_2

■ : data computed without using the neutral key bits of K_1

Fig. 6. 4-round partial matching of LED-128

forward computation, and the values of $T_{10}[1, 2, 3, 4, 5, 6, 8, 9, 11, 12, 14, 15]$ without using $\mathcal{K}_{(1)}$ in the backward computation as shown in Fig. 6.
Then we do the matching between these values by using the match through `MixColumnsSerial`. In particular, we obtain the following equations with respect to the first columns of T_{10},

$$T_{10}[4] = (8 \cdot T_9[0]) \oplus (6 \cdot T_9[4]) \oplus (5 \cdot T_9[8]) \oplus (6 \cdot T_9[12]), \qquad (1)$$

$$T_{10}[8] = (B \cdot T_9[0]) \oplus (E \cdot T_9[4]) \oplus (A \cdot T_9[8]) \oplus (9 \cdot T_9[12]), \qquad (2)$$

$$T_{10}[12] = (2 \cdot T_9[0]) \oplus (2 \cdot T_9[4]) \oplus (F \cdot T_9[8]) \oplus (B \cdot T_9[12]). \qquad (3)$$

For Eqs. (1)-(3), only $T_9[0]$ is unknown values in the matching. We eliminate this values by combining with these equations. Then we obtain the two independent equations that do not include the variable $T_9[0]$. As an example, we give the following equation obtained from Eqs. (1) and (2),

$$\begin{aligned} B \cdot T_{10}[4] \oplus 8 \cdot T_{10}[8] = B \cdot ((6 \cdot T_9[4]) \oplus (5 \cdot T_9[8]) \oplus (6 \cdot T_9[12])) \\ \oplus 8 \cdot ((E \cdot T_9[4]) \oplus (A \cdot T_9[8]) \oplus (9 \cdot T_9[12])). \end{aligned}$$

Since each column has such two independent 4-bit equations, it is equivalent to 32 ($= 4 \times 2 \times 4$)-bit matching.

– **Evaluation.** The complexity of the attack C_{comp} is estimated as

$$C_{comp} = \underbrace{2^{96}(2^{16} + 2^{16})}_{\text{MITM}} + \underbrace{2^{96}}_{\text{Key testing}} \approx 2^{112}.$$

The required memory is 2^{19} byte ($\approx 2^{16} \cdot 6$ bytes). For the backward chunk, only 16 bits of the plaintext are affected by $K_1[0, 7, 10, 13]$. Thus, when the start state is properly chosen similar to the attack on XTEA, the number of required plaintext/ciphertext pairs is estimated as 2^{16}.

MITM Attack on 8-round LED-64. We consider the 8-round variant of LED-64 starting from the round 1. Let K_a and K_b be the 32-bit left and right two columns of K, respectively, i.e., $K_a = K[0,1,4,5,8,9,12,13]$ and $K_b = K[2,3,6,7,10,11,14,15]$. Then the 64-bit key XOR of K is divided into two 32-bit key XORs of K_a and K_b. FF denotes the functions from rounds 1 to 4 (i.e., $F[1,4]$ of LED-64) including key addition of K_b, and FB denotes the first and last key addition of K_a. By using K_a and K_b, two chunks FB and FF can be computed independently (see Fig. 5).

Suppose that $\mathcal{K}_{(1)}$ is $K[10,15] \in K_b$ and $\mathcal{K}_{(2)}$ is $K'[0,13] \in K_a$, where $K' = MC^{-1}(K)$. Then, the 4-round partial matching can be constructed similar to the partial matching for the attack on the reduced LED-128. The complexity of the attack C_{comp} is estimated as

$$C_{comp} = \underbrace{2^{48}(2^8 + 2^8)}_{\text{MITM}} + \underbrace{2^{32}}_{\text{Key testing}} \approx 2^{56}.$$

The required memory is 2^{11} bytes and the required data is 2^8 chosen plaintext/ciphertext pairs.

4.3 Security of Piccolo against MITM Attack

We show MITM attacks on the 21-round and the 14-round reduced Piccolo-128 and Piccolo-80, respectively.

MITM Attack on 21-Round Piccolo-128. For the 21-round variant of Piccolo-128 starting from the round 2, using neutral key bits $\mathcal{K}_{(1)}(= k_{3[8-15]})$ and $\mathcal{K}_{(2)}(= k_{6[0-7]})$, two chunks $F[9,13]$ and $F^{-1}[19,22] \circ F^{-1}[2,5]$ can be computed independently (see Fig. 7). We give detailed explanations for the partial matching and the initial structure used in this attack, and the estimated attack complexity as follows.

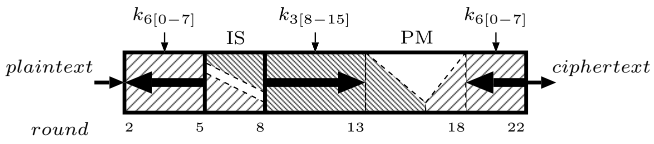


Fig. 7. Overview of the 21-round attack on Piccolo-128

– **Partial Matching.** For $F[14,18]$, in the forward process, X_1^{17} is obtained from S^{14} without using the value of $k_{6[0-7]}$. On the other hand, X_1^{17} is also computed from S^{19} without using the value of $k_{3[8-15]}$ in the backward process. Thus, the 5-round partial matching works (see Fig. 8).

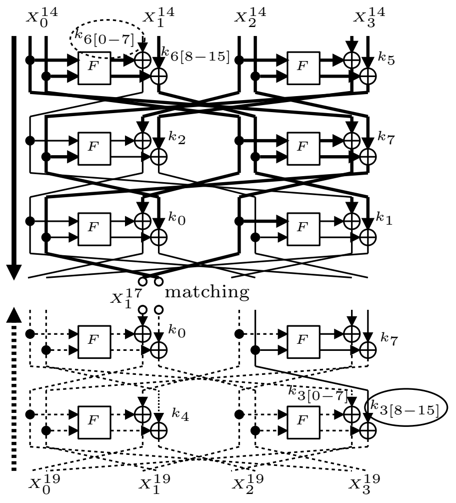


Fig. 8. 5-round PM of Piccolo-128

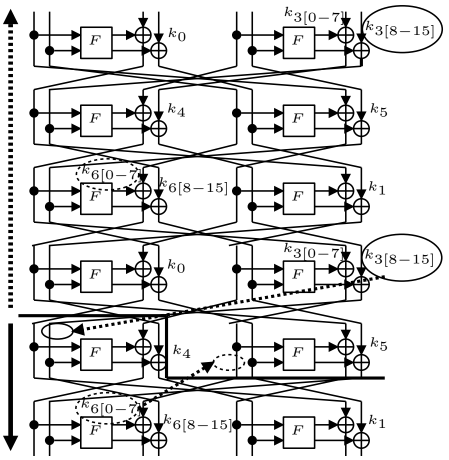


Fig. 9. 3-round IS of Piccolo-128

– **Initial Structure.** For $F[6, 8]$, we aim to exchange the positions of $k_{6[0-7]}$ for $k_{3[8-15]}$. Here, differences of $k_{6[0-7]}$ in the forward process and $k_{3[8-15]}$ in the backward process are independent in $F[6, 8]$. It means that one differential trail does not affect the others, since these trails do not share any non-linear elements. Thus, these values are exchangeable by splitting $F[6, 8]$ as illustrated in Fig. 9. This can be seen as the 3-round bicliques, and finding independent differentials directly leads the construction of the initial structure as mentioned in [5].

– **Evaluation.** The complexity of the attack on the 21-round reduced Piccolo-128 C_{comp} is estimated as

$$C_{comp} = \underbrace{2^{112}(2^8 + 2^8)}_{\text{MITM}} + \underbrace{2^{112}}_{\text{Key testing}} \approx 2^{121}.$$

The required memory is 2^9 bytes ($= 2^8 \cdot 2$ bytes), which is the cost of the table used in the MITM stage. The number of required plaintext/ciphertext pairs depends on the range of $F^{-1}[2, 8]$, because $k_{6[0-7]}$ is included in the initial structure. Due to the 4-round full diffusion property, the required data is 2^{64} plaintext/ciphertext pairs, which is called a code book attack. In case the code book attack is not allowed, the number of attacked rounds might be decreased.

MITM Attack on 14-Round Piccolo-80. For the MITM attack on the 14-round variant of Piccolo-80 starting from the round 5, using neutral key bits $\mathcal{K}_{(1)}(= k_{0[0-7]})$ and $\mathcal{K}_{(2)}(= k_{4[8-15]})$, two chunks $F[5, 8] \circ F[15, 18]$ and $F^{-1}[13, 14]$ can be computed independently. For $F[9, 12]$, the 4-round partial

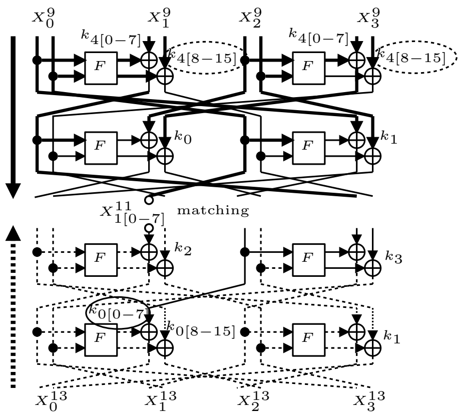


Fig. 10. 4-round partial matching of Piccolo-80

matching is constructed as illustrated in Fig. 10. The complexity of the attack on the 14-round reduced Piccolo-80 C_{comp} is estimated as

$$C_{comp} = \underbrace{2^{64}(2^8 + 2^8)}_{\text{MITM}} + \underbrace{2^{72}}_{\text{Key testing}} \approx 2^{73}.$$

The required memory is 2^8 bytes ($= 2^8 \cdot 1$ bytes), which is the cost of the table used in the MITM stage. The number of required plaintext/ciphertext pairs depends on the range of $F[15, 18]$, i.e., size of the possible output values of $F[15, 18]$ during the MITM stage. In general, if neutral key bits $k_{0[0-7]}$ does not affect the all bits of the output of $F[15, 18]$, it is possible to avoid code book attack by fixing some bits of the output similar to the AES attack [5]. However, $k_{0[0-7]}$ affects all bits of the output due to the 4-round full diffusion property. Thus, the required data is 2^{64} plaintext/ciphertext pairs. In our search, such 14-round attack without relying on the code book cannot be found. Thus, we expect that the number of attacked rounds may be slightly reduced in the non-code book attack on the reduced Piccolo-80.

5 Discussion

This section discusses the security of lightweight block ciphers against the speed-up keysearch based on the MITM attack. Then we compare our results with the previous attack results on our target ciphers.

5.1 Security against Speed-up Keysearch Based on MITM Attack

The speed-up keysearch based on the MITM attack is the recently proposed novel technique, which is known as the first attack for the full AES [5]. It makes use of *matching with precomputation* in conjunction with a *biclique* which is a formal concept of the initial structure. In general, it works slightly faster than the exhaustive key search, i.e., by a factor of 2 to 5, since it essentially tests all possible keys. This cryptanalysis is naturally applicable to lightweight block ciphers. Indeed, it has been applied to HIGHT and KATAN [14,17], whose attacks are slightly faster than that of the exhaustive key search. As mentioned in [4], for many block ciphers, one also can speed up the exhaustive keysearch with simple techniques, e.g., *distributive technique* and *early abort technique.* Therefore, it is debatable whether such a marginal improvement regarding the time complexity should be considered as a real attack in terms of exploiting algorithmic weaknesses. However, it is still meaningful from the view of the security evaluation of such ciphers. In the following, we roughly estimate the required time complexity for the speed-up keysearch of the lightweight block ciphers, XTEA, LED and Piccolo, assuming that the code book is allowed to use and the rounds of the partial matching and the initial structure can be omitted, i.e., the cost of the recomputation is regarded as zero in these rounds. Due to such strong assumptions, our estimations may not be accurate but provide indications of the security evaluations of the speed-up keysearch based on the MITM attack.

For XTEA, assuming the 12-round partial matching and two 6-round independent chunks, the time complexity for the speed-up keysearch of the full XTEA is estimated as $2^{127.32} (= 2^{128} \times ((64 - 12 - 12)/64))$. For LED-64/128, the 4-round partial matching is constructed. Exploiting the two 4-round independent chunks, the time complexities for the speed-up keysearch of the full LED-64/128 are roughly estimated as $2^{127.59} (= 2^{128} \times ((48 - 4 - 8)/48))$ and $2^{79.32} (= 2^{80} \times ((32 - 4 - 8)/32))$, respectively. For Piccolo-80/128, the 3-round initial structure and the 5-round partial matching are constructed as described in Section 4.3. Assuming the two 5-round independent chunks, the time complexity for the speed-up keysearch of the full Piccolo-128 is roughly estimated as $2^{126.74} (= 2^{128} \times ((31 - 3 - 5 - 10)/31))$. In a similar way, the time complexity of the full Piccolo-80 is roughly estimated as $2^{78.53} (= 2^{80} \times ((25 - 3 - 5 - 8)/25))$.

We emphasize that the above observations are rough estimations of the speed-up keysearch based on the MITM attack. Thus, the time complexities seem to be improved by analyzing deeply[3]. However, since these are marginal improvements in time complexity compared to the exhaustive key search, we do not claim them to be real attacks based on algorithmic weaknesses.

5.2 Comparison with Other Cryptanalysis Results

We compare our MITM attacks to the previously proposed attacks for each cipher. Note that we focus only on the single-key setting, which is the practical setting of fields in where lightweight cipher are needed, e.g., RFID tags and sensor nodes.

For XTEA, the previously best attacks regarding the number of the attacked rounds are the impossible differential attack and the MITM attack on the 23-

[3] This types of the speed-up keysearch on the reduced Piccolo-80/128 has been estimated in [24].

round variants [21,9][4]. Thus, our MITM attack, which is the 29-round attack, greatly improves their attacks with respect to the number of attacked rounds.

For LED-64/128 and Piccolo-80/128, there exist only designers' self evaluations, i.e., no external cryptanalysis has been published so far. According to [13], the maximum differential probability and the best linear hull probability of 4 rounds of LED-64/128 are both upper bounded by 2^{-32}. Thus, it was implicitly claimed that there is no useful differentials or linear hulls over 8 rounds of LED-64/128 in the single-key setting, though the actual attack has not been presented so far. By using the equivalent transformation technique, we showed the MITM attacks on the 8-round LED-64 and the 16-round LED-128. While the designers showed chosen key distinguishers for the 15-round LED-64 and the 27-round LED-128, our MITM attacks can be regarded as the best attacks on the reduced LED in the single-key setting.

For Piccolo-80/128, according to [22], the 9-round Piccolo-80/128 have a sufficient security level against differential and the linear cryptanalysis by the evaluations based on the number of the active F-functions. On the other hand, our MITM attacks work on the 14-round Piccolo-80 and the 21-round Piccolo-128, though both attacks require full plaintext/ciphertext pairs, which are called code book attacks. These results imply that MITM-type attacks are more effective than differential and linear cryptanalysis for both LED-64/128 and Piccolo-80/128 in the single-key setting, because the number of rounds to be attacked by the MITM attack is much larger than those of differential and linear cryptanalysis.

Therefore it is demonstrated that MITM attacks are the most powerful attacks on these lightweight ciphers in the single-key setting.

6 Conclusion

We have analyzed the security of several lightweight block ciphers that have the direct-key expansion against the meet-in-the-middle attack. While a simple key expanding function like the direct-key expansion leads compact implementation, its security analysis has not been studied well so far. On the other hand, since MITM attack mainly exploits low key-dependency that large parts of the cipher are independent from some key bits, a block cipher having the direct-key expansion is likely to be vulnerable to MITM attack. Moreover, since the MITM attack does not require related-keys, i.e., it works in the single-key setting, its security analysis is considered to be more important than related-key attacks.

In this paper, we have shown new MITM attacks on several lightweight block ciphers that have the direct-key expansion. Combined with new techniques such as equivalent transformation technique, we have presented the MITM attacks on 29 (out of 64), 8 (out of 32), 16 (out of 48), 14 (out of 25) and 21 (out of 31) rounds of XTEA, LED-64, LED-128, Piccolo-80 and Piccolo-128, respectively. Note that all of our attacks presented in this paper are the best or the first

[4] Recently, the zero correlation linear attack on the 27-round reduced XTEA was proposed in [26].

attacks in literature. For instance, the attack on the 29-round reduced XTEA is the best attack with respect to the number of attacked rounds. These results imply that MITM attack is actually effective for lightweight block ciphers. Thus, its security analysis is crucial, even if the cipher is expected to be secure against the other attacks.

Acknowledgments. We would like to thank to the anonymous referees for their insightful comments and suggestions.

References

1. 3rd Generation Partnership Project, Technical Specification Group Services and System Aspects, 3G Security, Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 2: KASUMI Specification, V3.1.1
2. Aoki, K., Sasaki, Y.: Preimage Attacks on One-Block MD4, 63-Step MD5 and More. In: Avanzi, R.M., Keliher, L., Sica, F. (eds.) SAC 2008. LNCS, vol. 5381, pp. 103–119. Springer, Heidelberg (2009)
3. Biham, E., Dunkelman, O., Keller, N.: A Related-Key Rectangle Attack on the Full KASUMI. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 443–461. Springer, Heidelberg (2005)
4. Biham, E., Dunkelman, O., Keller, N., Shamir, A.: New data-efficient attacks on reduced-round IDEA. IACR Cryptology ePrint Archive, vol. 2011, p. 417 (2011)
5. Bogdanov, A., Khovratovich, D., Rechberger, C.: Biclique Cryptanalysis of the Full AES. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 344–371. Springer, Heidelberg (2011)
6. Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C.: PRESENT: An Ultra-Lightweight Block Cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007)
7. Bogdanov, A., Rechberger, C.: A 3-Subset Meet-in-the-Middle Attack: Cryptanalysis of the Lightweight Block Cipher KTANTAN. In: Biryukov, A., Gong, G., Stinson, D.R. (eds.) SAC 2010. LNCS, vol. 6544, pp. 229–240. Springer, Heidelberg (2011)
8. De Cannière, C., Dunkelman, O., Knežević, M.: KATAN and KTANTAN — A Family of Small and Efficient Hardware-Oriented Block Ciphers. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 272–288. Springer, Heidelberg (2009)
9. Chen, J., Wang, M., Preneel, B.: Impossible differential cryptanalysis of the lightweight block ciphers TEA, XTEA and HIGHT. IACR Cryptology ePrint Archive, vol. 2011, p. 616 (2011)
10. Diffie, W., Hellman, M.E.: Exhaustive cryptanalysis of the NBS Data Encryption Standard. IEEE Computer 10, 74–84 (1977)
11. Dunkelman, O., Keller, N., Shamir, A.: A Practical-Time Related-Key Attack on the KASUMI Cryptosystem Used in GSM and 3G Telephony. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 393–410. Springer, Heidelberg (2010)
12. FIPS, Advanced Encryption Standard (AES). Federal Information Processing Standards Publication 197

13. Guo, J., Peyrin, T., Poschmann, A., Robshaw, M.: The LED Block Cipher. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 326–341. Springer, Heidelberg (2011)
14. Hong, D., Koo, B., Kwon, D.: Biclique attack on the full HIGHT. In: ICISC 2011(2011) (to appear)
15. Khovratovich, D., Leurent, G., Rechberger, C.: Narrow-Bicliques: Cryptanalysis of Full IDEA. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 392–410. Springer, Heidelberg (2012)
16. Khovratovich, D., Rechberger, C., Savelieva, A.: Bicliques for preimages: Attacks on Skein-512 and the SHA-2 family. In: FSE 2012 (to appear, 2012)
17. Knellwolf, S.: Meet-in-the-middle cryptanalysis of KATAN. In: Proceedings of the ECRYPT Workshop on Lightweight Cryptography (2011)
18. Needham, R.M., Wheeler, D.J.: Tea extensions. Techniacl report, Computer Laboratory, University of Cambridge (October 1997), http://www.cix.co.uk/~klockstone/xtea.pdf
19. Sasaki, Y., Aoki, K.: Finding Preimages in Full MD5 Faster Than Exhaustive Search. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 134–152. Springer, Heidelberg (2009)
20. Sasaki, Y.: Meet-in-the-Middle Preimage Attacks on AES Hashing Modes and an Application to Whirlpool. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 378–396. Springer, Heidelberg (2011)
21. Sekar, G., Mouha, N., Velichkov, V., Preneel, B.: Meet-in-the-Middle Attacks on Reduced-Round XTEA. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 250–267. Springer, Heidelberg (2011)
22. Shibutani, K., Isobe, T., Hiwatari, H., Mitsuda, A., Akishita, T., Shirai, T.: *Piccolo*: An Ultra-Lightweight Blockcipher. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 342–357. Springer, Heidelberg (2011)
23. van Oorschot, P.C., Wiener, M.: A Known-Plaintext Attack on Two-Key Triple Encryption. In: Damgård, I.B. (ed.) EUROCRYPT 1990. LNCS, vol. 473, pp. 318–325. Springer, Heidelberg (1991)
24. Wang, Y., Wu, W., Yu, X.: Biclique Cryptanalysis of Reduced-Round Piccolo Block Cipher. In: Ryan, M.D., Smyth, B., Wang, G. (eds.) ISPEC 2012. LNCS, vol. 7232, pp. 337–352. Springer, Heidelberg (2012)
25. Wei, L., Rechberger, C., Guo, J., Wu, H., Wang, H., Ling, S.: Improved Meet-in-the-Middle Cryptanalysis of KTANTAN (Poster). In: Parampalli, U., Hawkes, P. (eds.) ACISP 2011. LNCS, vol. 6812, pp. 433–438. Springer, Heidelberg (2011)
26. Bogdanov, A., Wang, M.: Zero Correlation Linear Cryptanalysis with Reduced Data Complexity. In: FSE 2012 (to appear, 2012)

Improved Known-Key Distinguishers on Feistel-SP Ciphers and Application to Camellia

Yu Sasaki[1], Sareh Emami[2], Deukjo Hong[3], and Ashish Kumar[4]

[1] NTT Corporation, Japan
sasaki.yu@lab.ntt.co.jp
[2] Macquarie University, Australia
sareh.emami@mq.edu.au
[3] The Attached Institute of ETRI, Korea
hongdj@ensec.re.kr
[4] Indian Institute of Technology Kharagpur, India
kumarashish.iitkgp@gmail.com

Abstract. This paper revisits previous known-key distinguishers on generic Feistel-SP ciphers based on rebound attacks. In this paper first we propose a new 5-round inbound phase that requires 2^c computations, while the previous work requires 2^{2c} computations (c is a size of the S-box). The new method also improves the number of rounds which can be attacked. Then, we apply the new procedure to Camellia. After several optimizations for Camellia, it is shown that collisions are efficiently generated against 9 rounds out of 18 rounds of Camellia-128 including FL and whitening layers in the compression function modes such as MMO and Miyaguchi-Preneel modes. The attack on Camellia is verified by a machine experiment and the generated results are presented in the paper.

Keywords: block cipher, Feistel-SP, Camellia, known-key, rebound.

1 Introduction

Block-ciphers are often used as building blocks of secret-less primitives such as hash functions, hence recently cryptographers have started to evaluate the security of block-ciphers as hash functions. Known-key distinguishers proposed by Knudsen and Rijmen [1] are the evidence of this approach, whereas a block-cipher becomes a fixed permutation for a fixed key. The known-key distinguisher efficiently detects non-ideal properties of a random instantiation of a fixed permutation, while the same properties cannot be observed in a random permutation with the same complexity. Knudsen and Rijmen presented a known-key distinguisher on 7-round Feistel ciphers. They also pointed out that their attack detects collisions in hashing modes such as MMO and Miyaguchi-Preneel modes.

At FSE 2011, Sasaki and Yasuda presented another known-key distinguisher on Feistel ciphers [2] with the rebound attack proposed by Mendel *et al.* [3]. They showed that 11 rounds could be attacked if the round function consists of the subkey addition, S-box applications and the permutation layer. In the

Table 1. Complexities against Feistel-SP Ciphers. Notations are defined in Sect. 2.

N	n	c	r	5-round inbound	11-round distinguisher	11-round half-collision	9-round collision	9-round $(N-c)$-bit coll	7-round coll
Complexities of Previous Work [2]									
128	64	8	8	2^{16}	2^{16}	2^{24}	2^{32}	–	–
128	64	4	16	2^{16}	2^{16}	2^{24}	2^{32}	–	–
64	32	8	4	2^{16}	Impossible	Impossible	Impossible	2^{24}	2^{24}
64	32	4	8	2^{8}	2^{8}	2^{12}	2^{16}	–	–
Complexities of Our Attacks									
128	64	8	8	2^{16}	2^{16}	2^{16}	2^{24}	–	–
128	64	4	16	2^{8}	2^{8}	2^{16}	2^{24}	–	–
64	32	8	4	2^{16}	Impossible	Impossible	2^{24}	2^{16}	Impossible
64	32	4	8	2^{8}	2^{8}	2^{12}	2^{16}	–	–

hashing modes, the distinguishers are exploited to apply collision attacks or its variants. Examples of the known-key attack were provided in [4,5,6,2].

A chosen-key distinguisher was firstly proposed by Biryukov and Nikolić [7]. These distinguishers are able to choose the key value. Hash functions are suitable subjects to apply chosen-key distinguishing attacks, since the attacker can control the key values on the hash functions. Several papers have discussed chosen-key distinguishers on block ciphers [8,9].

In this paper, firstly we revisit the known-key distinguisher against generic Feistel-SP ciphers by [2]. One of the core techniques of [2] is a 5-round inbound phase that requires 2^{2c} computations, where c is the size of the S-box. We show that complexity of the 5-round inbound phase can be improved to 2^c computations (i.e. square root of the previous complexity). The new technique makes the attack possible for more number of rounds. Summary of our results in comparison to the previous attacks is shown in Table 1.

In the second part of this paper, we apply the proposed attack on generic Feistel-SP ciphers to Camellia [10]. Camellia is not a plain Feistel-SP cipher due to the P operation, FL, and whitening layers, so the attack needs several modifications. We evaluate Camellia using 128-bit keys including the FL and whitening layers. The best related work were the key-recovery attacks by Lu *et al.* [11] and Li *et al.* [12], which recover the key for 10 rounds of Camellia-128[1]. However, the key-recovery attack does not indicate a faster collision attack than the birthday attack. We show several attacks on hashing modes of Camellia-128 including collisions for 9 rounds. The results are shown in Table 2.

2 Preliminaries

We introduce the following notations. Recall that many of the block ciphers are equipped with 128-bit or 64-bit blocks and use 8-bit or 4-bit S-boxes.

[1] After the submission, 11-round key recovery attacks have been reported [13,14].

Table 2. Summary of attack results on Camellia-128 with FL and whitening layers

Target Modes	#Rounds	Approach	Complexity	Target structure	Reference
Block-Cipher	10	Imp. Diff.	2^{118}		[11]
Hash Function	7	4-sum	2^{32}	⬚⬚■■■▯⬚⬚	Ours
Compression Function	9	Half-collision	2^{16}	⬚⬚⬚■■■▯⬚⬚⬚	Ours
Compression Function	9	4-sum	2^{40}	⬚⬚⬚■■■▯⬚⬚⬚	Ours
Compression Function	9	Collision	2^{48}	⬚▯⬚■■■■■▯⬚⬚	Ours

⬚', ■', and ▯' represent inbound round, outbound round, and FL layer, respectively.

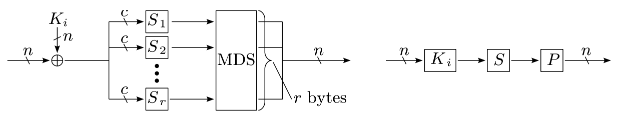


Fig. 1. Left: Detailed description of the SP round function **Right:** Simplified one

N: The block length of the cipher (in bits),
n: The word size in bits, equal to the size of the round function ($n = N/2$).
c: The size of an S-box in bits,
r: The number of S-box sequences, so that $r = n/c$.

SP Round Function. We denote ciphers with the Feistel network and an SP-round function by *Feistel-SP ciphers*, which is specified in Fig. 1.

Key XOR: The round-function input is XORed with a round key K_i.
S-box layer: Each input value is substituted with the output value by S-box. For simplicity, we assume that the S-box is designed to resist differential and linear cryptanalysis, like the one in AES [15,16].
Permutation layer: The linear diffusion is introduced to output sequences of the S-boxes. We make the assumption that the branch number of P is $r + 1$ e.g., a multiplication by an Maximum Distance Separable matrix.

Note that the assumptions on S-box and Permutation layers are not necessary. In fact, the branch number of the permutation layer of Camellia is not $r + 1$. In addition, Whirlpool [17] adopts a more biased S-box, however Lamberger *et al.* [18] showed that the rebound attack for Whirlpool can work similar to AES.

Hashing Modes. Preneel *et al.* [19] considered all possible configurations of a compression function built from a block cipher and proved that 12 modes are secure (providing the block cipher as a family of random permutations indexed by the key [20]). Given a block cipher E_K with a key K, the compression function for the so-called MMO mode computes H_i by $H_i = E_{H_{i-1}}(M_{i-1}) \oplus M_{i-1}$ for a message M_{i-1} and a previous chaining value H_{i-1}. While the Miyaguchi-Preneel mode computes H_i by $H_i = E_{H_{i-1}}(M_{i-1}) \oplus M_{i-1} \oplus H_{i-1}$. To build a hash function, the domain extender must be defined. Because our attack works on 1-block message, we only assume that the initial value H_0 is a fixed constant.

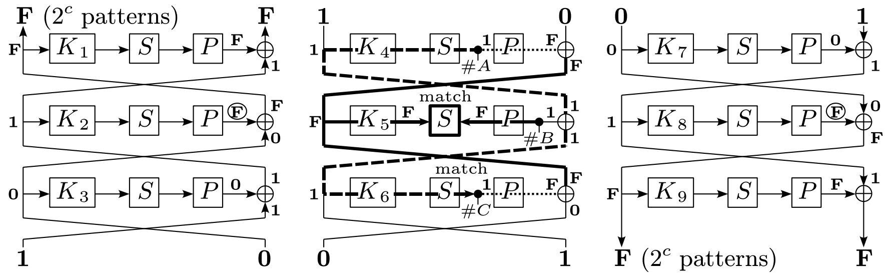


Fig. 2. A differential path for three round inbound and three round outbounds by [2]

3 Previous and Related Work

3.1 Previous Rebound Attack on Feistel-SP Ciphers

Similarly to Sasaki and Yasuda [2], we are going to use the following notations:

- **0**: A word where all the differences are equal to zero.
- **1**: A word where the difference on the j-th byte position is non-zero (we call it active) and the differences on the other byte positions are zeros,
- **F**: A word where all the differences are non-zeros.

The goal of the rebound attack is to find a pair of values that satisfies the truncated differential path. The rebound attack consists of two phases: *inbound* and *outbound*. In the *inbound* phase, the attacker computes differential path using the meet-in-the-middle strategy. In the *outbound* phase, the differential paths are extended outwards (to the input and output) using either deterministic or probabilistic arguments. A sample differential path including 3-round inbound and 3-round outbound phases is given in Fig. 2 (taken from [2]).

The inbound phase starts from the difference $(\mathbf{1}, \mathbf{0})$ and ends with $(\mathbf{0}, \mathbf{1})$. Sasaki and Yasuda [2] presented a procedure to find such a path through 3 and 5 rounds with a complexity of $r \cdot 2^{2c}$ and $r \cdot 2^{2c}$, respectively.

The outbound phase examines whether a solution of the inbound phase satisfies the outbound differential path or not. They showed that the difference $(\mathbf{1}, \mathbf{0})$ goes to the difference $(P(\mathbf{1}), \mathbf{F})$ after 3 rounds backward computation, with probability 1. Similarly, the difference $(\mathbf{0}, \mathbf{1})$ results the difference $(P(\mathbf{1}), \mathbf{F})$ after 3 rounds forward computation, with probability 1. The outbound phase sometimes only covers 2 rounds for each direction. In this case, with probability 1, the differences $(\mathbf{1}, \mathbf{0})$ and $(\mathbf{0}, \mathbf{1})$ on the inbound sides result differences $(\mathbf{1}, P(\mathbf{1}))$ and $(\mathbf{1}, P(\mathbf{1}))$ after 2 rounds backward and forward computations, respectively.

The 11-round distinguisher is built from a 5-round inbound part followed by 3-round outbound parts in both directions. It allows to find a pair of plaintext values whose difference is $(P(\mathbf{1}), \mathbf{F})$ and a pair of ciphertext values with the difference $(P(\mathbf{1}), \mathbf{F})$. The complexity of this attack is $r \cdot 2^{2c}$. On the other hand, the complexity of the generic attack is equal to the birthday attack on $n - c$ bits, which requires $2^{(n-c)/2}$ computations. Moreover this does not always work,

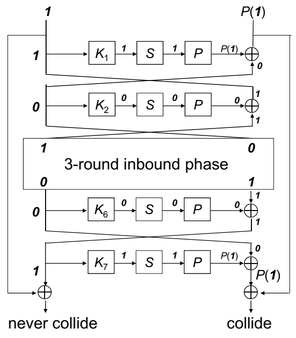


Fig. 3. Differential Path for 7R Collisions

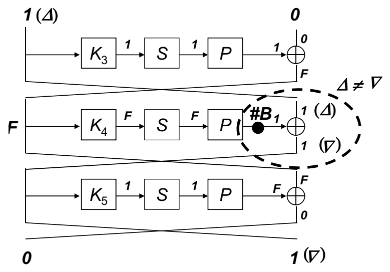


Fig. 4. Inbound Phase with $\varDelta \neq \nabla$

when $(N, c) = (128, 4)$, two bytes must be activated to increase the degree of freedom, and the complexity is higher, 2^{4c} in this specific case.

The 11-round distinguisher was exploited in [2] to attack the hashing modes. In the MMO and Miyaguchi-Preneel modes, the plaintext and ciphertext are XORed. Hence, if the left half of the differences in the 11-round distinguisher is cancelled, the half-state collision is obtained. Several other attacks were presented in [2]. The attack complexities are summarized in Table 1.

4 New Known-Key Distinguishers on Feistel-SP Ciphers

First of all, we explain the impossibility of the previous 7-round collision attack in Sect. 4.1. Then an improved 5-round inbound phase will be described in Sect. 4.2. Finally, in Sect. 4.3, we show that 4-sums can be detected even on the full state of hashing modes with 11-round Feistel-SP ciphers.

4.1 Flaw of Previous 7-Round Collisions for $(N, c) = (64, 8)$

It was claimed that collisions could be obtained for 7 rounds with parameters $(N, c) = (64, 8)$ in the MMO and Miyaguchi-Preneel modes [2, Sect.4.4]. Fig. 3 illustrates the attack with 3-round inbound phase and two (2-round) outbound phases. The attacker first generates a pair of values satisfying the differences of the inbound phase, namely $(\varDelta, 0) \xrightarrow{3\mathrm{R}} (0, \nabla)$. Through the outbound phase, the plaintext difference and ciphertext difference becomes $(\varDelta, P(\varDelta'))$ and $(\nabla, P(\nabla'))$. It was claimed that $\varDelta = \nabla$ and $\varDelta' = \nabla'$ are satisfied with probability 2^{-2c}. However, we prove that $\varDelta = \nabla$ is an impossible event and always $\varDelta \neq \nabla$.

Lemma 1. *Given a 7-round Feistel-SP cipher, the collision attack based on the rebound attack with 3-round inbound phase and 2-round outbound phases always fails.*

Proof. Inside the 3-round inbound phase in Fig. 4, to obtain $\Delta = \nabla$, the difference right after the P-layer in the middle round (denoted by $\#B$) must be $\mathbf{0}$. However, to perform the rebound attack, the difference at $\#B$ is always $\mathbf{1}$. Hence, $\Delta = \nabla$ is never satisfied. □

4.2 Improved 5-Round Inbound Phase with 2^c Computations

In this section, the improved inbound phase with 2^c computations is described. First we give an overview of the attack presented in [2].

First Inbound Phase: A pair of values which follow the differential path is obtained. 2^c solutions are generated with the workload of 2^c computations.

Second Inbound Phase: A pair of values for the 4th and 5th inbound rounds that follow the differential path is obtained. It costs 2^c computations.

Merge Inbound Phase: All possible solutions of the first and second inbound phases are combined in the third inbound round.

Validity Check Phase: Pairs of values generated by the merge inbound phase are computed from the third inbound round to the first and fifth inbound rounds, respectively. Then, 1-byte match is performed in both of the first and fifth rounds. The probability of each match is 2^{-c}, thus 2^{-2c} totally. Finally, by trying 2^{2c} combinations at the merge inbound phase, a pair which satisfies two matches is obtained.

In our improved attack, the differential path is developed using S-box differential profile. This approach leads to an attack with better complexity. The core of the improvement is the merging phase, in which we combine the results of the two inbound phases so that an n-bit match condition is always satisfied. This reduces the complexity of each step below by 2^c. In more details, we change the merge inbound phase so that the validity check is carried out in the third and fifth inbound rounds. We first choose a solution of the first inbound phase from 2^c candidates. Then, at the validity check in the third inbound round, we only choose solutions of the second inbound phase which satisfy the n-bit condition. Finally, the validity check at the fifth inbound round succeeds with probability 2^{-c}. By iterating this for 2^c candidates of the first inbound phase, we will succeed the validity check in the fifth round with a negligible cost.

Parallel Check of Differences in the First Inbound Phase. The goal of the first inbound phase is to find the differential path for the first two inbound rounds which is shown in Fig. 5. The original attack procedure is as follows [2]:

1. Search for several pairs of differences $\Delta\#A^j$ and $\Delta\#B^j$ such that $P(\Delta\#A^j)$ and $P^{-1}(\Delta\#B^j)$ have solutions for all S-boxes in the second inbound round (bold line in Fig. 5).
2. For each pair, exhaustively try 2^c values of $\#A^j$ which is denoted by x, and check if $S^{-1}(x) \oplus S^{-1}(x \oplus \Delta\#A^j)$ can cancel $\Delta\#B^j$ (broken line in Fig. 5).

Step 1 and Step 2 are independently performed. Thus the possibility of the 1-byte match at Step 2 is never considered during Step 1.

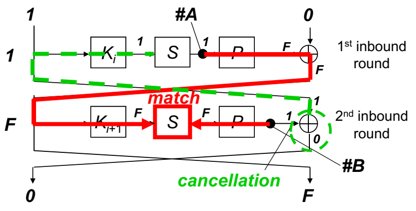


Fig. 5. Parallel check of the match of differences

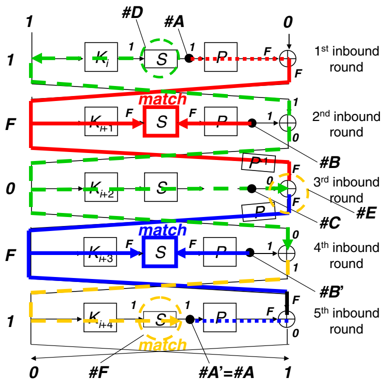


Fig. 6. Improved 5-round inbound phase

We notice that fixing the differences $\Delta\#A^j$ and $\Delta\#B^j$ for Step 1 immediately fixes the input and output differences of the active S-box for Step 2. This indicates that $(\Delta\#A^j, \Delta\#B^j)$ determined at Step 1 may not have any solution for $\Delta\#A^j \xrightarrow{S^{-1}} \Delta\#B^j$ at Step 2. Obviously, spending 2^c computations for such $(\Delta\#A^j, \Delta\#B^j)$ at Step 2 is meaningless. In other words, by only choosing $(\Delta\#A^j, \Delta\#B^j)$ which is known to have solutions at Step 1, the probability of the match at Step 2 is increased.

For any input and output differences of the S-box $(\Delta I, \Delta O)$ which is known to have solutions, $Pr\big[\big(S(x) \oplus S(x \oplus \Delta I)\big) = \Delta O\big] \geq 2^{-c+1}$, where x is a randomly chosen value. Hence, the probability increases from 2^{-c} to 2^{-c+1}. So the parallel check is applied to the match in the fourth and fifth inbound rounds (second inbound phase). Therefore, the entire probability increases from 2^{-2c} to 2^{-2c+2}.

Attack Procedure. For simplicity, we first assume that all S-boxes are identical and r and c satisfy $c \geq r + 1$ (It only happens for $(N, c) = (64, 8)$). The assumption is going to be weakened later on. In the following, we explain the attack procedure as illustrated in Fig. 6. In Fig. 6, the equivalent transformation is applied to the third inbound round. The first inbound phase is denoted by red lines. Similarly, the second inbound phase, the merge inbound phase, and the validity check are denoted by blue, green and yellow lines, respectively.

First Inbound Phase: Choose a difference at $\#A$ (i.e. $\Delta\#A$), and compute $P(\Delta\#A)$, which is an input to the S-layer in the second inbound round. Then, choose all differences at $\#B$ (i.e. $\Delta\#B$) such that the differential propagation through the active S-box in the first inbound round can have solutions, namely, $\exists x : S(x) \oplus S(x \oplus \Delta\#B) = \Delta\#A$. The number of such $\Delta\#B$ is approximately 2^{c-1}. For 2^{c-1} choices of $\Delta\#B$, compute $P^{-1}(\Delta\#B)$, which is an output of the S-layer in the second inbound round. Check if all S-boxes in the second inbound round have solutions. If the check succeeds, for each of the possible solutions, store the corresponding pair of values at state $\#B$ by computing P in the second inbound round and pair of values at state $\#E$ by computing the subkey XOR and then $P^{-1}(\cdot)$. Let T_1 be the table in which these values are stored. T_1 is expected to have 2^{c-1} entries.

Second Inbound Phase: Set $\Delta\#A' \leftarrow \Delta\#A$. Similar to the first inbound phase, compute 2^{c-1} solutions of the last two inbound rounds and store the paired values at $\#E$ in table T_2.

Merge Inbound Phase: For 2^{c-1} solutions of the first two inbound rounds stored in T_1 and all solutions (2 solutions on average) of the active S-box in the first inbound round (at state $\#D$), do the followings: Regarding only the active byte, compute the value up to the state $\#C$. If the computed value at state $\#E$ matches one of the entries in T_2, fix the solution for the last two rounds to this value. Go to the validity check with this value.

Validity Check Phase: Regarding only the active byte, compute the value up to the output of the active S-box in the fifth inbound round ($\#F$). If the computed difference matches $\Delta\#A'$, the paired values are the valid solutions. Otherwise, go back to the merge inbound phase.

Attack Evaluation. The complexity of the first inbound phase is 2^c 1-round computations in time and 2^{c-1} state in memory. The number of solutions over the P-layer in the second middle round is 2^{c-1}, and for each of them, 2 solutions are obtained for the active S-box in the first inbound round. Overall, the first inbound phase produces 2^c solutions with a cost of 2^c. The evaluation for the second inbound phase is the same. In the merge inbound phase, the number of trials is 2^c. The match at state #E succeeds with probability 2^{-c}. Since 2^c solutions are stored in T_2, we expect to find one match for each trial. Then, the matched result is computed up to state $\#F$ for the validity check. The success probability is 2^{-c}. Since the merge inbound phase is iterated 2^c times, we expect to find one solution of the validity check. The merge inbound phase and the validity check require 2^c 5-round computations. Finally, one solution of the inbound phase is computed with 2^c 1-round $+\,2^c$ 1-round $+\,2^c$ 5-round $=$ 2^c 7-round computations and 2^c state in memory.

Remarks for Other Parameters. The above attack can also be applied to other parameters with several minor changes. First of all we explain the case for $c = r$. $(N, c) = (128, 8)$ is included in this parameter. In this case, the freedom degrees will be slightly short in the first inbound phase (Step 5) because only $2^{c-1} = 2^{r-1}$ pairs can be examined. This problem is solved by running the first inbound phase for two different $\Delta\#A$. Hence, the complexity does not change from 2^c, and our procedure can also be applied to the parameter $c = r$.

The attack is also applied to the cases $(N, c) = (128, 4)$ and $(64, 4)$. We regard a group of two S-boxes as a big S-box with the size of $2c$ bits. This gives enough freedom degrees to find the match of differences over the S-layer. The attack becomes the same as $(N, c) = (128, 8)$ and $(64, 8)$.

Impact of the Improvement. Due to the improvement of the complexity of the 5-round inbound phase, many attacks presented in [2] are improved. The updated results are summarized in Table 1.

4.3 4-Sums on Compression Function Modes

The notion of *4-sum* represents 4 different inputs where the XOR-sum of the corresponding outputs is 0. The 4-sum is known to have many applications such as an attack on a random oracle instantiation [21] and on a signature scheme [22]. In the 11-round attack, the right halves of the plaintext and ciphertext do not have any property. Thus, the previous attack could only detect a non-ideal property on the left-half of the state. We explain that 4-sums can be obtained on the full state in compression function modes e.g. MMO and Miyaguchi-Preneel modes.

Each solution of the inbound phase produces a pair of outputs whose difference is the form $(P(\mathbf{1}), \mathbf{F})$. Let t_1 and t_2 be the paired solutions of the inbound phase, and $O(t_1)$ and $O(t_2)$ be corresponding outputs. Then, $O(t_1) \oplus O(t_2) = (P(\mathbf{1}), \mathbf{F})$. Assume that you have two pairs (t_1, t_2) and (t'_1, t'_2). The second-order difference $(O(t_1) \oplus O(t_2)) \oplus (O(t'_1) \oplus O(t'_2))$ becomes 0 if $O(t_1) \oplus O(t_2) = O(t'_1) \oplus O(t'_2)$. Because $O(t_1) \oplus O(t_2)$ takes 2^{c+n} possibilities, 4-sums can be generated with $2^{(c+n)/2}$ solutions of the inbound phase due to the birthday paradox.

5 Applications to Camellia and Its Hashing Modes

5.1 Specification of Camellia

Camellia was jointly designed by NTT and Mitsubishi Electric Corporation. It is widely standardized such as ISO [23], NESSIE [24], and CRYPTREC [25]. This paper attacks Camellia-128, where both of the key and block sizes are 128 bits.

Let M and K be 128-bit plaintext and secret key, respectively. Eighteen 64-bit round keys $K_1, \ldots, K_{18}$, four 64-bit whitening keys $kw_1, \ldots, kw_4$, and four 64-bit subkeys for the FL layer $kl_1, \ldots, kl_4$ are generated from K. Let L_r and R_r $(0 \leq r \leq 18)$ be left and right 64-bits of the internal state in each round. The plaintext is loaded into $L_0 \| R_0$ after the whitening operation, i.e. $L_0 \| R_0 \leftarrow M \oplus (kw_1 \| kw_2)$. $(L_{18} \| R_{18})$ is computed by $L_r = R_{r-1} \oplus F(L_{r-1}, K_r), R_r = L_{r-1}$ for $1 \leq r \leq 18$. Note that the FL and FL^{-1} functions are applied to L_r and R_r for $r = 6$ and 12. Finally, $(L_{18} \| R_{18}) \oplus (kw_3 \| kw_4)$ is the ciphertext.

Key Schedule. The key schedule takes 128-bit key K as input and firstly produces another 128-bit value K_A. In our known-key attacks, subkeys are randomly given values, thus the key schedule function is irrelevant. In chosen-key attacks, we only need to control the values of kl_1, kl_2, kl_3, and kl_4. kl_1 is the left 64-bits of $(K_A \lll 30)$ and kl_2 is the right 64-bits of $(K_A \lll 30)$. kl_3 is the left 64-bits of $(K_L \lll 77)$ and kl_4 is the right 64-bits of $(K_L \lll 77)$.

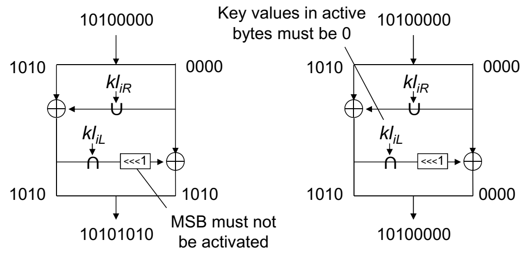


Fig. 7. Differential propagation through FL^{-1} for known-key (left) and chosen-key (right)

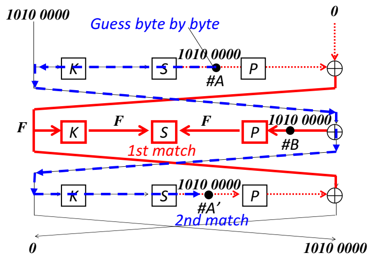


Fig. 8. 3-round inbound phase with 2 active bytes

Round Function. The round function consists of the subkey addition, S-layer and P-layer. For the S-layer, 4 S-boxes are defined. The DDTs for these S-boxes have the same property as the one for AES. Let $(z_1\|z_2\|\cdots\|z_8)$ be 64-bit values input to the P-layer. The output $(z'_1\|z'_2\|\cdots\|z'_8)$ is computed as follows. Here, $z[s,t,u,\cdots]$ means $z_s\oplus z_t\oplus z_u\oplus\cdots$. The branch number of P is only 5.

$$\begin{aligned} z'_1 &= z[1,3,4,6,7,8], & z'_3 &= z[1,2,3,5,6,8], & z'_5 &= z[1,2,6,7,8], & z'_7 &= z[3,4,5,6,8],\\ z'_2 &= z[1,2,4,5,7,8], & z'_4 &= z[2,3,4,5,6,7], & z'_6 &= z[2,3,5,7,8], & z'_8 &= z[1,4,5,6,7]. \end{aligned}$$

***FL* and *FL*$^{-1}$ Functions.** The FL function takes a 64-bit value $(X_L\|X_R)$ and a 64-bit subkey $(kl_L\|kl_R)$ as input and produces a 64-bit value $(Y_L\|Y_R)$ by computing $Y_R=\big((X_L\cap kl_L)\lll 1\big)\oplus X_R$ and $Y_L=(Y_R\cup kl_R)\oplus X_L$, where $\cap$, $\cup$, and $\lll 1$ are the logical AND, OR, and a left cyclic shift by 1 bit, respectively.

5.2 Applications to Camellia Hashing Modes

A Small Branch Number in the *P*-Layer. The attack in Sect. 4 assumes that the branch number of the P function is $r+1$, i.e. 9 for Camellia. Otherwise, $P(\mathbf{1})$ and $P^{-1}(\mathbf{1})$ may not become full active in the first inbound phase. However, the branch number of the P function in Camellia is 5.

We avoid this problem by activating more bytes at state $\#A$ and $\#B$ in Fig. 6. Since increasing the number of active bytes makes the attack inefficient, we need to choose active byte positions carefully. The conditions of active byte positions are as follows; 1) Active byte positions for $\#A$ and $\#B$ must be identical. 2) The active byte positions of $P(\Delta\#A)$ and $P^{-1}(\Delta\#B)$ must be identical.

We first search for a single active byte position satisfying these conditions. Since the value of r is 8, we have 8 possibilities. Unfortunately, any of the 8 cases cannot satisfy the conditions. We then search to find the positions of two active bytes. There are $\binom{8}{2}=28$ possibilities. The result is shown in Table 3. We found 8 solutions for two-active byte differences. Hereafter, 5th and 7th bytes are activated. We also use the notation `10100000` to represent that only 5th and 7th bytes have differences in some states.

Table 3. Active-byte positions

	0th	1st	2nd	3rd	4th	5th	6th	7th
0th		×	✓	×	×	×	×	×
1st			×	✓	×	×	×	×
2nd				×	×	×	×	×
3rd					×	×	×	×
4th						✓	✓	✓
5th							✓	✓
6th								✓

Table 4. Sample solutions of the improved 5-round inbound phase for Camellia-128 (in hexadecimal)

	L	$\triangle_L$	R	$\triangle_R$
Input	3C0EF35D6F89AE61	0F00A20000000000	33F035B96274F068	0000000000000000
Output	32848AF48CE3D7EB	0000000000000000	A383FB0B17307503	2200810000000000
Input	1447452393AF07BF	D5006B0000000000	D7A0E862AF8343B9	0000000000000000
Output	FD6901A999C97E6C	0000000000000000	CE1249DCA3LC251B	C400D40000000000
Input	AB131279BE6E5342	3E00780000000000	6EF4FD4CA00881EB	0000000000000000
Output	798EB0992C1C6160	0000000000000000	301DBF0D730C71AD	0600780000000000
Input	62B5DB720A58E01D	33002D0000000000	95AAF7AD94613A12	0000000000000000
Output	8B24C8FAA65E8E33	0000000000000000	D2F68668ADE68225	C6001B0000000000

***FL* and $\boldsymbol{FL^{-1}}$ Functions.** These functions mix the difference of the half state. If they are inserted inside the inbound rounds, the differential path would be broken. Hence, we choose the starting round of our attacks carefully. As shown in Table 2, the FL-layer is located immediately after the inbound phase. This avoids inserting the FL-layer in the middle of the inbound phase.

In the following, we analyze the differential propagation through FL and FL^{-1}. Assume that the FL-layer is used only once immediately after the inbound phase. For the FL-layer, the form of the input differences is always $(\mathbf{0}, \texttt{10100000})$. It is obvious that $\mathbf{0}$ results in $\mathbf{0}$. Therefore, we only need to consider the right half of the state, where `10100000` is input to the FL^{-1} function. The differential propagation through FL^{-1} is described in Fig. 7. Distinguishers want to keep the number of active bytes low. Hence, in the known-key setting, distinguishers avoid activating the most significant bit (MSB) in each byte in order to prevent the difference from propagating through $\lll 1$ operation. This reduces the degrees of freedom while selecting differences. However, there is enough freedom for the attack to proceed as expected. Note that if the MSB in each active byte of the key kl_{iL} is 1, distinguishers can activate the MSB in each byte. Such weak keys exist with probability 2^{-2}, and the number of weak keys is $2^{128-2} = 2^{126}$. In the chosen key scenario, distinguishers choose the key value. Choosing one subkey value is trivially done with complexity 1 for any key value. This is because kl_i is a part of K or K_A. If it is a part of K, distinguishers directly choose the value. If it is a part of K_A, distinguishers firstly choose K_A and then invert it to K through the key schedule.

As shown in Table 2, distinguishers need to control two FL-layers in the chosen-key attacks with 5-round inbound phase. The FL-layer is inserted between the first and the second rounds of the backward outbound. According to Fig. 2, the form of the input differences to the inverse of the FL-layer is $(\mathbf{0}, \texttt{10100000})$. Therefore, we need to analyze the differential propagation of $(FL^{-1})^{-1}(\texttt{10100000})$, which is equivalent to $FL(\texttt{10100000})$. The analysis is the same as the previous one, and we omit it. As a result, if distinguishers can choose the values of 2 active bytes of the subkey, the form of the output difference is unchanged from `10100000`. Finally, distinguishers need to control two bytes in K_A and two bytes in K. We search for such keys by the brute force manner. The success probability of this event is 2^{-16}.

3-Round Inbound Phase and Its Application. The procedure of the 3-round inbound phase is heavily based on the one in previous work [2]. Our attack activates 2-bytes, however, we can keep the attack complexity unchanged with some optimization. The procedure is described in Fig. 8. The attack first chooses the differences of state #A and #B so that the differences can match over the S-layer in the second inbound round. If they match, one solution for all bytes are chosen and the corresponding paired values denoted by red in Fig. 8 are computed. Then, the attack searches for the value of each active byte at state $\#A$ that can satisfy the difference of the same byte position at state $\#A'$. Finally, the 3-round inbound phase is carried out with 2^8 computations, and once it is satisfied, up to 2^{40} solutions can be generated for free.

As listed in Table 2, the 3-round inbound phase has 3 applications. Due to the page limitation, we only explain 4-sums on the 7-round hash function. Hereafter we denote by **2** the difference form of `10100000`, and by **4** the difference form of `10101010`. The differential path that we use is $(\mathbf{2}, P(\mathbf{2})) \xrightarrow{2^{\text{nd}}\text{R}} (\mathbf{0}, \mathbf{2}) \xrightarrow{3^{\text{rd}}\text{R}} (\mathbf{2}, \mathbf{0})$ for the first outbound rounds, $(\mathbf{2}, \mathbf{0}) \xrightarrow{4^{\text{th}}\text{R}} (\mathbf{F}, \mathbf{2}) \xrightarrow{5^{\text{th}}\text{R}} (\mathbf{2}, \mathbf{F}) \xrightarrow{6^{\text{th}}\text{R}} (\mathbf{0}, \mathbf{2})$ for the inbound rounds, and $(\mathbf{0}, \mathbf{2}) \xrightarrow{FL} (\mathbf{0}, \mathbf{4}) \xrightarrow{7^{\text{th}}\text{R}} (\mathbf{4}, \mathbf{0}) \xrightarrow{8^{\text{th}}\text{R}}$ for the second outbound rounds. The number of active bytes increases by the FL-layer. After the feed-forward, the output should be $\Big(\mathbf{2} \oplus \mathbf{4}, P(2) \oplus P(4)\Big)$, which is in the space of $(\mathbf{4}, P(4))$. With the technique in Sect. 4.3, if we generate $2^{8*4} = 2^{32}$ pairs, differences on two pairs collide, and form a 4-sum. Since 3-round inbound phase generates up to 2^{40} solutions for free, our attack requires 2^{32} computations, which is equivalent to the information theoretic bound to generate a 4-sum $(2^{128/4})$, and faster than the generalized birthday attack.

5-Round Inbound Phase and Its Application. The procedure for the 5-round inbound phase is basically the same as the generic case while activating two S-boxes in Camellia is the same as activating 1 big S-box of the size $2c = 16$ bits. Hence, one solution of the 5-round inbound phase is obtained with 2^{16} in both time and memory. This leads to collisions on the 9-round compression function. The differential path that we use is $(\mathbf{2}, P(\mathbf{2})) \xrightarrow{6^{\text{th}}\text{R}} (\mathbf{0}, \mathbf{2}) \xrightarrow{FL} (\mathbf{0}, \mathbf{2}) \xrightarrow{7^{\text{th}}\text{R}} (\mathbf{2}, \mathbf{0})$ for the first outbound rounds, $(\mathbf{2}, \mathbf{0}) \xrightarrow{8^{\text{th}}\text{R}} (\mathbf{F}, \mathbf{2}) \xrightarrow{9^{\text{th}}\text{R}} (\mathbf{0}, \mathbf{F}) \xrightarrow{10^{\text{th}}\text{R}} (\mathbf{F}, \mathbf{0}) \xrightarrow{11^{\text{th}}\text{R}} (\mathbf{2}, \mathbf{F}) \xrightarrow{12^{\text{th}}\text{R}} (\mathbf{0}, \mathbf{2})$ for the inbound rounds, and $(\mathbf{0}, \mathbf{2}) \xrightarrow{FL} (\mathbf{0}, \mathbf{2}) \xrightarrow{13^{\text{th}}\text{R}} (\mathbf{2}, \mathbf{0}) \xrightarrow{14^{\text{th}}\text{R}} (\mathbf{2}, P(\mathbf{2}))$ for the second outbound rounds. The difference of the output of the compression function has the form $(\mathbf{2}, P(\mathbf{2}))$. Hence, if we generate such pairs 2^{32} times, the difference is **0** in one pair. With the improved 5-round inbound phase in Sect. 4.2, 1 solution of the inbound phase is generated with 2^{16} computations. Hence, 2^{32} solutions are generated with 2^{48} computations, which is faster than the birthday attack on a 128-bit value.

5.3 Experiments and Generated Data

To verify the attacks, we implemented the chosen-key 5-round inbound phase. First of all, a valid key is chosen to bypass the FL^{-1} layer after round 6. We find these keys in the expected time of about 2^{16} computations in average. Then, the chosen keys are used to find solutions of the new 5-round inbound phase. Table 4 shows some of the found solutions. Most of the solutions are found in less than 2^8 computations in our experiments. Therefore the experimental complexity is less than predicated in theory (i.e. 2^{16} computations).

6 Concluding Remarks

In this paper, we revisited the known-key attacks on generic Feistel-SP ciphers. Our main contribution is a new 5-round inbound phase which results in an improved complexity and works for a high number of rounds. Then with several modifications, the framework was applied to Camellia-128. Our results have been confirmed by computer simulations. We have presented several new attacks on the hashing modes of Camellia-128 including a collision attack on 9 rounds.

References

1. Knudsen, L.R., Rijmen, V.: Known-Key Distinguishers for Some Block Ciphers. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 315–324. Springer, Heidelberg (2007)
2. Sasaki, Y., Yasuda, K.: Known-Key Distinguishers on 11-Round Feistel and Collision Attacks on Its Hashing Modes. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 397–415. Springer, Heidelberg (2011)
3. Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 260–276. Springer, Heidelberg (2009)
4. Mendel, F., Peyrin, T., Rechberger, C., Schläffer, M.: Improved Cryptanalysis of the Reduced Grøstl Compression Function, ECHO Permutation and AES Block Cipher. In: Jacobson Jr., M.J., Rijmen, V., Safavi-Naini, R. (eds.) SAC 2009. LNCS, vol. 5867, pp. 16–35. Springer, Heidelberg (2009)
5. Minier, M., Phan, R.C.-W., Pousse, B.: Distinguishers for Ciphers and Known Key Attack against Rijndael with Large Blocks. In: Preneel, B. (ed.) AFRICACRYPT 2009. LNCS, vol. 5580, pp. 60–76. Springer, Heidelberg (2009)
6. Sasaki, Y.: Known-Key Attacks on Rijndael with Large Blocks and Strengthening *ShiftRow* Parameter. In: Echizen, I., Kunihiro, N., Sasaki, R. (eds.) IWSEC 2010. LNCS, vol. 6434, pp. 301–315. Springer, Heidelberg (2010)
7. Biryukov, A., Nikolić, I.: A new security analysis of AES-128. Rump session of CRYPTO 2009 (2009), http://rump2009.cr.yp.to/
8. Biryukov, A., Nikolić, I.: Automatic Search for Related-Key Differential Characteristics in Byte-Oriented Block Ciphers: Application to AES, Camellia, Khazad and Others. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 322–344. Springer, Heidelberg (2010)
9. Nikolić, I., Pieprzyk, J., Sokołowski, P., Steinfeld, R.: Known and Chosen Key Differential Distinguishers for Block Ciphers. In: Rhee, K.-H., Nyang, D. (eds.) ICISC 2010. LNCS, vol. 6829, pp. 29–48. Springer, Heidelberg (2011)
10. Aoki, K., Ichikawa, T., Kanda, M., Matsui, M., Moriai, S., Nakajima, J., Tokita, T.: *Camellia*: A 128-Bit Block Cipher Suitable for Multiple Platforms - Design and Analysis. In: Stinson, D.R., Tavares, S. (eds.) SAC 2000. LNCS, vol. 2012, pp. 39–56. Springer, Heidelberg (2001)

11. Lu, J., Wei, Y., Kim, J., Fouque, P.A.: Cryptanalysis of reduced versions of the Camellia block cipher. In: Selected Areas in Cryptography 2011 (2011), http://sac2011.ryerson.ca/SAC2011/LWKF.pdf
12. Li, L., Chen, J., Jia, K.: New Impossible Differential Cryptanalysis of Reduced-Round Camellia. In: Lin, D., Tsudik, G., Wang, X. (eds.) CANS 2011. LNCS, vol. 7092, pp. 26–39. Springer, Heidelberg (2011)
13. Liu, Y., Li, L., Gu, D., Wang, X., Liu, Z., Chen, J., Li, W.: New observations on impossible differential cryptanalysis of reduced-round Camellia. In: Fast Software Encryption 2012. Springer (to appear, 2012)
14. Bai, D., Li, L.: New Impossible Differential Attacks on Camellia. In: Ryan, M.D., Smyth, B., Wang, G. (eds.) ISPEC 2012. LNCS, vol. 7232, pp. 80–96. Springer, Heidelberg (2012)
15. Daemen, J., Rijmen, V.: AES Proposal: Rijndael (1998)
16. U.S. Department of Commerce, National Institute of Standards and Technology: Specification for the ADVANCED ENCRYPTION STANDARD (AES) (Federal Information Processing Standards Publication 197) (2001)
17. Rijmen, V., Barreto, P.S.L.M.: The WHIRLPOOL hashing function. Submitted to NISSIE (2000)
18. Lamberger, M., Mendel, F., Rechberger, C., Rijmen, V., Schläffer, M.: Rebound Distinguishers: Results on the Full Whirlpool Compression Function. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 126–143. Springer, Heidelberg (2009)
19. Preneel, B., Govaerts, R., Vandewalle, J.: Hash Functions Based on Block Ciphers: A Synthetic Approach. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 368–378. Springer, Heidelberg (1994)
20. Black, J.A., Rogaway, P., Shrimpton, T.: Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 320–335. Springer, Heidelberg (2002)
21. Leurent, G., Nguyen, P.Q.: How Risky Is the Random-Oracle Model? In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 445–464. Springer, Heidelberg (2009)
22. Wagner, D.: A Generalized Birthday Problem. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 288–303. Springer, Heidelberg (2002)
23. International Organization for Standardization: ISO/IEC 18033-3, Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers (2005)
24. NESSIE-New European Schemes for Signatures, Integrity, and Encryption: Final report of European project IST-1999-12324 (1999), https://www.cosic.esat.kuleuven.be/nessie/Bookv015.pdf
25. CRYPTREC-C-Cryptography Research and Evaluation Committees: e-Government recommended ciphers list (2003), http://www.cryptrec.go.jp/english/index.html

Low Data Complexity Attack on Reduced Camellia-256[⋆]

Jiazhe Chen[1,2,⋆⋆] and Leibo Li[1]

[1] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, School of Mathematics, Shandong University, Jinan 250100, China
[2] KU Leuven, ESAT/COSIC and IBBT, Belgium

Abstract. This paper proposes a low data complexity attack on reduced-round block cipher Camellia. Utilizing a 7-round meet-in-the-middle distinguisher with an FL layer between the fifth and the sixth round, one can attack 12-round Camellia-256 with 2^{19} chosen plaintexts and $2^{231.2}$ encryptions. This attack starts from the first round of Camellia-256, so as to keep the property of Camellia that inserting the FL layers every 6 rounds; it also takes the whitening keys into account. Compared with the recent proposed attacks on Camellia-256, the attack in this paper has much lower data complexity; at the same time, it is also the best attack on Camellia-256 in terms of the number of rounds and the time complexity, if one only consider the 'regular' reduced Camellia with 6 rounds before (after) the first (last) FL layer and with whitening keys.

Keywords: Block Cipher, Camellia, Meet-in-the-Middle Attack, Low Data Complexity, Cryptanalysis.

1 Introduction

Block cipher Camellia is proposed by NTT and Mitsubishi in 2000 [1]. Its block size is 128 bits and it supports 128-, 192- and 256-bit key sizes with 18, 24 and 24 rounds respectively. Camellia was selected as an e-government recommended cipher by CRYPTREC [6] and recommended in NESSIE [23] block cipher portfolio. Then it was selected as an international standard by ISO/IEC 18033-3 [12]. It is also a part of the OpenSSL Project [24] and the Mozilla's NSS (Network Security Services) module [22].

Due to its wide use, Camellia has been analyzed since it was proposed. The structure of Camellia is Feistel structure with FL and FL^{-1} functions inserted

⋆ This work is supported by the National Natural Science Foundation of China (Grant No. 61133013 and No. 60931160442), in part by the Research Council K.U.Leuven: GOA TENSE, the IAP Program P6/26 BCRYPT of the Belgian State (Belgian Science Policy), and in part by the European Commission through the ICT program under contract ICT-2007-216676 ECRYPT II.

⋆⋆ This author is supported by Graduate Independent Innovation Foundation of Shandong University (No. 11140070613183).

every 6 rounds. The FL and FL^{-1} functions[1] are keyed linear functions which are designed to provide faster diffusion. A number of cryptanalytic results on simplified Camellia which excludes the FL layers and whitening are given, for example [9,14,18,21,25,26,27,28]. Although it will increase the difficulty for cryptanalysis, the attacks that take the FL layers into account seem more attractive [11,15].

Recently, more papers are studying the security of Camellia with FL layers. On the one hand, several impossible differential attacks [4,13] are proposed; these attacks are based on impossible differentials with FL layers. Chen et al. presented a 6-round impossible differential with an FL layer in the middle; with this impossible differential, they gave attacks on 10-round Camellia-192 and 11-round Camellia-256 [5]. With the same impossible differential, Lu. et al. presented impossible differential attacks on 10-round Camellia-128[2] and 11-round Camellia-192 [19]. Later, Li et al. proposed a 7-round impossible differential with an FL layer before the 7th round [16] to analyze 10-/10-/11-round Camellia-128/-192/-256 (from the first round). Then 7-round impossible differentials (with FL layers after the 5th round or before the 3rd round) which hold with probability 75% were proposed in [17]; the weak-key impossible differential attacks on 10-/11-/12-round Camellia-128/-192/-256 and 14-round irregular Camellia-256 could also be converted to attacks that hold for the whole key space. In [17], they also built a set of differentials which contains at least one 8-round impossible differential of Camellia with two FL layers, which led to attacks on irregular Camellia-128/-192/-256 reduced to 11/12/13 rounds. There is also a 7-round impossible differential with 2 FL layers given by Bai et al. [3], which results in reduced irregular Camellia without whitening. All the data complexities of these attacks are higher than 2^{116} bytes, due to the inherent requirement of impossible differential attack that one has to find enough pairs to meet the input and output differences of the impossible differential thus to discard the wrong keys.

On the other hand, Lu et al. also proposed attacks on 10-/11-/12-round Camellia-128/-192/-256 with FL layers (without whitening keys) [20], but with a method named higher-order meet-in-the-middle attack. The data complexities of their attacks are around 2^{94} bytes.

Since data can only be collected during the online phase of the attack, sometimes it is more difficult to obtain than time and memory. Hence in this paper, we attempt to reduce the data complexity of the 12-round attack on regular Camellia-256 with whitening keys. This is achieved by getting rid of the integral property, extending the distinguisher in [20] to 7-rounds, and leveraging the meet-in-the-middle attack to Camellia-256 reduced to 12 rounds.

[1] The FL and FL^{-1} functions are called FL layers for simplicity in the rest of the paper.

[2] However, this attack is from Round 8 to Round 17, i.e., the target reduced Camellia that can be broken using chosen plaintext attack has only 5 rounds (less than 6 rounds) before the FL layer; in this paper, we call this kind of variants of reduced Camellia as *irregular*, while the ones with 6 rounds before/after the first/last FL layer (in chosen-plaintext/-ciphertext setting) as *regular*.

The meet-in-the-middle (MITM) attack was first applied to block cipher by Diffie and Hellman [8]. The idea is as follows: A block cipher E_K can be regard as the cascade of two subciphers $E_K = E^2_{K_2} \circ E^1_{K_1}$; for a plaintext-ciphertext pair (P, C), the adversary guesses K_1, K_2 and check whether $E^1_{K_1}(P) = E^{2^{-1}}_{K_2}(C)$. If so, then the adversary might have the right key; otherwise, the guessed key must be wrong. This attack will be slower than the exhaustive search if the key information included in the union set of K_1 and K_2 ($K_1 \bigcup K_2$) is more than that of K. However, the adversary can apply the time-memory trade-off technique to pre-compute the value of $E^1_{K_1}(P)$ under each K_1 and store then in a hash table; then he guesses K_2, calculate $E^{2^{-1}}_{K_2}(C)$ and looks for it in the pre-computed table. Let us denote $|X|$ as the number of bits in X; this strategy will succeed if $|K_1| < |K|$ and $|K_2| < |K|$. Nevertheless, for a block cipher with a good key schedule, these would be satisfied only for a very small number of rounds, which means that the number of rounds could be broken is limited.

Demirci et al. [7] went a step further when they applied the MITM attack to AES. They follow the similar idea in [10] and treated the cipher E as $E_K = E^2_{K_2} \circ E^m \circ E^1_{K_1}$. In their strategy, at first, the adversary builds a distinguisher in E^m: For the input of E^m, all values of one byte (denoted by x) are chosen, while the other bytes are fixed values. Then the output of E^m can be expressed as a function of x and some parameters that are determined by the fixed values of the input and the subkeys involved in E^m. If the total number of bits of the parameters is small enough (e.g. less than $|K| - 8$), then the distinguisher will work, and the adversary pre-computes the output (usually one byte or a function of the output) sequence of $E^m(x)$ by the expression of the parameters and stores them in a hash table where in each row the values are sorted according to the order of x. With the distinguisher, the adversary can mount an attack by guessing the K_1, choosing suitable plaintexts (thus obtaining the corresponding ciphertexts) and partially decrypting the ciphertexts (guessing K_2) to check whether the sequence $E^{2^{-1}}_{K_2}(C)(x)$ can match a row in the pre-computed table. By this means, some wrong (K_1, K_2) pairs will be discarded while the right one will be kept.

Our Contribution. In this paper, we apply the attacking model in [7] to Camellia-256. In our MITM distinguisher, we express one bit of a function of $E^m(x)$ by some 8-bit parameters and 1-bit parameters. For the byte-oriented ciphers, the expressions of one bit and one byte usually need the same number of parameters; however, the scenario for Camellia is different due to the keyed linear functions FL and FL^{-1}. Our distinguisher could be seen as an improvement of the 6-round higher-order MITM distinguisher (with the FL layer between the 4th and 5th rounds) of Camellia-256 in [20]; the 6-round higher-order MITM distinguisher is hard to be extended to a 7-round one as the four round integral property is difficult to expand to 5 rounds, see Sect. 3.1 for the detailed explanation. We get rid of the integral property and built a standard MITM distinguisher with the FL layer between Round 5 and Round 6 by one-bit expression to reduce as many parameters as possible. Due to the improved

Table 1. Summary of the Attacks on Camellia-256 with FL layers

#Rounds	Method	Data	Time	Memory	Source
11♯	Higher Order DC	2^{93}CC	$2^{255.6}$	2^{98}	[11]
11	Impossible DC	2^{121}CP	$2^{206.8}$	2^{166}	[5]
11	Impossible DC	$2^{119.6}$CP	$2^{194.5}$	2^{135}	[16]
12 (Weak Key)	Impossible DC	$2^{121.12}$CP	$2^{202.55}$	$2^{142.12}$	[17]
12	Impossible DC	$2^{116.17}$CP/CC	2^{240}	$2^{150.17}$	[17]
12†	Higher-Order MITM	2^{94}CP	$2^{237.3}$	2^{174}	[20]
12	**MITM**	2^{19}**CP**	$2^{231.2}$	2^{229}	**this paper**
13‡	Impossible DC	2^{123}CP	$2^{251.1}$	2^{208}	[17]
14†‡	Impossible DC	2^{120}CC	$2^{250.5}$	2^{131}	[17]
14†‡	Impossible DC	$2^{121.2}$CP	$2^{238.2}$	$2^{180.2}$	[3]

DC: differential cryptanalysis; CP: chosen plaintext; CC: chosen ciphertext
♯: this attack is mounted on the last 11 rounds
†: this attack excludes the whitening keys
‡: this is an attack on irregular Camellia-256

distinguisher, we can mount an attack on 12-round Camellia-256 with only 2^{19} chosen plaintexts. The time complexity of our attack is $2^{231.2}$, which is also the best for regular Camellia-256. We summarize our result along with some major previous results of Camellia-256 with FL layers in Table 1. The rest of this paper is organized as follows. Section 2 gives some notations and a brief description of Camellia-256. Our 7-round MITM distinguisher and the attack on Camellia-256 reduced to 12 rounds are proposed in Sect. 3 and Sect. 4, respectively. Finally, we conclude the paper in Sect. 5.

2 Preliminaries

This section gives some notations used through out the paper and a brief description of Camellia-256.

2.1 Notations

L^{r-1} : the left half of the 128-bit r-th round input
R^{r-1} : the right half of the 128-bit r-th round input
A_i: the i-th byte of a 64-bit value A $(i = 1, ..., 8)$
$B \lll j$: left rotation of B by j bits
$C_{(t)}$: the t-th bit of C, where C can be a 64-bit value (here $t = 0, ..., 63$), a 32-bit value (here $t = 0, ..., 31$) or a 8-bit value (here $t = 0, ..., 7$). The big-endian ordering is used, hence the 0th bit is the most significant bit.
$C_{(i-j)}$: the i-th bit to the j-th bit of C $(0 \leq i < j)$
$X_{L<64>}$: the left half of a 128-bit word X
$X_{R<64>}$: the right half of a 128-bit word X
$Y_{L<32>}$: the left half of a 64-bit word Y
$Y_{R<32>}$: the right half of a 64-bit word Y

$||$: the cascade of two words
$\overline{x}$: the bitwise complement of x
$\oplus$, $\cap$, $\cup$: bitwise exclusive-OR(XOR), AND, OR

2.2 Brief Description of Camellia

Camellia [1] is a 128-bit block cipher with Feistel structure. It has 18 rounds for 128-bit key, and 24 rounds for 192-/256-bit key. We give the encryption procedure of Camellia-256 as follows, see Fig. 1.

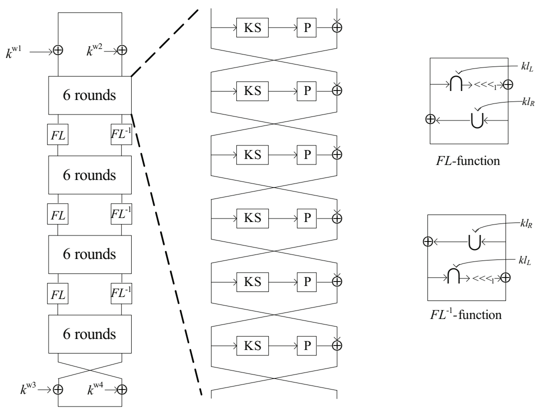


Fig. 1. Camellia-256

Encryption Procedure. The input of the encryption procedure is a 128-bit plaintext M, and 64-bit subkeys k^{wi} $(i = 1, ..., 4)$, k^r $(r = 1, ..., 24)$ and kl^j $(j = 1, ..., 6)$. First M is XORed with k^{w1} and k^{w2} to get two 64-bit intermediate values L^0 and R^0: $L^0||R^0 = M \oplus (k^{w1}||k^{w2})$. Then the following operations are carried out for $i = 1$ to 24, except for $r = 6$, 12 and 18:

$$L^r = R^{r-1} \oplus F(L^{r-1}, k^r), R^r = L^{r-1}.$$

For $r = 6$, 12 and 18, do the following:

$$L^{*r} = R^{r-1} \oplus F(L^{r-1}, k^r), R^{*r} = L^{r-1}.$$
$$L^r = FL(L^{*r}, kl^{2r/6-1}), R^r = FL^{-1}(R^{*r}, kl^{2r/6}).$$

Finally the 128-bit ciphertext C is computed as: $C = (R^{24}||L^{24}) \oplus (k^{w3}||k^{w4})$.

The FL function is defined as: $(X_{L<32>}||X_{R<32>}, kl_{L<32>}||kl_{R<32>}) \mapsto (Y_{L<32>}||Y_{R<32>})$, where:

$$Y_{R<32>} = ((X_{L<32>} \cap kl_{L<32>}) \lll 1) \oplus X_{R<32>},$$
$$Y_{L<32>} = (Y_{R<32>} \cup kl_{R<32>}) \oplus X_{L<32>}.$$

The FL^{-1} function is the inverse of FL function, and FL and FL^{-1} are linear as long as the keys are fixed [2].

The round function F is composed of the key-addition layer, S-box layer S and linear transformation P. In the key-addition layer, the input of the round function is XORed with the subkey. There are four 8×8 S-boxes S_1, S_2, S_3, S_4 used in the S-box layer, and each S-box is used twice. Finally, the linear transformation $P : (\{0,1\}^8)^8 \rightarrow (\{0,1\}^8)^8$ maps $(z_1, ..., z_8) \rightarrow (y_1, ..., y_8)$. P function and its inverse function P^{-1} are:

$$\begin{array}{ll}
y_1 = z_1 \oplus z_3 \oplus z_4 \oplus z_6 \oplus z_7 \oplus z_8 & z_1 = y_2 \oplus y_3 \oplus y_4 \oplus y_6 \oplus y_7 \oplus y_8 \\
y_2 = z_1 \oplus z_2 \oplus z_4 \oplus z_5 \oplus z_7 \oplus z_8 & z_2 = y_1 \oplus y_3 \oplus y_4 \oplus y_5 \oplus y_7 \oplus y_8 \\
y_3 = z_1 \oplus z_2 \oplus z_3 \oplus z_5 \oplus z_6 \oplus z_8 & z_3 = y_1 \oplus y_2 \oplus y_4 \oplus y_5 \oplus y_6 \oplus y_8 \\
y_4 = z_2 \oplus z_3 \oplus z_4 \oplus z_5 \oplus z_6 \oplus z_7 & z_4 = y_1 \oplus y_2 \oplus y_3 \oplus y_5 \oplus y_6 \oplus y_7 \\
y_5 = z_1 \oplus z_2 \oplus z_6 \oplus z_7 \oplus z_8 & z_5 = y_1 \oplus y_2 \oplus y_5 \oplus y_7 \oplus y_8 \\
y_6 = z_2 \oplus z_3 \oplus z_5 \oplus z_7 \oplus z_8 & z_6 = y_2 \oplus y_3 \oplus y_5 \oplus y_6 \oplus y_8 \\
y_7 = z_3 \oplus z_4 \oplus z_5 \oplus z_6 \oplus z_8 & z_7 = y_3 \oplus y_4 \oplus y_5 \oplus y_6 \oplus y_7 \\
y_8 = z_1 \oplus z_4 \oplus z_5 \oplus z_6 \oplus z_7 & z_8 = y_1 \oplus y_4 \oplus y_6 \oplus y_7 \oplus y_8
\end{array}$$

Key Schedule. For Camellia-256, the 256-bit main key $K = K_L || K_R$, where K_L and K_R are 128 bits. Using K_L and K_R, the key schedule algorithm first calculate K_A and K_B, which is described in Fig. 2. Where F is the round function of Camellia and C_i $(1 \leq i \leq 6)$ are constants used as the keys. Then the subkeys k^{wi} $(i = 1, ..., 4)$, k^r $(r = 1, ..., 24)$ and kl^j $(j = 1, ..., 6)$ are derived from rotating K_L, K_R, K_A or K_B. For details of Camellia, we refer to [1].

It can be known from Fig. 2 that, if K_B and K_A are known, then K_R is known. Therefore, one can get K_L using the relation between K_L and K_A described in Sect. 3.2 of [21]. So once K_B and K_A are known, K can be computed.

3 7-Round Meet-in-the-Middle Distinguisher with *FL* Layer

In this section, we first briefly reintroduce the higher-order MITM distinguishers in [20], and then propose our improved distinguisher.

3.1 Lu et al.'s Higher-Order MITM Distinguishers

In [20], Lu et al.'s introduce 5-round and 6-round distinguishers with FL layers. They call their distinguishers higher-order MITM distinguishers because they use the integral property of 3-/4-round Camellia [30] with FL layers to cancel some parameters; we use the 6-round distinguisher (see Fig. 5 in Appendix B) as an example to show how this can be achieved. In Fig. 5, 'C' stands for a constant byte, 'A' stands for an active byte (a byte that takes all 256 possible values) and 'B' stands for a balanced byte ($\bigoplus_{i=0}^{255}$B=0). We can see from Fig. 5 that if the input of the 6-round distinguisher is of the form

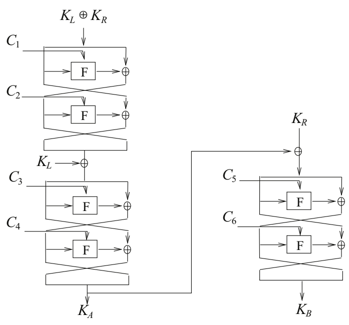


Fig. 2. The Calculation of K_A and K_B

(P_L=(C,C,C,C,C,C,C,C), P_R=(C,A,C,C,C,C,C,C)), then each byte of T is balanced, which leads to the balance of W (since FL^{-1} is linear for a specific key). Hence we have: $\bigoplus_{i=0}^{255}(P^{-1}(Z))_j = \bigoplus_{i=0}^{255} Y_j \oplus (P^{-1}(W))_j = \bigoplus_{i=0}^{255} Y_j$, for $j = 1, ..., 8$. As a result, several parameters can be canceled by the integral property when $\bigoplus_{i=0}^{255}(P^{-1}(Z))_j$ is expressed. See [20] for detail. However, if there are 5 rounds before the FL layer, the integral property would disappear, hence one cannot get a 7-round higher-order MITM distinguisher by extending one round forward. Adding one more round at the bottom of the 6-round higher-order MITM distinguisher is also infeasible, since the required parameters is too many. Instead, we remove the integral property and attempt to build a valid 7-round MITM distinguisher.

3.2 7-Round Meet-in-the-Middle Distinguisher

As claimed by Lu et al. in [20], 12 more 8-bit parameters are required if one uses an MITM technique to build the 6-round distinguisher. It is true if one aims to express one byte of $P^{-1}(Z) = Y \oplus P^{-1}(W)$; but if we only want to know the expression of one bit of $P^{-1}(Z)$, then fewer parameters are needed. Furthermore, we can even add one more round to the top of the 6-round MITM distinguisher and get a 7-round one, provided that we are looking for the expression of only one bit. See the observation below:

Observation 1. *Let the input to 7-round Camellia with an FL layer between Round 5 and Round 6 be: $L^0 = (\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7, \alpha_8)$, $R^0 = (x, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6, \beta_7, \beta_8)$, where x take all 2^8 values and α_i ($i = 1, ..., 8$), β_i ($i = 2, ..., 8$) are constants (see Fig. 3). Denote $Z = R^7$ and $Z(t)$ as the value of Z when $x = t$, then $(P^{-1}(Z(0)))_{6(0)} \oplus (P^{-1}(Z(x)))_{6(0)}$ (for $x = 1, ..., 255$) is a function of x fully determined by 224-bit fixed parameters, i.e., 26 8-bit parameters $(b_1, d_1, d_2, d_3, d_5, d_8,\ e_1, ..., e_8,\ f_1, ..., f_8, K_{L1(9-16)},\ e'_2, e'_3, g)$ and 16 1-bit*

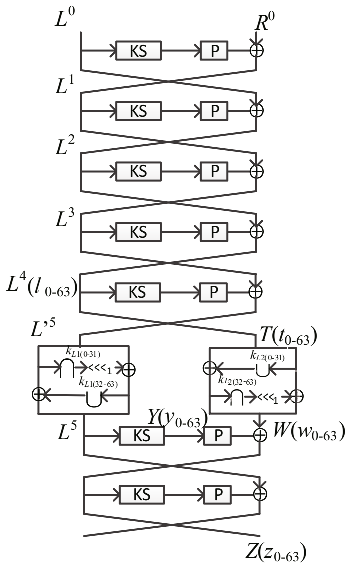


Fig. 3. 7-Round MITM Distinguisher of Camellia-256

parameters $(K_{L2(1)}, K_{L2(8)}, K_{L2(9)}, K_{L2(16)}, K_{L2(25)}, K_{L2(33)}, K_{L2(41)}, K_{L2(57)}, c_{1(1)}, c_{2(1)}, c_{4(1)}, c_{5(1)}, c_{6(0)}, c_{6(1)}, c_{7(0)}, c_{8(1)})$.

The proof of Observation 1 can be found in Appendix A.

We know that there are 2^{255} possible values for a 255-bit sequence, hence a 7-round MITM distinguisher is acquired for Camellia-256 from Observation 1.

4 Meet-in-the-Middle Attack on 12-Round Camellia-256 with Low Data Complexity

In this section, an attack on 12-round Camellia-256 is proposed; as mentioned, compared with the best previous attack on regular Camellia-256, the data complexity of our attack is much lower, while the time complexity is also better. When attacking the first 12 rounds of Camellia-256, we add one round before the 7-round distinguisher and 4 rounds after, see Fig. 4. The procedure of the attack is depicted as follows:

Precomputation.A 7-round distinguisher is build according to Observation 1. For all values of 8-bit parameters $(b_1, d_1, d_2, d_3, d_5, d_8, e_1, ..., e_8, f_1, ..., f_8, K_{L1(9-16)}, e'_2, e'_3, g)$ and 1-bit parameters $(K_{L2(1)}, K_{L2(8)}, K_{L2(9)}, K_{L2(16)}, K_{L2(25)}, K_{L2(33)}, K_{L2(41)}, K_{L2(57)}, c_{1(1)}, c_{2(1)}, c_{4(1)}, c_{5(1)}, c_{6(0)}, c_{6(1)}, c_{7(0)}, c_{8(1)})$, calculate $(P^{-1}(Z(0)))_{6(0)} \oplus (P^{-1}(Z(x)))_{6(0)}$ (for $x = 1, ..., 255$) and insert them into a hash table H. In each row of the table, $(P^{-1}(Z(0)))_{6(0)} \oplus (P^{-1}(Z(x)))_{6(0)}$ are sorted according to the value of x; the size of Table H is 2^{229} bytes. The time complexity

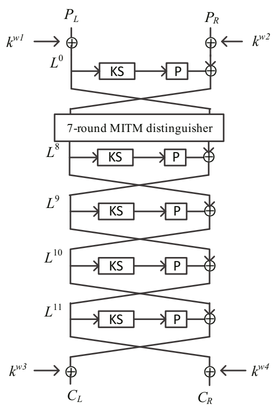


Fig. 4. 12-Round MITM Attack on Camellia-256

of the precomputation is less than 2^{232} 7-round encryptions, equivalent to $2^{231.2}$ 12-round encryptions.

Data Collection. Choose 2^3 structures of plaintexts; in each structure, guess (k_1^{w1}, k_1^1) and compute the plaintexts with the form $P_L = (x \oplus k^{w1}, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6, \gamma_7, \gamma_8)$ and $P_R = (S(x \oplus k_1^1) \oplus \eta_1, S(x \oplus k_1^1) \oplus \eta_2, S(x \oplus k_1^1) \oplus \eta_3, \eta_4, S(x \oplus k_1^1) \oplus \eta_5, \eta_6, \eta_7, S(x \oplus k_1^1) \oplus \eta_8)$. Where x takes all 2^8 values, while $\gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6, \gamma_7, \gamma_8, \eta_1, \eta_2, \eta_3, \eta_4, \eta_5, \eta_6, \eta_7, \eta_8$ are randomly chosen constants. Encrypt the plaintexts in each structure to get corresponding ciphertexts.

Key Recovery. In order to deal with the whitening keys, we first denote some equivalent keys of Round 9 to Round 12: $k^a = k^{12} \oplus k^{w4}$, $k^b = k^{11} \oplus k^{w3}$, $k^c = k^{10} \oplus k^{w4}$, $k^d = k^9 \oplus k^{w3}$ and $k^e = k^{w4}_{2(0)} \oplus k^{w4}_{3(0)} \oplus k^{w4}_{5(0)} \oplus k^{w4}_{6(0)} \oplus k^{w4}_{8(0)}$.

For each (k_1^{w1}, k_1^1) and each structure obtained from the data collection phase, guess k^a, k^b, k_2^c, k_3^c, k_5^c, k_6^c, k_8^c, k_6^d and k^e. Under each key guess, we calculate L_1^0 (the value of x) and $(P^{-1}(L^7(x)))_{6(0)} = (S(L_6^8 \oplus k_6^d))_{(0)} \oplus (P^{-1}(L^9))_{6(0)}$ for all the 256 plaintext-ciphertext pairs in the structure.

If for all the structures, the corresponding sequences $(P^{-1}(L^7(0)))_{6(0)} \oplus (P^{-1}(L^7(x)))_{6(0)}$ (for $x = 1, ..., 255$) are in H, then we say that the guessed key is the right key; otherwise, we discard the wrong key and try another one. Since for a wrong key, the probability that one can find the sequence $(P^{-1}(L^7(0)))_{6(0)} \oplus (P^{-1}(L^7(x)))_{6(0)}$ (for $x = 1, ..., 255$) in a structure to be consistent with a specific row of H is 2^{-255}, and there are 2^{224} rows in H, the probability that all sequences in the structures can be found in H is $(2^{224-255})^8 = 2^{-248}$. As a consequence, the expected number of remaining wrong 193-bit keys $(k_1^{wl}, k^1, k^a, k^b, k_2^c, k_3^c, k_5^c, k_6^c, k_8^c, k_6^d, k^e)$ is $2^{193} \times 2^{-248} = 2^{-55}$; the one that is kept is supposed to be the right one.

Note that $k^{11} = (K_A \lll 45)_{L<64>}$, $k^{12} = (K_A \lll 45)_{R<64>}$, $k^{w3} = (K_B \lll 111)_{L<64>}$ and $k^{w4} = (K_B \lll 111)_{R<64>}$; if we guess K_A, then K_B can also be deduced from the right key we obtained. From K_A and K_B, the main key can be calculated as described in Sect. 2. Then we test the main key by trial encryptions. Consequently, the complexity for recovering the main key from the equivalent keys is about 2^{128}.

From the discussion above, one can know that the complexity of the key recovery phase is about $2^{193} \times 2^8 \times 2^3 = 2^{204}$ 4-round encryptions; the time and memory complexities of the whole attack are dominated by the precomputation phase. The data complexity is 2^{19} chosen plaintexts.

5 Conclusion

This paper proposes an MITM attack on 12-round Camellia-256 with much lower data complexity than the previous attacks. It is also the best attack on Camellia-256 in terms of time complexity and the number of rounds if we consider the 'regular' Camellia which has 6 rounds before the first FL layer (in a chosen plaintext scenario) or 6 rounds after the last FL layer (in a chosen ciphertext scenario). Note that our attack does not harm the security of Camellia-256, since it is an analysis on reduced Camellia-256 and impractical.

Acknowledgements. We are grateful to the anonymous reviewers for their valuable comments on this paper.

References

1. Aoki, K., Ichikawa, T., Kanda, M., Matsui, M., Moriai, S., Nakajima, J., Tokita, T.: *Camellia*: A 128-Bit Block Cipher Suitable for Multiple Platforms - Design and Analysis. In: Stinson, D.R., Tavares, S. (eds.) SAC 2000. LNCS, vol. 2012, pp. 39–56. Springer, Heidelberg (2001)
2. Aoki, K., Ichikawa, T., Kanda, M., Matsui, M., Moriai, S., Nakajima, J., Tokita, T.: Specification of Camellia-a 128-bit Block Cipher. version 2.0 (2001), `http://info.isl.ntt.co.jp/crypt/eng/camellia/specifications.html`
3. Bai, D., Li, L.: New Impossible Differential Attacks on Camellia. IACR Cryptology ePrint Archive 2011, 661 (2011)
4. Biham, E., Biryukov, A., Shamir, A.: Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 12–23. Springer, Heidelberg (1999)
5. Chen, J., Jia, K., Yu, H., Wang, X.: New Impossible Differential Attacks of Reduced-Round Camellia-192 and Camellia-256. In: Parampalli, U., Hawkes, P. (eds.) ACISP 2011. LNCS, vol. 6812, pp. 16–33. Springer, Heidelberg (2011)
6. CRYPTREC-Cryptography Research and Evaluation Committees, report, Archive (2002), `http://www.cryptrec.go.jp/english/index.html`
7. Demirci, H., Selçuk, A.A.: A Meet-in-the-Middle Attack on 8-Round AES. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 116–126. Springer, Heidelberg (2008)
8. Diffie, W., Hellman, M.: Special Feature Exhaustive Cryptanalysis of the NBS Data Encryption Standard. Computer 10, 74–84 (1977)
9. Duo, L., Li, C., Feng, K.: Square Like Attack on Camellia. In: Qing, S., Imai, H., Wang, G. (eds.) ICICS 2007. LNCS, vol. 4861, pp. 269–283. Springer, Heidelberg (2007)

10. Gilbert, H., Minier, M.: A Collision Attack on 7 Rounds of Rijndael. In: AES Candidate Conference, pp. 230–241 (2000)
11. Hatano, Y., Sekine, H., Kaneko, T.: Higher Order Differential Attack of Camellia (II). In: Nyberg, K., Heys, H.M. (eds.) SAC 2002. LNCS, vol. 2595, pp. 129–146. Springer, Heidelberg (2003)
12. International Standardization of Organization (ISO), International Standard-ISO/IEC 18033-3, Information technology-Security techniques-Encryption algorithms -Part 3: Block ciphers (2005)
13. Knudsen, L.: DEAL - A 128-bit Block Cipher. In: NIST AES Proposal (1998)
14. Lee, S., Hong, S.H., Lee, S.-J., Lim, J.-I., Yoon, S.H.: Truncated Differential Cryptanalysis of Camellia. In: Kim, K.-c. (ed.) ICISC 2001. LNCS, vol. 2288, pp. 32–38. Springer, Heidelberg (2002)
15. Lei, D., Chao, L., Feng, K.: New Observation on Camellia. In: Preneel, B., Tavares, S. (eds.) SAC 2005. LNCS, vol. 3897, pp. 51–64. Springer, Heidelberg (2006)
16. Li, L., Chen, J., Jia, K.: New Impossible Differential Cryptanalysis of Reduced-Round Camellia. In: Lin, D., Tsudik, G., Wang, X. (eds.) CANS 2011. LNCS, vol. 7092, pp. 26–39. Springer, Heidelberg (2011)
17. Liu, Y., Li, L., Gu, D., Wang, X., Liu, Z., Chen, J., Li, W.: New Observations on Impossible Differential Cryptanalysis of Reduced-Round Camellia. To appear at FSE 2012 (2012)
18. Lu, J., Kim, J.-S., Keller, N., Dunkelman, O.: Improving the Efficiency of Impossible Differential Cryptanalysis of Reduced Camellia and MISTY1. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 370–386. Springer, Heidelberg (2008)
19. Lu, J., Wei, Y., Kim, J., Fouque, P.A.: Cryptanalysis of Reduced Versions of the Camellia Block Cipher, http://sac2011.ryerson.ca/SAC2011/LWKF.pdf
20. Lu, J., Wei, Y., Kim, J., Pasalic, E.: The Higher-Order Meet-in-the-Middle Attack and Its Application to the Camellia Block Cipher. Presented in part at the First Asian Workshop on Symmetric Key Cryptography (ASK 2011), Singapore (August 2011), https://sites.google.com/site/jiqiang/HO-MitM.pdf
21. Mala, H., Shakiba, M., Dakhilalian, M., Bagherikaram, G.: New Results on Impossible Differential Cryptanalysis of Reduced–Round Camellia–128. In: Jacobson Jr., M.J., Rijmen, V., Safavi-Naini, R. (eds.) SAC 2009. LNCS, vol. 5867, pp. 281–294. Springer, Heidelberg (2009)
22. Mozilla: Camellia cipher added to Firefox. Mozilla in Asia (2009)
23. NESSIE-New European Schemes for Signatures, Integrity, and Encryption, final report of European project IST-1999-12324. Archive (1999), https://www.cosic.esat.kuleuven.be/nessie/Bookv015.pdf
24. NTT: The Open Source Community OpenSSL Project Adopts the Next Generation International Standard Cipher "Camellia" Developed in Japan (2008)
25. Shirai, T.: Differential, Linear, Boomerang and Rectangle Cryptanalysis of Reduced-Round Camellia. In: Proceedings of the Third NESSIE Workshop, Munich, Germany, November 6-7 (2002)
26. Sugita, M., Kobara, K., Imai, H.: Security of Reduced Version of the Block Cipher Camellia against Truncated and Impossible Differential Cryptanalysis. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 193–207. Springer, Heidelberg (2001)
27. Wu, W., Zhang, W., Feng, D.: Impossible Differential Cryptanalysis of Reduced-Round ARIA and Camellia. Journal of Computer Science and Technology 22(3), 449–456 (2007)

28. Wenling, W., Dengguo, F., Hua, C.: Collision Attack and Pseudorandomness of Reduced-Round Camellia. In: Handschuh, H., Hasan, M.A. (eds.) SAC 2004. LNCS, vol. 3357, pp. 252–266. Springer, Heidelberg (2004)
29. Wu, W., Zhang, L., Zhang, W.: Improved Impossible Differential Cryptanalysis of Reduced-Round Camellia. In: Avanzi, R., Keliher, L., Sica, F. (eds.) SAC 2008. LNCS, vol. 5381, pp. 442–456. Springer, Heidelberg (2009)
30. Yeom, Y., Park, S., Kim, I.: A Study of Integral Type Cryptanalysis on Camellia. In: Proceedings of the 2003 Symposium on Cryptography and Information Security, pp. 453-456 (2003)

A Proof of Observation 1

Observation 1 benefits from the facts that FL^{-1} is keyed linear and P is linear. For simplicity, let us denote the subkeys used in the FL and FL^{-1} functions as K_{L1} and K_{L2}, those used in the seven rounds as $K^1, ..., K^7$ and the cascade of the key-addition and S-box layer as SK. We will also allow ourself to abuse the notation S as the application of the S-box layer to the intermediate values and the application of an S-box to a byte (we will not specify which one of the 4 S-boxes is being used as well).

From the input we have: $P(SK(L^0)) = (a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8)$, where $a_1, ..., a_8$ are determined by $\alpha_1, ..., \alpha_8$ and K^1. Then $L^1 = (x \oplus a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8)$, $SK(L^1) = (S(x \oplus b_1), b_2, b_3, b_4, b_5, b_6, b_7, b_8)$, where $b_1 = a_1 \oplus K_1^2$, $b_i = S(a_i \oplus K_i^2)$ $(i = 2, ..., 8)$.

Similarly we have the following equations.
$L_1^2 = S(x \oplus b_1) \oplus c_1$, $SK(L_1^2) = S(S(x \oplus b_1) \oplus d_1)$,
$L_2^2 = S(x \oplus b_1) \oplus c_2$, $SK(L_2^2) = S(S(x \oplus b_1) \oplus d_2)$,
$L_3^2 = S(x \oplus b_1) \oplus c_3$, $SK(L_3^2) = S(S(x \oplus b_1) \oplus d_3)$,
$L_4^2 = c_4$, $SK(L_4^2) = d_4$,
$L_5^2 = S(x \oplus b_1) \oplus c_5$, $SK(L_5^2) = S(S(x \oplus b_1) \oplus d_5)$,
$L_6^2 = c_6$, $SK(L_6^2) = d_6$,
$L_7^2 = c_7$, $SK(L_7^2) = d_7$,
$L_8^2 = S(x \oplus b_1) \oplus c_8$, $SK(L_8^2) = S(S(x \oplus b_1) \oplus d_8)$.

Since $L^3 = P(SK(L^2)) \oplus L^1$,
$SK(L_1^3) = S(S(S(x \oplus b_1) \oplus d_1) \oplus S(S(x \oplus b_1) \oplus d_3) \oplus S(S(x \oplus b_1) \oplus d_8) \oplus x \oplus e_1)$,
$SK(L_2^3) = S(S(S(x \oplus b_1) \oplus d_1) \oplus S(S(x \oplus b_1) \oplus d_2) \oplus S(S(x \oplus b_1) \oplus d_5) \oplus S(S(x \oplus b_1) \oplus d_8) \oplus e_2)$,
$SK(L_3^3) = S(S(S(x \oplus b_1) \oplus d_1) \oplus S(S(x \oplus b_1) \oplus d_2) \oplus S(S(x \oplus b_1) \oplus d_3) \oplus S(S(x \oplus b_1) \oplus d_5) \oplus S(S(x \oplus b_1) \oplus d_8) \oplus e_3)$,
$SK(L_4^3) = S(S(S(x \oplus b_1) \oplus d_2) \oplus S(S(x \oplus b_1) \oplus d_3) \oplus S(S(x \oplus b_1) \oplus d_5) \oplus e_4)$,
$SK(L_5^3) = S(S(S(x \oplus b_1) \oplus d_1) \oplus S(S(x \oplus b_1) \oplus d_2) \oplus S(S(x \oplus b_1) \oplus d_8) \oplus e_5)$,
$SK(L_6^3) = S(S(S(x \oplus b_1) \oplus d_2) \oplus S(S(x \oplus b_1) \oplus d_3) \oplus S(S(x \oplus b_1) \oplus d_5) \oplus S(S(x \oplus b_1) \oplus d_8) \oplus e_6)$,
$SK(L_7^3) = S(S(S(x \oplus b_1) \oplus d_3) \oplus S(S(x \oplus b_1) \oplus S(y \oplus b_2) \oplus d_5) \oplus S(S(x \oplus b_1) \oplus d_8) \oplus e_7)$,
$SK(L_8^3) = S(S(S(x \oplus b_1) \oplus d_1) \oplus S(S(x \oplus b_1) \oplus d_5) \oplus e_8)$,

From $L^4 = P(SK(L^3)) \oplus L^2$,
$L^4_1 = SK(L^3_1) \oplus SK(L^3_3) \oplus SK(L^3_4) \oplus SK(L^3_6) \oplus SK(L^3_7) \oplus SK(L^3_8) \oplus S(x \oplus b_1) \oplus c_1$,
$L^4_2 = SK(L^3_1) \oplus SK(L^3_2) \oplus SK(L^3_4) \oplus SK(L^3_5) \oplus SK(L^3_7) \oplus SK(L^3_8) \oplus S(x \oplus b_1) \oplus c_2$,
$L^4_3 = SK(L^3_1) \oplus SK(L^3_2) \oplus SK(L^3_3) \oplus SK(L^3_5) \oplus SK(L^3_6) \oplus SK(L^3_8) \oplus S(x \oplus b_1) \oplus c_3$,
$L^4_4 = SK(L^3_2) \oplus SK(L^3_3) \oplus SK(L^3_4) \oplus SK(L^3_5) \oplus SK(L^3_6) \oplus SK(L^3_7) \oplus c_4$,
$L^4_5 = SK(L^3_1) \oplus SK(L^3_2) \oplus SK(L^3_6) \oplus SK(L^3_7) \oplus SK(L^3_8) \oplus S(x \oplus b_1) \oplus c_5$,
$L^4_6 = SK(L^3_2) \oplus SK(L^3_3) \oplus SK(L^3_4) \oplus SK(L^3_7) \oplus SK(L^3_8) \oplus c_6$,
$L^4_7 = SK(L^3_3) \oplus SK(L^3_4) \oplus SK(L^3_5) \oplus SK(L^3_6) \oplus SK(L^3_8) \oplus c_7$,
$L^4_8 = SK(L^3_1) \oplus SK(L^3_4) \oplus SK(L^3_5) \oplus SK(L^3_6) \oplus SK(L^3_7) \oplus S(x \oplus b_1) \oplus c_8$,

$SK(L^4_1) = S(SK(L^3_1) \oplus SK(L^3_3) \oplus SK(L^3_4) \oplus SK(L^3_6) \oplus SK(L^3_7) \oplus SK(L^3_8) \oplus S(x \oplus b_1) \oplus f_1)$,
$SK(L^4_2) = S(SK(L^3_1) \oplus SK(L^3_2) \oplus SK(L^3_4) \oplus SK(L^3_5) \oplus SK(L^3_7) \oplus SK(L^3_8) \oplus S(x \oplus b_1) \oplus f_2)$,
$SK(L^4_3) = S(SK(L^3_1) \oplus SK(L^3_2) \oplus SK(L^3_3) \oplus SK(L^3_5) \oplus SK(L^3_6) \oplus SK(L^3_8) \oplus S(x \oplus b_1) \oplus f_3)$,
$SK(L^4_4) = S(SK(L^3_2) \oplus SK(L^3_3) \oplus SK(L^3_4) \oplus SK(L^3_5) \oplus SK(L^3_6) \oplus SK(L^3_7) \oplus f_4)$,
$SK(L^4_5) = S(SK(L^3_1) \oplus SK(L^3_2) \oplus SK(L^3_6) \oplus SK(L^3_7) \oplus SK(L^3_8) \oplus S(x \oplus b_1) \oplus f_5)$,
$SK(L^4_6) = S(SK(L^3_2) \oplus SK(L^3_3) \oplus SK(L^3_4) \oplus SK(L^3_7) \oplus SK(L^3_8) \oplus f_6)$,
$SK(L^4_7) = S(SK(L^3_3) \oplus SK(L^3_4) \oplus SK(L^3_5) \oplus SK(L^3_6) \oplus SK(L^3_8) \oplus f_7)$,
$SK(L^4_8) = S(SK(L^3_1) \oplus SK(L^3_4) \oplus SK(L^3_5) \oplus SK(L^3_6) \oplus SK(L^3_7) \oplus S(x \oplus b_1) \oplus f_8)$,

We know that $L^{*5} = P(SK(L^4)) \oplus L^3$, hence
$L^{*5}_2 = SK(L^4_1) \oplus SK(L^4_2) \oplus SK(L^4_4) \oplus SK(L^4_5) \oplus SK(L^4_7) \oplus SK(L^4_8) \oplus S(S(x \oplus b_1) \oplus d_1) \oplus S(S(x \oplus b_1) \oplus d_2) \oplus S(S(x \oplus b_1) \oplus d_5) \oplus S(S(x \oplus b_1) \oplus d_8) \oplus e'_2$,
$L^{*5}_3 = SK(L^4_1) \oplus SK(L^4_2) \oplus SK(L^4_3) \oplus SK(L^4_5) \oplus SK(L^4_6) \oplus SK(L^4_8) \oplus S(S(x \oplus b_1) \oplus d_1) \oplus S(S(x \oplus b_1) \oplus d_2) \oplus S(S(x \oplus b_1) \oplus d_3) \oplus S(S(x \oplus b_1) \oplus d_5) \oplus S(S(x \oplus b_1) \oplus d_8) \oplus e'_3$,
$L^{*5}_6 = SK(L^4_2) \oplus SK(L^4_3) \oplus SK(L^4_4) \oplus SK(L^4_7) \oplus SK(L^4_8) \oplus S(S(x \oplus b_1) \oplus d_2) \oplus S(S(x \oplus b_1) \oplus d_3) \oplus S(S(x \oplus b_1) \oplus d_5) \oplus S(S(x \oplus b_1) \oplus d_8) \oplus e'_6$,

After the FL function,
$L^5_6 = ((L^{*5}_{2(1-7)} || L^{*5}_{3(0)}) \cap K_{L1(9-16)}) \oplus L^{*5}_6 = ((L^{*5}_{2(1-7)} || L^{*5}_{3(0)}) \cap K_{L1(9-16)}) \oplus SK(L^4_2) \oplus SK(L^4_3) \oplus SK(L^4_4) \oplus SK(L^4_7) \oplus SK(L^4_8) \oplus S(S(x \oplus b_1) \oplus d_2) \oplus S(S(x \oplus b_1) \oplus d_3) \oplus S(S(x \oplus b_1) \oplus d_5) \oplus S(S(x \oplus b_1) \oplus d_8) \oplus e'_6$.

As a result,
$Y_6 = S(((L^{*5}_{2(1-7)} || L^{*5}_{3(0)}) \cap K_{L1(9-16)}) \oplus SK(L^4_2) \oplus SK(L^4_3) \oplus SK(L^4_4) \oplus SK(L^4_7) \oplus SK(L^4_8) \oplus S(S(x \oplus b_1) \oplus d_2) \oplus S(S(x \oplus b_1) \oplus d_3) \oplus S(S(x \oplus b_1) \oplus d_5) \oplus S(S(x \oplus b_1) \oplus d_8) \oplus g)$.

Hence Y_6 is a function of x that entirely determined by 8-bit parameters $(b_1, d_1, d_2, d_3, d_5, d_8, e_1, ..., e_8, f_1, ..., f_8, K_{L1(9-16)}, e'_2, e'_3, g)$.

Furthermore, $Y_6 = (P^{-1}(W \oplus Z))_6$, $Y_{6(0)} = (P^{-1}(W \oplus Z))_{6(0)}$, and $(P^{-1}(W))_{6(0)} = w_8 \oplus w_{16} \oplus w_{32} \oplus w_{40} \oplus w_{56}$. Here $w_8 = K_{L2(8)} l_{(40)} + l_{(8)} + l_{(40)} + K_{L2(8)}$, $w_{16} = K_{L2(16)} l_{(48)} + l_{(16)} + l_{(48)} + K_{L2(16)}$, $w_{32} = K_{L2(33)} K_{L2(1)} l_{(33)} + K_{L2(33)} l_{(1)} + K_{L2(33)} l_{(33)} + K_{L2(33)} K_{L2(1)} + l_{(32)}$, $w_{40} = K_{L2(41)} K_{L2(9)} l_{(41)} + K_{L2(41)} l_{(9)} + K_{L2(41)} l_{(41)} + K_{L2(41)} K_{L2(9)} + l_{(40)}$ and $w_{56} = K_{L2(57)} K_{L2(25)} l_{(57)} + K_{L2(57)} l_{(25)} + K_{L2(57)} l_{(57)} + K_{L2(57)} K_{L2(25)} + l_{(56)}$, hence

$(P^{-1}(Z))_{6(0)} = Y_{6(0)} \oplus K_{L2(33)}K_{L2(1)}l_{(33)} \oplus K_{L2(41)}K_{L2(9)}l_{(41)} \oplus K_{L2(57)}K_{L2(25)} l_{(57)} \oplus K_{L2(8)}l_{(40)} \oplus K_{L2(16)}l_{(48)} \oplus K_{L2(33)}l_{(1)} \oplus K_{L2(33)}l_{(33)} \oplus K_{L2(33)}K_{L2(1)} \oplus K_{L2(41)}l_{(9)} \oplus K_{L2(41)}l_{(41)} \oplus K_{L2(41)}K_{L2(9)} \oplus K_{L2(57)}l_{(25)} \oplus K_{L2(57)}l_{(57)} \oplus K_{L2(57)}K_{L2(25)} \oplus l_{(8)} \oplus l_{(16)} \oplus l_{(32)} \oplus l_{(48)} \oplus l_{(56)} \oplus K_{L2(8)} \oplus K_{L2(16)}$.
Where $l_{(i)} = L^4_{i/8+1(i\%8)}$ for $i = \{1, 8, 9, 16, 25, 32, 33, 40, 41, 48, 56, 57\}$. As a result, $(P^{-1}(Z))_{6(0)}$ can be expressed as function of x with 26 8-bit parameters $(b_1, d_1, d_2, d_3, d_5, d_8, e_1, ..., e_8, f_1, ..., f_8, K_{L1(9-16)}, e'_2, e'_3, g)$ and 20 1-bit parameters $(K_{L2(1)}, K_{L2(8)}, K_{L2(9)}, K_{L2(16)}, K_{L2(25)}, K_{L2(33)}, K_{L2(41)}, K_{L2(57)}, c_{1(1)}, c_{2(0)}, c_{3(0)}, c_{5(0)}, c_{8(0)}, c_{2(1)}, c_{4(1)}, c_{5(1)}, c_{6(0)}, c_{6(1)}, c_{7(0)}, c_{8(1)})$. Furthermore, $(P^{-1}(Z(0)))_{6(0)} \oplus (P^{-1}(Z(x)))_{6(0)}$ (for $x = 1, ..., 255$) can be determined by 26 8-bit parameters $(b_1, d_1, d_2, d_3, d_5, d_8, e_1, ..., e_8, f_1, ..., f_8, K_{L1(9-16)}, e'_2, e'_3, g)$ and 16 1-bit parameters $(K_{L2(1)}, K_{L2(8)}, K_{L2(9)}, K_{L2(16)}, K_{L2(25)}, K_{L2(33)}, K_{L2(41)}, K_{L2(57)}, c_{1(1)}, c_{2(1)}, c_{4(1)}, c_{5(1)}, c_{6(0)}, c_{6(1)}, c_{7(0)}, c_{8(1)})$. This is because the four bits of $(c_{2(0)}, c_{3(0)}, c_{5(0)}, c_{8(0)})$ will be eliminated when XORing $(P^{-1}(Z(0)))_{6(0)}$ with $(P^{-1}(Z(x)))_{6(0)}$ (for $x = 1, ..., 255$). $\square$

B Lu et al.'s 6-Round Distinguisher

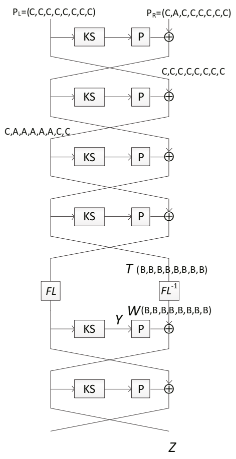


Fig. 5. 6-Round Higher-Order MITM Distinguisher from [20]

Cryptanalysis of RSA with a Small Parameter*

Xianmeng Meng[1] and Xuexin Zheng[2]

[1] School of Mathematics
Shandong University of Finance and Economics
Jinan, 250014, P.R. China
mxmeng@gmail.com

[2] Key Lab of Cryptologic Technology and Information Security
Ministry of Education, Shandong University
Jinan, 250100, P.R. China
zhxuexin@mail.sdu.edu.cn

Abstract. This paper investigates the security of RSA system with short exponents. Let $N = pq$ be an RSA modulus with balanced primes p and q. Denote the public exponent by e and the private exponent by d. Then e and d satisfy $ed - 1 = k\phi(N)$, which is usually called the RSA equation. When e and d are both short, and parameter k is the smallest unknown variable in RSA equation, we prove that there exist two new square root attacks. One attack applies the baby-step giant-step method, the other applies the Pollard's ρ method. We show that if K is a known upper bound of k, then k can be recovered in time $\tilde{O}(\sqrt{K})$ and memory $\tilde{O}(\sqrt{K})$ by using the baby-step giant-step method, and in time $\tilde{O}(\sqrt{K})$ and negligible memory by applying Pollard ρ method. As an application of our new attacks, we present the cryptanalysis on an RSA-type scheme proposed by Sun et al.

Keywords: RSA, square root attack, cryptanalysis.

1 Introduction

RSA scheme is the most famous and widely used public-key cryptosystem so far. It was proposed by Rivest, Shamir and Adleman [11] in 1978. Let $N = pq$ be RSA modulus. Usually, primes p and q are balanced with $q < p < 2q$. The public exponent e and private exponent d are chosen to be inverses of each other modulo $\varphi(N) = (p-1)(q-1)$, where $\varphi(N) = (p-1)(q-1)$ is Euler's totient function. The public key is then (N, e) and the secret key is (p, q, d). The security of RSA is based on the hardness of factorization.

To defend the factoring attack, usually RSA modulus N is chosen to be larger, e.g. $l_N = 1024$. Though it is hard to factor N, there are some other attacks as summarized in [8]. Among all the attacks, small private exponent attack is well-known. Using continued fractions, Wiener [22] described an attack, which applies

* This research is partially supported by the National Natural Science Foundation of China (Grant No. 61133013 and No. 60931160442) and the Technology Foundation of Ministry of Education of China (Grant No. 210123).

when the private exponent is smaller than $N^{0.25}$. Later, Boneh and Durfee [3,4] improved the result by applying Coppersmith's method [5] and showed that if the private exponent is less than $N^{0.292}$, then the RSA scheme can be broken in polynomial time. Weger [21] present extended attacks of Wiener [22] and Boneh-Durfee [3,4] on condition of small RSA prime difference. In [2], Blömer and May showed a generalized Wiener attack by applying the continued fraction method and Coppersmith's method.

All the above attacks show that choosing small exponents may cause the RSA scheme insecure. However, RSA is computationally complex, because of its requirement of exponentiation operations modulo a large integer N. The RSA encryption and decryption time is nearly proportional to the number of bits in the exponent. To lower the RSA decryption time, people attempt to design some RSA variants with short private exponent which can defend the private exponent attacks listed above. In [14,15], three RSA variants are given to get a secure private exponent shorter than the lower bound of Wiener [22] and Boneh-Durfee [3,4]. Unfortunately, the first and the third variants are broken by Durfee and Nyugen [7] and the second one is proved to be insecure if parameters are chosen careless.

In [16,17], Sun et al. improved the second variant of [14,15] and proposed two RSA schemes. One scheme has two balanced public and private exponents that can balance the encryption costs and decryption costs, and the other scheme can shift the work from decryptor to encryptor through changing the values of the public exponent and private exponent. By applying balanced primes, and these two schemes can defend the attacks of [7,2], etc. However, Sarkar and Maitra [12] presented a partial key exposure attack on RSA schemes of Sun et al. [16,17].

The former works [7,2,12] are all based on Coppersmith's method [5]. Here we investigate square root attacks against RSA. In RSA scheme, the public exponent e and private exponent d satisfy the following equation

$$ed - 1 = k\phi(N), \tag{1}$$

which is usually called the RSA equation. One can see that if e and d are short exponents, such as $l_e = 624$, $l_d = 512$ and usually $l_N = 1024$, then $l_k = 112$. In this case, k is the smallest parameters in (1). This observation leads us to recover k first, then p and q. Now we investigate two square root attacks to recover k. One attack applies the baby-step giant-step method, the other applies the Pollard's method. For these two methods, one can see [6,9] and for improved methods see [1,18,19,20]. We show that if K is a known upper bound of k such that $1 \le k < K$, there exist two probabilistic algorithms to recover k and then p and q in time $\tilde{O}(\sqrt{K})$. Here and in the sequel, $\tilde{O}()$ is the usual notation hiding polynomial arithmetic terms. As an application of the new attacks, we analyze the security of RSA schemes of Sun et al. [16,17] and find that parameter k should be chosen larger than the values suggested in [16,17].

Notation. We denote the bit-length of an integer u by l_u. Denote by $\lfloor r \rfloor$ the integral part of a real number r. Let $v \equiv u \bmod e$ denote that v is congruent to u modulo e. Let $v = u \bmod e$ denote that v equals the least non-negative remainder of u modulo e.

This paper is organized as follows. In section 2, we apply the baby-step giant-step method to present an attack. In section 3, we present an attack by apply Pollard's ρ method. Conclusions are finally drawn in section 4.

2 New Attack by Applying Shanks' Method

2.1 Baby-Step Giant-Step Method

This method is developed by Shanks [13]. Comparing to the exhaustive search method, it is a method of time/memory tradeoff. Usually, exhaustive search requires negligible memory and exponential time T. The baby-step giant-step method balances those two costs. Its time and space complexities both become roughly $\sqrt{T}$. Therefore it only works well for moderate sized T. The baby-step giant-step method has been applied to some problems such as the discrete logarithm problem, the factoring problem, etc. One can see [6] for details. Here we apply the baby-step giant-step method on RSA problem with short exponents, and obtain the following result.

Theorem 1. *Suppose $N = pq$ be an RSA modulus with primes p and q satisfying $q < p < 2q$. Let e and d be the public and private exponents satisfying (1). Let K be a known upper bound of k such that $1 < k < K$. If $e > k(p+q)$, then there exists a probabilistic algorithm to recover p and q in time $\tilde{O}(\sqrt{K})$ and space $\tilde{O}(\sqrt{K})$.*

Proof. By (1), we have

$$ed = 1 + k(N + 1 - p - q). \tag{2}$$

Dividing both sides of the above equality by e, we have

$$d = \frac{1 + k(N + 1 - p - q)}{e}. \tag{3}$$

Since d is an integer and $e > k(p + q)$, we have

$$d = \left\lfloor \frac{kN}{e} \right\rfloor. \tag{4}$$

Thus by (1) we have

$$k\phi(N) = e \left\lfloor \frac{kN}{e} \right\rfloor - 1. \tag{5}$$

Define

$$\Xi(x) = e \left\lfloor \frac{xN}{e} \right\rfloor - 1. \tag{6}$$

Let $M = \lceil\sqrt{K}\rceil$. Then parameter k can be written as $k = k_1M + k_2$ with $0 \leq k_1, k_2 < M$. Let a be an integer satisfying $\gcd(a, N) = 1$. By Euler's formula, we have

$$a^{\phi(N)} \equiv 1 \bmod N.$$

By (5) and (6), we have

$$a^{\Xi(k_1M+k_2)} \equiv 1 \bmod N. \tag{7}$$

Since

$$\left\lfloor \frac{(t_1+t_2)N}{e} \right\rfloor = \left\lfloor \frac{t_1N}{e} \right\rfloor + \left\lfloor \frac{t_2N}{e} \right\rfloor \text{ or } \left\lfloor \frac{t_1N}{e} \right\rfloor + \left\lfloor \frac{t_2N}{e} \right\rfloor + 1,$$

and by (6), we have that

$$\Xi(k_1M + k_2) = \Xi(k_1M) + \Xi(k_2) + 1 \text{ or } \Xi(k_1M) + \Xi(k_2) + e + 1.$$

In the following, for simplicity, we only suppose that

$$\Xi(k_1M + k_2) = \Xi(k_1M) + \Xi(k_2) + 1. \tag{8}$$

Then by (7) and (8), we have

$$a^{\Xi(k_1M)} \equiv a^{-\Xi(k_2)-1} \bmod N. \tag{9}$$

We will apply the above equality to find the right values of k_1 and k_2.

Note that k_1 and k_2 are in the range $0 \leq k_1, k_2 < M$, we construct two lists as follows,

$$L_1 = \left\{ a^{\Xi(k_1M)} \bmod N \;\middle|\; 0 \leq k_1 < M \right\},$$

and

$$L_2 = \left\{ a^{-\Xi(k_2)-1} \bmod N \;\middle|\; 0 \leq k_2 < M \right\}.$$

We can now build a table containing the M values of L_1. Then for each value of L_2 we check if it presents in the table. Finally, we find a match

$$a^{\Xi(k_{10}M)} \equiv a^{-\Xi(k_{20})-1} \bmod N. \tag{10}$$

For a match (k_{10}, k_{20}), compute $k_0 = k_{10}M + k_{20}$ and $s = (N + 1 + k_0^{-1}) \bmod e$. By (1), we have that the right value of s equals $p+q$. Thus solving the equation $x^2 - sx + N = 0$ in $\mathbb{Z}_e$, we obtain p and q, which yields the factorization of N.

2.2 Cryptanalysis on RSA Schemes of Sun et al.

We now apply Theorem 1 to analyze the security of RSA Schemes proposed by Sun et al. [16,17]. For example, the following RSA instance constructed in [16,17]: p and q are of 512 bits, d of 512 bits, and e of 624 bits, the attack of Theorem 1 can be applied. Considering the square root attack, parameter k should be chosen much larger than the value $l_k = 112$ as suggested by Sun et al.

3 New Attack by Applying Pollard's Method

A disadvantage of the baby-step giant-step method is that it requires a lot of storage. Pollard's ρ method, developed by Pollard [9], runs in approximately the same time as baby-step giant-step method, but requires very little storage. Recently, some authors investigated the Pollard's ρ method and obtained more efficient algorithms, see Teske [18,19,20] and Bai and Brent [1], for example.

3.1 Pollard's ρ Method

Let G be a finite cyclic group of order R. The Pollard's algorithm can be applied to any group for which the following is satisfied (see [18,20]).
◇ Given any two group elements g_i and g_j, we can compute the product $g_i * g_j$.
◇ Given any two group elements g_i and g_j, we can check whether $g_i = g_j$.
◇ Given a small integer r, we can divide the group G into r disjoint sets $G_1, \cdots, G_r$ of roughly equal size, and given any group elements g we can check to which these sets it belongs.

Choose a function $f: G \mapsto G$ that behaves rather randomly. Then start with a random element g_1 and compute the iterations $g_{i+1} = f(g_i)$. Since G is a finite set, there will be some indices $i_0 < j_0$ such that $g_{i_0} = g_{j_0}$. Then

$$g_{i_0+1} = f(g_{i_0}) = f(g_{j_0}) = g_{j_0+1},$$

and similarly, $g_{i_0+l} = g_{j_0+l}$ for all $l \geq 0$. Therefore, the sequence g_i is periodic with period $j_0 - i_0$ (or possibly a divisor of $j_0 - i_0$). The picture describing this process looks like the Greek letter ρ, which is why it is called Pollard's ρ method. If f is a randomly chosen function, we expect to find a match with j_0 in time roughly $\sqrt{R}$. For an analysis of the running time for various choices of function f, one can see [18].

As a simple implementation of the method, one can store all the points g_i until a match is found. This takes around $\sqrt{R}$ storage. However, it is possible to do much better at the cost of a little more computation. The key idea is that once there is a match for two indices differing by l, all subsequent indices differing by l will yield matches. This is just the periodicity mentioned above. Therefore, we can compute pairs (g_i, g_{2i}), for $i = 1, 2, \cdots$, but only keep the current pair, we do not store the previous pairs. These can be calculated by the rules $g_{i+1} = f(g_i)$ and $g_{2(i+1)} = f(f(g_{2i}))$. Suppose $i \geq i_0$ and i is a multiple of l. Then the indices $2i$ and i differ by a multiple of l and hence yield a match: $g_i = g_{2i}$. Since $d \leq j_0$ and $i_0 < j_0$, it follows easily that there is a match for $i \leq j_0$. Therefore the number of steps to find a match is expected to be at most a constant multiple of $\sqrt{R}$. Now it remains of how to choose a suitable function f. Three iterating functions are listed in [19]: Pollard's walk, linear walk and combined walk. In the following, we will apply the iterating function of linear walk to obtain the following result.

Theorem 2. *Suppose that $N = pq$ is an RSA modulus with primes p and q satisfying $q < p < 2q$, and the public exponent e, private exponent d and parameter k satisfy (1). Let K be a known upper bound of k such that $1 < k < K$. If*

$e > 2\sqrt{5N}$, then there exists a probabilistic algorithm to recover p and q in time $\tilde{O}(\sqrt{K})$.

Here is the roadmap for the proof of Theorem 2.

1. We prove that if a multiple of k is known, then k can be recovered in polynomial time;
2. We apply the Pollard ρ method to find a multiple of k;
3. When k is known, we recover the factorization of N.

Proof. First we prove that if a multiple of k is known, then k can be recovered. Suppose that m_k is a known multiple of k. For simplicity, we suppose that $\gcd(m_k, e) = 1$. Obviously, one can factor m_k and check each factor to find k. Here we present another method to obtain k when $(m_k^{-1} \bmod e) > p + q$. Let $\sigma = m_k^{-1} \bmod e$. Then σ is a known integer in the range $\|0, e\|$, where $\|0, e\|$ denotes the set of integers between 0 and e.

By (1), there exists an integer $\tilde{d}$ such that

$$e\tilde{d} = k^{-1} \bmod e + \phi(N). \tag{11}$$

Since $e > 2\sqrt{N} > p + q$, we have $\tilde{d} = \lfloor N/e \rfloor$. We write $m_k = s \cdot k$ with s and k unknown. For integer s satisfying $\gcd(s, e) = 1$, there exists an integer $\bar{d}$ such that

$$e\bar{d} = s \cdot ((sk)^{-1} \bmod e) + \phi(N),$$

that is,

$$e\bar{d} = s \cdot \sigma + \phi(N). \tag{12}$$

Let (12) minus (11), we have

$$e(\bar{d} - \tilde{d}) = s\sigma - (k^{-1} \bmod e).$$

Since $(k^{-1} \bmod e) < e$, we have

$$\bar{d} = \tilde{d} + \frac{s\sigma - (k^{-1} \bmod e)}{e} = \tilde{d} + \left\lfloor \frac{s\sigma}{e} \right\rfloor. \tag{13}$$

By (12), we have

$$\frac{e\bar{d}}{\sigma} = s + \frac{\phi(N)}{\sigma}.$$

If $\sigma > p + q$, then

$$\left\lceil \frac{e\bar{d}}{\sigma} \right\rceil = s + \left\lfloor \frac{N}{\sigma} \right\rfloor + \varpi, \tag{14}$$

where $\varpi = 0$ or 1. Insert (13) into (14), we have

$$\left\lceil \frac{e\tilde{d}}{\sigma} + \frac{e}{\sigma}\left\lfloor \frac{s \cdot \sigma}{e} \right\rfloor \right\rceil = \left\lfloor \frac{N}{\sigma} \right\rfloor + s + \varpi. \tag{15}$$

Note that in the above equation, all parameters but s are known. Thus one can solve s easily, e.g., by the bisection method. Once s is known, we can calculate k by $k = (s\sigma)^{-1} \bmod e$.

Next, we use Pollard ρ method to find a multiple of k. We also prove that with a probability of at least $\frac{1}{2}$ we can find an integer ϱ being a multiple of k_0 and satisfying $(\varrho^{-1} \bmod e) > p+q$ in time $\tilde{O}(\sqrt{K})$. Let k_0 denote the right value of k in the following. Define

$$G_i = \{g \mid g \in \mathbb{Z} \text{ with } g \equiv i \bmod 20\}$$

where $i = 1, 2, \cdots, 20$. Let M_j be integers chosen randomly, $j = 1, 2, \cdots, 20$. Define a function $f : \mathbb{Z} \mapsto \mathbb{Z}$ such that

$$f(g) = g + M_j, \text{ if } g \in G_j. \tag{16}$$

Finally, we choose random integer g_0, and let g_0 be the starting point for the random walk. While computing g_i and g_{2i} by the rules

$$g_{i+1} = f(g_i), \quad g_{2(i+1)} = f(f(g_{2i})),$$

and store the pair. If g_i and g_{2i} satisfy

$$|g_i - g_{2i}| \equiv 0 \bmod (20k_0), \tag{17}$$

we call that g_i and g_{2i} is a match. We are going to explain how to check if g_i and g_{2i} is a match. (In fact, the above process can also be stated as the follows: Define $G = \mathbb{Z}_{20k_0}$ and $f' : G \mapsto G$ with $f'(g) = g + M_j \bmod 20k_0$, if $g \in G_j$. Then by [19], one can find a match g_i and g_{2i} satisfying (17) by the linear walk method. One can see that if $g \in G_j$, then $g \bmod (20k_0) \in G_j$. Thus we need not compute the modulo operations to find a match. This explains why there is no modulo in (16).)

Let $\varrho = |g_i - g_{2i}|$. If g_i and g_{2i} is a match, then ϱ is a multiple of k_0. We can apply the discussions between (11) and (15) to recover k. Here if $\gcd(\varrho, e) \neq 1$, we used $\frac{\varrho}{\gcd(\varrho,e)}$ instead of ϱ. The probability that $\varrho^{-1} \bmod e > p+q$ is $1 - \frac{p+q}{e} > \frac{1}{2}$.

In other words, for each pair g_j and g_{2j}, we can check if it is a match suitable to recover k_0. By Pollard ρ method, we can find a match of g_i and g_{2i} satisfying (17) in time $\tilde{O}(\sqrt{K})$, and finally recover k_0 in time $\tilde{O}(\sqrt{K})$. Once the right value of k is known, p and q can be calculated as the discussions at the end of §2.1.

3.2 Cryptanalysis on RSA schemes of Sun et al.

In the RSA schemes of Sun et al. [16,17], bit sizes of the parameters in Schemes A and B satisfy $l_e \geq \frac{1}{2}(l_k + l_N)$. Thus Theorem 2 can be applied to analyze the security of Schemes A and B. Considering the square root attack, the size of parameter k should not be chosen larger than 112 bits.

4 Conclusion

Since Coppersmith presented new methods of finding small modular and integer roots of polynomial equations in 1996, variations of these methods have been widely applied on the cryptanalysis of RSA. However, the square root attacks can also be used to recover unknown variables whose size is small (e.g., less than 160 bits) in RSA equation, even when Coppersmith's method does not work. This paper investigates the baby-step giant-step method and the Pollard's ρ method. Our main result, Theorems 1 and 2 to be exactly, presents an application of these two methods on RSA cryptanalysis.

References

1. Bai, S., Brent, R.P.: On the efficiency of Pollards rho method for discrete logarithms. In: Harland, J., Manyem, P. (eds.) CATS 2008, pp. 125–131. Australian Computer Society (2008)
2. Blömer, J., May, A.: A Generalized Wiener Attack on RSA. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 1–13. Springer, Heidelberg (2004)
3. Boneh, D., Durfee, G.: Cryptanalysis of RSA with Private Key d Less than $N^{0.292}$. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 1–11. Springer, Heidelberg (1999)
4. Boneh, D., Durfee, G.: Cryptanalysis of RSA with private key d less than $N^{0.292}$. IEEE Trans. on Information Theory 46(4), 1339–1349 (2000)
5. Coppersmith, D.: Small solutions to polynomial equations and low exponent RSA vulnerabilities. Journal of Cryptology 10(4), 223–260 (1997)
6. Crandall, R., Pomerance, C.: Prime Number, 2nd edn. Springer (2005)
7. Durfee, G., Nguyên, P.Q.: Cryptanalysis of the RSA Schemes with Short Secret Exponent from Asiacrypt '99. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 14–29. Springer, Heidelberg (2000)
8. May, A.: Using LLL-reduction for solving RSA and factorization problems: a survey. In: LLL+25 Conference in Honour of the 25th Birthday of the LLL Algorithm (2007)
9. Pollard, J.M.: Monte Carlo methods for index computation ($\bmod p$). Math. Comp. 32(143), 918–924 (1978)
10. Quisquater, J.J., Couvreur, C.: Fast decipherment algorithm for RSA public-key cryptosystem. Electronic Letters 18, 905–907 (1982)
11. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Commun. of the ACM 21, 120–126 (1978)
12. Sarkar, S., Maitra, S.: Partial key exposure attacks on RSA and its variant by guessing a few bits of one of the prime factors. Bull. Korean Math. Soc. 46(4), 721–741 (2009)
13. Shanks, D.: Class number, a theory of factorization and genera. In: 1969 Number Theory Institute (Proc. Sympos. Pure Math., vol. XX, State Univ. New York, Stony Brook, NY, 1969), pp. 415–440 (1969)
14. Sun, H.-M., Yang, W.-C., Laih, C.-S.: On the Design of RSA with Short Secret Exponent. In: Lam, K.-Y., Okamoto, E., Xing, C. (eds.) ASIACRYPT 1999. LNCS, vol. 1716, pp. 150–164. Springer, Heidelberg (1999)
15. Sun, H.M., Yang, C.T., Lai, C.S.: On the design of RSA with short secret exponent. Journal of Information Science and Engineering 18(1), 1–18 (2002)

16. Sun, H.-M., Yang, C.-T.: RSA with Balanced Short Exponents and Its Application to Entity Authentication. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 199–215. Springer, Heidelberg (2005)
17. Sun, H.M., Yang, C.T., Wu, M.: Short exponent RSA. IEICE Trans. Fundamentals E92-A(3), 912–918 (2009)
18. Teske, E.: Speeding Up Pollard's Rho Method for Computing Discrete Logarithms. In: Buhler, J.P. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 541–554. Springer, Heidelberg (1998)
19. Teske, E.: A space efficient algorithm for group structure computation. Mathematics Computation 67(224), 1637–1663 (1998)
20. Teske, E.: On random walks for Pollards rho method. Mathematics of Computation 70(234), 809–825 (2001)
21. de Weger, B.: Cryptanalysis of RSA with small prime difference. Applicable Algebra in Engineering 13, 17–28 (2002)
22. Wiener, M.: Cryptanalysis of short RSA secret exponents. IEEE Transactions on Information Theory 36, 553–558 (1990)

An Algebraic Broadcast Attack against NTRU

Jintai Ding[1,2], Yanbin Pan[3], and Yingpu Deng[3]

[1] Chongqing University,
[2] Department of Mathematical Sciences, University of Cincinnati
[3] Key Laboratory of Mathematics Mechanization,
Academy of Mathematics and Systems Science, Chinese Academy of Sciences
jintai.ding@gmail.com, {panyanbin,dengyp}@amss.ac.cn

Abstract. In this paper, we propose an algebraic broadcast attack against NTRU, which recovers a single message encrypted multiple times using different NTRU public keys. Namely, when a message is broadcasted, under some reasonable assumptions, our attack can be completed in polynomial time and space. To the best of our knowledge, this is the first successful broadcast attack against NTRU.

Keywords: Broadcast attack, NTRU, lattice-based cryptosystems, LWE.

1 Introduction

The NTRU cryptosystem of Hoffstein, Pipher, Silverman [12] is one of the most practical public key schemes, which is an IEEE 1363.1 Standard [18]. It features reasonably short, easily created keys, high speed, and low memory requirements.

Coppersmith and Shamir [3] showed the security of NTRU is related to the hardness of certain lattice problems. Although there are many attacks, like the ciphertext-only attacks [3,19,15,13], no significant weakness has seemed to be yet found on NTRU encryption. The most effective attacks may have been the chosen-ciphertext attacks[14,8], most of which utilize the NTRU's decryption failures. When a single message is encrypted multiple times using a NTRU public key, Hoffstein and Silverman [10] proposed a multiple transmission attack.

Another type of attack, the broadcast attack, was first proposed by Hästad [9] in 1988. The attack enables an attacker to recover the single message sent by the sender to multiple recipients, who use the same type of cryptosystrems but with different keys, without requiring any knowledge of the recipients' secret keys. Another effective attack is the hybrid attack [13,17].

In 2009, Plantard and Susilo [22] first considered the broadcast attack against the lattice-based public key cryptosystems. They constructed many lattices that share the same short vector carrying the information of a single message. By intersecting these lattices, they then gave some heuristic attacks using the lattice reduction algorithm. However, they also showed that their attacks do not apply to NTRU, since the lattices in general do not share the same short target vector.

In this paper, we propose an algebraic broadcast attack against NTRU, which is based on the new algorithms in [5,6,7,1]. Instead of the lattice reduction algorithm, we present a new algebraic algorithm to complete the attack. The new algorithm involves nonlinearization and linearization, and can also be used to solve the learning with errors problem with bounded errors.

Since there are many variants of NTRU, we give our attacks against the main instantiations: NTRU-1998 [12], NTRU-2001 [11] (with an odd d_g whose definition can be found in Section 2.2) and NTRU-2005 [16]. Under some reasonable assumption, to complete the attacks, we need gather $O(N)$ (resp. $O(N^2)$) recipients' public keys and ciphertexts, and solve a set of $O(N^2)$ (resp. $O(N^3)$) linear equations with $O(N^2)$ (resp. $O(N^3)$) variables for NTRU-2001 (with an odd d_g) and NTRU-2005 (resp. NTRU-1998), where N is the main parameter of NTRU. The attacks are efficient since they may be completed in polynomial time and space, but these attacks become more difficult in practice as N increases.

The hybrid of these attacks and some other attacks, for example, the lattice reduction attack and the meet-in-the-middle attack, may lead better attacks. With XL [4,2] type of algorithms, we can further reduce the number of recipients, but increase the computation complexity. It remains an open problem to find an efficient broadcast attack with a constant number of recipients. Compared with the chosen-ciphertext attacks, our attacks don't need the decryption oracle or the decryption failures. Compared with the multiple transmission attack, our attacks allow that the recipients' public keys are different. Different public keys make attacking NTRU a hard task. Our broadcast attacks do not work for more sophisticated padding schemes of NTRU. We can also use our method to attack some other lattice-based cryptosystems which have a similar linear structure to NTRU and bounded random perturbations over $\mathbb{Z}_q$.

To attack NTRU-1998 and NTRU-2001 with an odd d_g, we have to extend the algorithm in [5,6,7,1] over finite field onto the ring $\mathbb{Z}_{2^k}$. Further more, we can derive N linear equations with fewer variables from every recipient instead of one linear equation with much more variables. Thus, we use much fewer recipients, and solve much fewer linear equations for much fewer variables to complete the attacks.

The paper is organized as follows. Section 2 gives some preliminaries. Section 3 describes our broadcast attacks. Section 4 presents the conclusion.

2 Preliminaries

Let $\mathbb{Z}$ be the integer ring, $\mathbb{Z}_q$ the residue class ring $\mathbb{Z}/q\mathbb{Z}$, $\mathbb{F}_q$ the finite fields $\mathbb{Z}_q$ when q is a prime. We use bold letters to denote vectors, in column notation. For a vector $\mathbf{v}$, we denote by $\mathbf{v}_i$ the i-th entry of $\mathbf{v}$.

2.1 The Learning with Errors Problem

The learning with errors (LWE) problem introduced by Regev [23] has many applications in constructing cryptosystems with security proofs.

An LWE problem has a parameter n, a prime modulus q, and an error probability distribution κ on the finite field $\mathbb{F}_q$. Let $\prod_{\mathbf{m},\kappa}$ on $\mathbb{F}_q$ be the probability distribution obtained by selecting an element $\mathbf{a}$ in $\mathbb{F}_q^n$ randomly and uniformly, choosing $r \in \mathbb{F}_q$ according to κ, and outputting $(\mathbf{a}, <\mathbf{a}, \mathbf{m}> +r)$, where $+$ is the addition that is performed in $\mathbb{F}_q$. We say that an algorithm solves LWE with modulus q and error distribution κ, if, for any $\mathbf{m}$ in $\mathbb{F}_q^n$, with an arbitrary number of independent samples from $\prod_{\mathbf{m},\kappa}$, it outputs $\mathbf{m}$ with high probability.

2.2 NTRU

We present a brief description of the NTRU-1998 cryptosystems. For more details see [12]. The NTRU-1998 cryptosystem depends on three integer parameters (N, p, q) and four sets $\mathcal{L}_f, \mathcal{L}_g, \mathcal{L}_r, \mathcal{L}_m$ of polynomials of degree $N-1$ with small integer coefficients. We choose p, q such that $\gcd(p, q) = 1$ and p is much smaller than q. Denote the ring $\mathbb{Z}[x]/(x^N - 1)$ by R and the multiplication in R by $*$. Every element in R can be represented as a polynomial. For example, $f \in R$ can be represented as $f = \sum_{i=0}^{N-1} f_i x^i$. We work over the ring R.

Key Generation: **Step 1.** Choose $f \in \mathcal{L}_f, g \in \mathcal{L}_g$ such that there exist $F_q, F_p \in R$ satisfying $f * F_q = 1 \bmod q$ and $f * F_p = 1 \bmod p$. **Step 2.** Let $h = p * F_q * g \bmod q$.
Public Key: h, p, q.
Private Key: f, F_p.
Encryption: To encrypt a message $m \in \mathcal{L}_m$, first we choose an $r \in \mathcal{L}_r$ randomly, then compute the ciphertext: $c = h * r + m \bmod q$.
Decryption: First we compute $a = f * c \bmod q == pg * r + f * m \bmod q$, then we choose the coefficients of a in the interval from $-\frac{q}{2}$ to $\frac{q}{2}$. By the fact that all the coefficients of $pg * r + f * m$ may be in the interval from $-\frac{q}{2}$ to $\frac{q}{2}$, we almost get $a = pg * r + f * m$. Then we recover the message m by computing $m = F_p * a \bmod p$.

As pointed in [20], analyzing NTRU is a tricky task, since there are several variants of NTRU. We may use totally different ways to attack different variants of NTRU instead of a uniform one. We summarize the main instantiations of NTRU in the table below as in [20]:

Variant	q	p	$\mathcal{L}_f$	$\mathcal{L}_g$	$\mathcal{L}_m$	$\mathcal{L}_r$	F	Ref
NTRU-1998	$2^k \in [\frac{N}{2}, N]$	3	$L(d_f, d_f - 1)$	$L(d_g, d_g)$	L_m	$L(d_r, d_r)$	-	[12]
NTRU-2001	$2^k \in [\frac{N}{2}, N]$	$2 + x$	$1 + p * F$	$\mathcal{B}(d_g)$	$\mathcal{B}$	$\mathcal{B}(d_r)$	$\mathcal{B}(d_F)$	[11]
NTRU-2005	prime	2	$1 + p * F$	$\mathcal{B}(d_g)$	$\mathcal{B}$	$\mathcal{B}(d_r)$	$\mathcal{B}(d_F)$	[16]

where $L_m = \{m \in R : m$ has coefficients lying between $-\frac{1}{2}(p-1)$ and $\frac{1}{2}(p-1)\}$, $L(d_1, d_2) = \{F \in R : F$ has d_1 coefficients equal $1, d_2$ coefficients equal -1, the rest $0\}$, $\mathcal{B}$ denotes the set of all polynomials with binary coefficients, $\mathcal{B}(d) = \{F \in R : F$ has d coefficients equal 1, the rest $0\}$.

2.3 Transforming NTRU into Its Linear Form

In NTRU, a polynomial $f = \sum_{i=0}^{N-1} f_i x^i \in R$ can also be represented as a vector: $\mathbf{f} = (f_0, f_1, \cdots, f_{N-1})^T$. The multiplication of f and g can be represented as $\begin{pmatrix} f_0 & f_{N-1} & \cdots & f_1 \\ f_1 & f_0 & \cdots & f_2 \\ \vdots & \vdots & \ddots & \vdots \\ f_{N-1} & f_{N-2} & \cdots & f_0 \end{pmatrix} \begin{pmatrix} g_0 \\ g_1 \\ \vdots \\ g_{N-1} \end{pmatrix}$. The result is the vector corresponding to $f * g$ in R. Then, for the equation in R: $c = h * r + m \bmod q$, we have the linear form

$$\mathbf{c} = H\mathbf{r} + \mathbf{m} \bmod q, \tag{1}$$

where $H = \begin{pmatrix} h_0 & h_{N-1} & \cdots & h_1 \\ h_1 & h_0 & \cdots & h_2 \\ \vdots & \vdots & \ddots & \vdots \\ h_{N-1} & h_{N-2} & \cdots & h_0 \end{pmatrix}$.

Proposition 1. *For a uniformly random instance of NTRU-1998 (also NTRU-2001 with an odd d_g, NTRU-2005), for any message $\boldsymbol{m}$, ciphertext $\boldsymbol{c}$ and the corresponding random vector $\boldsymbol{r}$, we can find $\widehat{H} \in \mathbb{Z}_q^{N\times N}$ and $\boldsymbol{b} \in \mathbb{Z}_q$ with very high probability in polynomial time, without knowing $\boldsymbol{m}$ and $\boldsymbol{r}$, such that*

$$\widehat{H}\boldsymbol{m} + \boldsymbol{r} = \boldsymbol{b} \textit{ mod } q.$$

Proof. For NTRU-2001 with an odd d_g and NTRU-2005, H is invertible in $\mathbb{Z}_q^{N\times N}$ with very high probability. So we can get easily from (1): $H^{-1}\mathbf{m} + \mathbf{r} = H^{-1}\mathbf{c} \bmod q$. Let $\widehat{H} = H^{-1}$, $\mathbf{b} = H^{-1}\mathbf{c}$, then we have: $\widehat{H}\mathbf{m} + \mathbf{r} = \mathbf{b} \bmod q$.

However, For NTRU-2001 with an even d_g, H is not invertible. It seems that we need some extra restrictions on the other parameters, for example, d_r, to get a similar result as the proposition. That's why we exclude the case.

For NTRU-1998, notice that H is not invertible in $\mathbb{Z}_q^{N\times N}$ either. However, for any public key h in NTRU-1998, we can usually find a polynomial $h' \in R$ with overwhelming probability such that for any $r \in L(d_r, d_r)$: $h' * h * r = r \bmod q$.

We say h is pseudo-invertible as in [21].

As pointed in [21], we can find h' in polynomial time as follows. Since $R_q = \mathbb{Z}_q[x]/(x^N - 1)$ is isomorphic to $P_1 \times P_2$ where $P_1 = \mathbb{Z}_q[x]/(x-1)$ and $P_2 = \mathbb{Z}_q[x]/(x^{N-1} + x^{N-2} + \cdots + 1)$, we have $\phi : R_q \to P_1 \times P_2$.

Since $h(1) = 0 \bmod q$, we have $\phi(h) = (0, \bar{h})$ where $\bar{h}$ denotes the reduction of h modulo $x^{N-1} + x^{N-2} + \cdots + 1$. With high probability, $\bar{h}$ is invertible in P_2. We denote its inverse in P_2 by $\widetilde{h}$. Considering the polynomial $h' = \phi^{-1}((1, \widetilde{h}))$ in R_q, it satisfies $h' * h * r = r \bmod q$ for $r \in L(d_r, d_r)$.

Let $H' = \begin{pmatrix} h'_0 & h'_{N-1} & \cdots & h'_1 \\ h'_1 & h'_0 & \cdots & h'_2 \\ \vdots & \vdots & \ddots & \vdots \\ h'_{N-1} & h'_{N-2} & \cdots & h'_0 \end{pmatrix}$, then we have: $H'\mathbf{m} + \mathbf{r} = H'\mathbf{c} \bmod q$, from (1). Similarly, let $\widehat{H} = H'$, $\mathbf{b} = H'\mathbf{c}$, then we have $\widehat{H}\mathbf{m} + \mathbf{r} = \mathbf{b} \bmod q$.

Obviously, from Proposition 1, for $i = 0, \cdots N-1$, we get N linear equations: $\sum_{j=0}^{N-1} \widehat{H}_{i+1,j+1}\mathbf{m}_j + \mathbf{r}_i = \mathbf{b}_i \bmod q$.

3 A Broadcast Attack against NTRU

Suppose there is a sender and n recipients. All recipients use NTRU cryptosystems with the same parameters N, q, p but different public/private keys. The sender encrypts the single message m with each recipient's public key with independent $r \in \mathcal{L}_r$ respectively, and sends the n ciphertexts to corresponding recipients. The broadcast attack is to recover m with these n ciphertexts. More precisely, the attacker wants to recover m from the n equations:

$$h_{(i)} * r_{(i)} + m = c_{(i)} \bmod q$$

where $h_{(i)}$ is the i-th recipient's public key.

For each recipient, by Proposition 1, we have N linear equations. For each linear equation, taking $\sum\limits_{j=0}^{N-1} \widehat{H}_{1,j+1}\mathbf{m}_j + \mathbf{r}_0 = \mathbf{b}_0 \bmod q$ as an example, let $\mathbf{a} = (\widehat{H}_{1,1}, \widehat{H}_{1,2}, \cdots, \widehat{H}_{1,N})$, we have a pair, $(\mathbf{a}, \mathbf{b}_0 =< \mathbf{a}, \mathbf{m} > + \mathbf{r}_0)$, which can be seen as a sample from an LWE oracle. So, recovering m from the n recipients is similar to solving an LWE problem for $\mathbf{m}$ from nN samples.

3.1 The Basic Algorithm for LWE with Bounded Errors

An LWE problem is with bounded errors if the error probability distribution is on a proper subset $ES = \{e_1, e_2, \cdots, e_D\}$ of $\mathbb{F}_q$ with fixed $D (D < q)$. The main steps of the algorithm for LWE with bounded errors are:

1. Nonlinearization. For any sample $(\mathbf{a}, < \mathbf{a}, \mathbf{m} > + r)$, let $b' =< \mathbf{a}, \mathbf{m} > + r$. We know that $\mathbf{m}$ satisfies: $\sum_{i=1}^{N} \mathbf{a}_i \mathbf{x}_i + r = b'$.
   So $\mathbf{m}$ also satisfies the corresponding nonlinear equation
   $$\prod_{r_k \in ES} (\sum_{i=1}^{N} \mathbf{a}_i \mathbf{x}_i + r_k - b') = 0. \tag{2}$$
   Note here that D needs to be less than q, otherwise the equation above will be totally trivial, namely the so-called field equations: $\mathbf{x}_i^q = \mathbf{x}_i$.
2. Linearization. We will solve Equation (2) by linearization. Notice that the total degree of every monomial of $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N$ in (2) is at most D. We assign each of such monomials a new variable $\mathbf{y}_i$. Let $Q(N, D) = \binom{N+D}{N}$. We know that the number of $\mathbf{y}_i$'s, denoted by d, is at most $Q(N, D) - 1$. What's more, we can assign $\mathbf{x}_i$ to be $\mathbf{y}_{d-N+i}$, then we transform (2) into a linear equation: $\sum\limits_i \mathbf{c}_i \mathbf{y}_i = b$, where $\mathbf{c}_i$ is the corresponding coefficient of $\mathbf{y}_i$ in (2).

3. Solving. Since there are d $\mathbf{y}_i$'s, when we have enough linear equations from above to find a $d \times d$ nonsingular matrix L with good probability such that, $L\mathbf{y} = \mathbf{b}$, where $\mathbf{b}$ is a constant vector, we can find $\mathbf{y}$ by solving the set of linear equations. Then we can solve $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N)^T$.

We next estimate the expectation of the number of samples in the algorithm under an optimistic assumption that $\mathbf{v}$ is chosen from $\mathbb{F}_q^d$ uniformly and independently.

Let X be the number of samples until we first have $|F| = d$, X_i be the number of samples until we first have $|F| = i$ starting from when we first have $|F| = i-1$, where $i = 1, 2, \cdots, d$. Obviously, $X = \sum\limits_{i=1}^{d} X_i$.

When $|F| = i-1$, the probability we have a new $\mathbf{v}$ that can't be generated by the vectors in F is $1 - \frac{q^{i-1}}{q^d}$. Hence, the expectation of X_i is $\frac{1}{1-\frac{q^{i-1}}{q^d}} = \frac{q^d}{q^d - q^{i-1}}$.

Then, $E(X) = \sum\limits_{i=1}^{d} E(X_i) = \sum\limits_{i=1}^{d} \dfrac{q^d}{q^d - q^{i-1}}$.

We just give an upper bound for $E(X)$ here. Since $\dfrac{q^d}{q^d - q^{i-1}} < \dfrac{q^d}{q^d - q^{d-1}} = \dfrac{q}{q-1}$, we have $E(X) < \dfrac{q}{q-1}d$.

We implemented the algorithm. When we make $d + O(N)$ queries, we have never failed to obtain an invertible L in all the extensive experiments (thousands and $q > 3$), hence to solve the LWE problem. So it is reasonable to believe the algorithm succeeds with very high probability.

3.2 The Broadcast Attack against NTRU-1998

For NTRU-1998, $ES = \{-1, 0, 1\}$ and $\mathbf{m} \in \{-1, 0, 1\}^N$. Since $q = 2^k$, notice that the algorithm above works over the finite field but not the ring $\mathbb{Z}_{2^k}$ and L is invertible over $\mathbb{Z}_{2^k}^{d \times d}$ if and only if L is invertible over $\mathbb{F}_2^{d \times d}$. So, we try to work over the finite field $\mathbb{F}_2$. A nature idea is to transform these equations into the ones over $\mathbb{F}_2$ by simply mapping each coefficient into $\mathbb{F}_2$. However, if we did so, we would get $ES = \{0, 1\}$ which is not a proper set of $\mathbb{F}_2$. So we have to involve some new ideas to solve the problem.

Since we will work over $\mathbb{F}_2$, we don't recover $\mathbf{m}$ directly, but to find another $\mathbf{m}' \in \mathbb{F}_2^N$ such that $\mathbf{m} = \mathbf{m}' \bmod 2$. We first show that this is enough for our attack.

Suppose we have already known $\mathbf{m}' = \mathbf{m} \bmod 2$, we know exactly which entries of $\mathbf{m}$ are zero. Without loss of generality, we assume $\mathbf{m}_i = 0$ for $i = 0, 1, \cdots, t-1$. Then for every recipient, we eliminate the first t columns of the corresponding $\widehat{H}$, and denote the remaining $N \times (N-t)$ matrix by $\widehat{H}'$. We have

$$\widehat{H}'(\mathbf{m}_t, \cdots, \mathbf{m}_{N-1})^T + \mathbf{r} = b \bmod 2^k,$$

where $\mathbf{m}_i$ is either -1 or 1 and q is a power of 2. So

$$\widehat{H}'(\mathbf{m}_t + 1, \cdots, \mathbf{m}_{N-1} + 1)^T + \mathbf{r} = b + \widehat{H}'(1, \cdots, 1)^T \bmod 2^k,$$

where $\mathbf{m}_i + 1$ is either 0 or 2. Notice that $\mathbf{r}_i = 0$ if and only if the $i-$th entry of $b + \widehat{H}'(1, \cdots, 1)^T$ is congruent to 0 modulo 2. We will get a set of $\mathbf{r}_i$'s which equal 0. For each of these $\mathbf{r}_i$'s, we get a linear equation: $\sum_{j=0}^{N-t-1} \widehat{H}'_{i+1,j+1}\mathbf{m}_{t+j} = \mathbf{b}_i$.

For each recipient, we can have $N - 2d_r$ such linear equations. When collecting enough equations, we can obtain these $\mathbf{m}_i$'s.

We next show that how to find $\mathbf{m}'$ such that $\mathbf{m} = \mathbf{m}' \bmod 2$.

From (2), we know $\mathbf{m}$ satisfies $\prod_{r_k \in \{-1,0,1\}} (\sum_{i=1}^N \mathbf{a}_i\mathbf{x}_i + r_k - b') = 0 \bmod 2^k$, i.e.

$$\begin{aligned}&\sum_{i=1}^{N} \mathbf{a}_i^3\mathbf{x}_i^3 + 6\sum_{i<j<k} \mathbf{a}_i\mathbf{a}_j\mathbf{a}_k\mathbf{x}_i\mathbf{x}_j\mathbf{x}_k + 3\sum_{i\neq j} \mathbf{a}_i^2\mathbf{a}_j\mathbf{x}_i^2\mathbf{x}_j\\ &-6\sum_{i<j} b'\mathbf{a}_i\mathbf{a}_j\mathbf{x}_i\mathbf{x}_j - 3\sum_{i=1}^{N} b'\mathbf{a}_i^2\mathbf{x}_i^2 + \sum_{i=1}^{N}(3(b')^2\mathbf{a}_i - \mathbf{a}_i)\mathbf{x}_i\\ =\ &(b')^3 - b' \bmod 2^k.\end{aligned}$$

Since $\mathbf{x}_i^3 = \mathbf{x}_i$ for $\mathbf{x}_i \in \{-1, 0, 1\}$ and there exists $\bar{\mathbf{x}}_i \in \{0, 1\}$ such that $\mathbf{x}_i^2 = \mathbf{x}_i + 2\bar{\mathbf{x}}_i$, substituting $\mathbf{x}_i$ for $\mathbf{x}_i^3$, and $\mathbf{x}_i + 2\bar{\mathbf{x}}_i$ for $\mathbf{x}_i^2$, we have

$$\begin{aligned}&6\sum_{i<j<k} \mathbf{a}_i\mathbf{a}_j\mathbf{a}_k\mathbf{x}_i\mathbf{x}_j\mathbf{x}_k + \sum_{i<j}(3\mathbf{a}_i^2\mathbf{a}_j + 3\mathbf{a}_j^2\mathbf{a}_i - 6b'\mathbf{a}_i\mathbf{a}_j)\mathbf{x}_i\mathbf{x}_j\\ &+6\sum_{i\neq j} \mathbf{a}_i^2\mathbf{a}_j\bar{\mathbf{x}}_i\mathbf{x}_j - 6\sum_{i=1}^{N} b'\mathbf{a}_i^2\bar{\mathbf{x}}_i + \sum_{i=1}^{N}(3(b')^2\mathbf{a}_i - \mathbf{a}_i + \mathbf{a}_i^3 - 3b'\mathbf{a}_i^2)\mathbf{x}_i\\ =\ &(b')^3 - b' \bmod 2^k.\end{aligned}$$

Since $t^3 = t \bmod 2$ and $t^2 = t \bmod 2$ for any integer t, obviously there is a positive integer $u \geq 1$ such that 2^u divides the greatest common divisor of all the coefficients but 2^{u+1} can not. If $u < k$, we have

$$\begin{aligned}&\frac{3}{2^{u-1}}\sum_{i<j<k} \mathbf{a}_i\mathbf{a}_j\mathbf{a}_k\mathbf{x}_i\mathbf{x}_j\mathbf{x}_k + \frac{1}{2^u}\sum_{i<j}(3\mathbf{a}_i^2\mathbf{a}_j + 3\mathbf{a}_j^2\mathbf{a}_i - 6b'\mathbf{a}_i\mathbf{a}_j)\mathbf{x}_i\mathbf{x}_j\\ &+\frac{3}{2^{u-1}}\sum_{i\neq j} \mathbf{a}_i^2\mathbf{a}_j\bar{\mathbf{x}}_i\mathbf{x}_j - \frac{3}{2^{u-1}}\sum_{i=1}^{N} b'\mathbf{a}_i^2\bar{\mathbf{x}}_i + \frac{1}{2^u}\sum_{i=1}^{N}(3(b')^2\mathbf{a}_i - \mathbf{a}_i + \mathbf{a}_i^3 - 3b'\mathbf{a}_i^2)\mathbf{x}_i\\ =\ &\frac{1}{2^u}((b')^3 - b') \bmod 2^{k-u}.\end{aligned}$$

As above, we assign each of the monomials a new variable $\mathbf{y}_i$(Notice that $\mathbf{y}_{d-N+i} = \mathbf{x}_i$ for $1 \leq i \leq N$), and transform it into a linear equation: $\sum_i \bar{\mathbf{c}}_i\mathbf{y}_i = \bar{b} \bmod 2^{k-u}$.

Let $\mathbf{c}_i = \bar{\mathbf{c}}_i \bmod 2$ and $b = \bar{b} \bmod 2$. We get a linear equation over $\mathbb{F}_2$: $\sum_i \mathbf{c}_i\mathbf{y}_i = b \bmod 2$.

Collecting enough equations in our experiments, we always get a matrix L, by performing Gaussian elimination, such that

$$L\mathbf{y} = \left(\begin{array}{c|c} 1 \cdots & \\ \ddots & * \\ \hline \mathbf{0} & I_{N\times N}\end{array}\right)\begin{pmatrix}\mathbf{y}_1\\ \vdots\\ \hline \mathbf{x}\end{pmatrix} = \mathbf{b}.$$

whereas L is not invertible, $\mathbf{x} = (\mathbf{x}_1, \cdots, \mathbf{x}_N)^T$. Nevertheless, it is enough for us to recover $\mathbf{m}'$ since we can solve for $\mathbf{x}$.

What's more, we can use the strategy below to get an invertible L with very high probability. Notice that $u = 1$ holds for very high probability. We just use those samples in which $u = 1$. So we have

$$\begin{aligned}
&\sum_{i<j<k} \mathbf{a}_i\mathbf{a}_j\mathbf{a}_k\mathbf{x}_i\mathbf{x}_j\mathbf{x}_k + \frac{1}{2}\sum_{i<j}(3\mathbf{a}_i^2\mathbf{a}_j + 3\mathbf{a}_j^2\mathbf{a}_i - 6b'\mathbf{a}_i\mathbf{a}_j)\mathbf{x}_i\mathbf{x}_j \\
&+ \sum_{i\neq j} \mathbf{a}_i\mathbf{a}_j\bar{\mathbf{x}}_i\mathbf{x}_j - \sum_{i=1}^{N} b'\mathbf{a}_i\bar{\mathbf{x}}_i + \frac{1}{2}\sum_{i=1}^{N}(3(b')^2\mathbf{a}_i - \mathbf{a}_i + \mathbf{a}_i^3 - 3b'\mathbf{a}_i^2)\mathbf{x}_i \qquad (3)\\
= &\frac{1}{2}((b')^3 - b') \bmod 2.
\end{aligned}$$

Denote by $\mathbf{y}_{(i,j)}$ the new variable corresponding to $\bar{\mathbf{x}}_i\mathbf{x}_j$, by $\mathbf{y}_{\{i,j,s\}}$ the new variable corresponding to $\mathbf{x}_i\mathbf{x}_j\mathbf{x}_s$ and by $\mathbf{c}_{(i,j)}$, $\mathbf{c}_{\{i,j,s\}}$ their coefficients in the equation above. We claim that

Proposition 2. *Denote by S the set $\{i_1, i_2, i_3\}$ where $1 \le i_k \le N$, we have:* **1).** $\boldsymbol{c}_{(i,j)} = \boldsymbol{c}_{(j,i)}$ *mod* 2; **2).** $\boldsymbol{c}_{(i,j)} = \sum_{\{i,j\}\subset S} \boldsymbol{c}_S$ *mod* 2.

Proof. The first one is obvious. It remains to prove the second one. Denote by $\widehat{h}(x)$ the polynomial corresponding to any row of the matrix $\widehat{H}$. Since $\widehat{h}(x)$ is invertible over the ring $\mathbb{Z}_{2^k}[x]/(x^N - 1)$, we have $\widehat{h}(1) = 1 \bmod 2$. Hence,

$$\sum_{i=1}^{N} \mathbf{a}_i = \widehat{h}(1) = 1 \bmod 2. \qquad (4)$$

Finally, we have $\mathbf{c}_{(i,j)} = \mathbf{a}_i\mathbf{a}_j \sum_{s=1}^{N} \mathbf{a}_s = \sum_{\{i,j\}\subset S} \mathbf{c}_S + \mathbf{a}_i^2\mathbf{a}_j + \mathbf{a}_i\mathbf{a}_j^2 = \sum_{\{i,j\}\subset S} \mathbf{c}_S + \mathbf{a}_i\mathbf{a}_j + \mathbf{a}_i\mathbf{a}_j = \sum_{\{i,j\}\subset S} \mathbf{c}_S \bmod 2.$

We involve some new variables $\bar{\mathbf{y}}_{\{i,j,s\}} = \mathbf{y}_{\{i,j,s\}} + \sum_{\{i,j\}\subset S}(\mathbf{y}_{(i,j)} + \mathbf{y}_{(j,i)})$.

Since $\mathbf{c}_{(i,j)}\mathbf{y}_{(i,j)} = \sum_{\{i,j\}\subset S} \mathbf{c}_S\mathbf{y}_{(i,j)}$ and $\mathbf{c}_{(j,i)}\mathbf{y}_{(j,i)} = \sum_{\{i,j\}\subset S} \mathbf{c}_S\mathbf{y}_{(j,i)}$, we have

$$\begin{aligned}
\sum_{i<j<k} \mathbf{c}_{\{i,j,s\}}\bar{\mathbf{y}}_{\{i,j,s\}} &= \sum_{i<j<k} \mathbf{a}_i\mathbf{a}_j\mathbf{a}_k\mathbf{x}_i\mathbf{x}_j\mathbf{x}_k + \sum_{i\neq j} \mathbf{a}_i\mathbf{a}_j\bar{\mathbf{x}}_i\mathbf{x}_j \mod 2\\
&= \sum_{i<j<k} \mathbf{c}_{\{i,j,s\}}\mathbf{y}_{\{i,j,s\}} + \sum_{i\neq j} \mathbf{c}_{(i,j)}\mathbf{y}_{(i,j)} \mod 2.
\end{aligned}$$

Using the new variables $\bar{\mathbf{y}}_{\{i,j,s\}}$ in Equation (3), we can eliminate the old variables $\mathbf{y}_{\{i,j,s\}}$ and $\mathbf{y}_{(i,j)}(i \neq j)$. Now the number of the variables is $d = \binom{N}{3} + \binom{N}{2} + 2N = O(N^3)$.

If we have enough such linear equations such that we can find a $d \times d$ non-singular matrix L satisfying $L\mathbf{y} = \mathbf{b}$, we can find $\mathbf{y}$ by solving the set of linear equations over $\mathbb{F}_2$. In our experiments, we can always find an invertible L. So we can find an $\mathbf{m}'$ such that $\mathbf{m} = \mathbf{m}' \bmod 2$.

Remark 3. Notice that from $\sum_i \bar{\mathbf{c}}_i \mathbf{y}_i = \bar{b} \bmod 2^{k-u}$, we can also get the linear equation: $\sum_i \mathbf{c}_i \mathbf{y}_i = b \bmod 4$, when $k - u > 1$ if we let $\mathbf{c}_i = \bar{\mathbf{c}}_i \bmod 4$ and $b = \bar{b} \bmod 4$. If we have enough linear equations, and use the similar strategy above, we may recover $\mathbf{m}$ directly.

3.3 The Broadcast Attack against NTRU-2001 with an Odd d_g

For NTRU-2001 with an odd d_g, $ES = \{0, 1\}$ and $\mathbf{m} \in \{0, 1\}^N$. As the section above, we can not use the algorithm for LWE directly. We also propose a new algorithm below.

First, from Equation (2), we have $(\sum_{i=1}^N \mathbf{a}_i\mathbf{x}_i - b')(\sum_{i=1}^N \mathbf{a}_i\mathbf{x}_i - b' + 1) = 0 \bmod 2^k$, i.e.

$$\sum_{i=1}^N \mathbf{a}_i^2\mathbf{x}_i^2 + 2\sum_{i<j} \mathbf{a}_i\mathbf{a}_j\mathbf{x}_i\mathbf{x}_j + \sum_{i=1}^N (\mathbf{a}_i - 2b'\mathbf{a}_i)\mathbf{x}_i = -(b')^2 + b' \bmod 2^k.$$

Since $\mathbf{x}_i^2 = \mathbf{x}_i$ holds for $\mathbf{x}_i \in \{0, 1\}$, we have

$$2\sum_{i<j} \mathbf{a}_i\mathbf{a}_j\mathbf{x}_i\mathbf{x}_j + \sum_{i=1}^N ((\mathbf{a}_i^2 + \mathbf{a}_i) - 2b'\mathbf{a}_i)\mathbf{x}_i = -(b')^2 + b' \bmod 2^k.$$

Since $\mathbf{a}_i^2 = \mathbf{a}_i \bmod 2$, there is obviously an integer $u \geq 1$ such that 2^u divides the greatest common divisor of all the coefficients but 2^{u+1} can not. We have

$$\frac{1}{2^{u-1}}(\sum_{i<j} \mathbf{a}_i\mathbf{a}_j\mathbf{x}_i\mathbf{x}_j + \sum_{i=1}^N (\frac{\mathbf{a}_i^2 + \mathbf{a}_i}{2} - b'\mathbf{a}_i)\mathbf{x}_i) = \frac{-(b')^2 + b'}{2^u} \bmod 2^{k-u}.$$

If $u < k$, we assign each of the monomials a new variable $\mathbf{y}_i$, and transform it into a linear equation: $\sum_i \bar{\mathbf{c}}_i \mathbf{y}_i = \bar{b} \bmod 2^{k-1-u}$.

Let $\mathbf{c}_i = \bar{\mathbf{c}}_i \bmod 2$ and $b = \bar{b} \bmod 2$. So we get a linear equation over $\mathbb{F}_2$: $\sum_i \mathbf{c}_i \mathbf{y}_i = b \bmod 2$.

As in Section 3.2, when collecting enough equations in our experiments, we always get a matrix L by performing Gaussian elimination, such that

$$L\mathbf{y} = \left(\begin{array}{cc|c} 1 & \cdots & \\ & \ddots & * \\ \hline & \mathbf{0} & I_{N\times N} \end{array}\right) \left(\begin{array}{c} \mathbf{y}_1 \\ \vdots \\ \hline \mathbf{x} \end{array}\right) = \mathbf{b}.$$

whereas L is not invertible. Nevertheless, it is enough for us to recover $\mathbf{m}$.

What's more, we can also use the strategy below to get an invertible L with very high probability. Notice that $u = 1$ holds for very high probability. We only use those samples in which $u = 1$. So we have

$$\sum_{i<j} \mathbf{a}_i\mathbf{a}_j\mathbf{x}_i\mathbf{x}_j + \sum_{i=1}^N (\frac{\mathbf{a}_i^2 + \mathbf{a}_i}{2} - b'\mathbf{a}_i)\mathbf{x}_i = \frac{-(b')^2 + b'}{2} \bmod 2. \quad (5)$$

Denote by $\mathbf{y}_{(i,j)}$ the variable corresponding to $\mathbf{x}_i\mathbf{x}_j$ $(i < j)$ and by $\mathbf{c}_{(i,j)}$ its coefficient, namely $\mathbf{a}_i\mathbf{a}_j$. We claim that

Proposition 4. *For* $1 \le i \le N-1$, $\mathbf{c}_{(i,N)} = \sum_{k=1}^{i-1} \mathbf{c}_{(k,i)} + \sum_{k=i+1}^{N-1} \mathbf{c}_{(i,k)}$ *mod* 2.

Proof. By Equation (4), we know that $\sum_{k=1}^{N} \mathbf{a}_k = 1 \bmod 2$. So

$$\begin{aligned}
\sum_{k=1}^{i-1} \mathbf{c}_{(k,i)} + \sum_{k=i+1}^{N-1} \mathbf{c}_{(i,k)} &= \sum_{k=1}^{i-1} \mathbf{a}_k\mathbf{a}_i + \sum_{k=i+1}^{N-1} \mathbf{a}_i\mathbf{a}_k && \bmod 2\\
&= \sum_{k=1}^{i-1} \mathbf{a}_k\mathbf{a}_i + \mathbf{a}_i(1 - \sum_{k=1}^{i-1} \mathbf{a}_k - \mathbf{a}_i - \mathbf{a}_N) && \bmod 2\\
&= \sum_{k=1}^{i-1} \mathbf{a}_k\mathbf{a}_i - \sum_{k=1}^{i-1} \mathbf{a}_k\mathbf{a}_i + \mathbf{a}_i - \mathbf{a}_i^2 - \mathbf{a}_i\mathbf{a}_N && \bmod 2\\
&= \mathbf{a}_i\mathbf{a}_N && \bmod 2\\
&= \mathbf{c}_{(i,N)} && \bmod 2
\end{aligned}$$

We involve some new variables: $\bar{\mathbf{y}}_{(i,j)} = \mathbf{y}_{(i,j)} + \mathbf{y}_{(i,N)} + \mathbf{y}_{(j,N)}$, where $1 \le i \le N-2$, $i < j \le N-1$. Since for $1 \le i \le N-1$, $\mathbf{c}_{(i,N)}\mathbf{y}_{(i,N)} = \sum_{j=1}^{i-1} \mathbf{c}_{(j,i)}\mathbf{y}_{(i,N)} + \sum_{j=i+1}^{N-1} \mathbf{c}_{(i,j)}\mathbf{y}_{(i,N)}$ mod 2, we have

$$\sum_{i<j<N} \mathbf{c}_{(i,j)}\bar{\mathbf{y}}_{(i,j)} = \sum_{i<j\le N} \mathbf{a}_i\mathbf{a}_j\mathbf{x}_i\mathbf{x}_j \bmod 2 = \sum_{i<j\le N} \mathbf{c}_{(i,j)}\mathbf{y}_{(i,j)} \bmod 2.$$

Using the new variables $\bar{\mathbf{y}}_{(i,j)}$ $(i < j < N)$ in Equation (5), we can eliminate the old variables $\mathbf{y}_{(i,j)}$ $(i < j \le N)$. Now the number of the variables is $d = \binom{N}{2} + 1 = O(N^2)$.

If we have enough such linear equations such that we can find a $d \times d$ nonsingular matrix L satisfying $L\mathbf{y} = \mathbf{b}$, we can find $\mathbf{y}$ by solving the set of linear equations over $\mathbb{F}_2$. In our experiments, we can always find an invertible L. So we can find $\mathbf{m}$.

3.4 The Broadcast Attack against NTRU-2005

For NTRU-2005, $ES = \{0,1\}$ and $\mathbf{m} \in \{0,1\}^N$. Since q is a prime, so we can use the algorithm for LWE directly. To decrease the number of the variables, we can also use the fact $\mathbf{x}_i^2 = \mathbf{x}_i$ holds for $\mathbf{x}_i \in \{0,1\}$. So we use the following equations

$$2\sum_{i<j} \mathbf{a}_i\mathbf{a}_j\mathbf{x}_i\mathbf{x}_j + \sum_{i=1}^{N}((\mathbf{a}_i^2 + \mathbf{a}_i) - 2b'\mathbf{a}_i)\mathbf{x}_i = -(b')^2 + b' \bmod q.$$

If we have enough such linear equations such that we can find a $d \times d$ nonsingular matrix L satisfying $L\mathbf{y} = \mathbf{b}$, where $d = \binom{N}{2} + N = O(N^2)$, we can find $\mathbf{y}$ by solving the set of linear equations over $\mathbb{F}_q$.

3.5 Analysis of the Attacks

Some Observations. Notice that we can obtain N linear equations from every recipient. If we assume that d linearly independent linear equations can be found from $O(d/N)$ recipients with very high probability, it is easy to show that we need gather $O(N)$ (resp. $O(N^2)$) recipients's information and solve a set of $O(N^2)$ (resp. $O(N^3)$) linear equations to complete the attack against NTRU-2001 with an odd d_g and NTRU-2005 (resp. NTRU-1998). With ordinary Gaussian elimination and multiplication, we have the following result.

Variant	N	d	Recipients	Space	Time
NTRU-1998	N	$O(N^3/6)$	$O(N^2/6)$	$O(N^6/36)$	$O(N^9/768)$
NTRU-2001	N	$O(N^2/2)$	$O(N/2)$	$O(N^4/4)$	$O(N^6/24)$
NTRU-2005	N	$O(N^2/2)$	$O(N/2)$	$O(N^4 \log N/4)$	$O(N^6 \log^2 N/24)$

It remains to discuss the assumption. We have done many experiments and find that d linearly independent linear equations can be found from $O(d/N)$ recipients with probability very close to one. So the broadcast attacks succeed with overwhelming probability. In the appendix, we give the concrete estimates for the attack complexities for the cases used in IEEE 1363.1 Standard [18] based on NTRU 1998, regardless of its padding. Next we give our experimental results.

Experimental Results. All experiments were performed on a Windows XP system with a 3.20 GHz Pentium 4 processor and 2 GByte RAM using Shoup's NTL library version 5.4.1 [24].

We implemented the three instantiations of NTRU and the successful attacks against them, where we find a L invertible. For NTRU-1998 and NTRU-2001, we adopted the algorithms for $u = 1$. In our experiments, we always obtained an invertible L with the recipients whose number is just a little more than or equal to d/N. Some results are listed below:

Variant	N	q	p	d_f	d_g	d_r	d	Recipients	Rank(L)	Result
NTRU-1998	47	32	3	7	8	5	17390	373	17390	Success
NTRU-2001	167	128	$2+x$	60	19	18	13862	88	13862	Success
NTRU-2005	107	97	2	25	24	25	5778	56	5778	Success

All the experiment finish within minutes.

3.6 Improving the Attack

We have some methods to improve the attack.

With the Known Bits. If we know some bits, either the message bits or the random bits, we can also improve the attack.

- If we know some bits of $\mathbf{m}$, for example, $\mathbf{m}_0, \mathbf{m}_2, \cdots, \mathbf{m}_{t-1}$, then we can eliminate those monomials containing at least one of these known bits.
- If we know some bits of $\mathbf{r}$, for example, $\mathbf{r}_0, \mathbf{r}_2, \cdots, \mathbf{r}_{t-1}$, then for $i = 0, \cdots, t-1$, we have t linear equations: $\sum_{j=0}^{N-1} \widehat{H}_{i+1,j+1}\mathbf{m}_j + \mathbf{r}_i - \mathbf{b}_i = 0 \bmod q$. Since

these equations are linear independent, we can represent some t $\mathbf{m}_i$'s by the other $N-t$ variables, hence also eliminate those monomials containing at least one of these t $\mathbf{m}_i$'s.

However, how can we obtain these bits? We next show that "guessing" is a good idea for NTRU as pointed in [19].

For any vector $\mathbf{v} = (\mathbf{v}_0, \mathbf{v}_1, \cdots, \mathbf{v}_{N-1})^T$, we denote by $\mathbf{v}^{(r)}$ its r-cycle:

$$\mathbf{v}^{(r)} = \begin{cases} \mathbf{v}, & r = 0; \\ (\mathbf{v}_{N-r}, \mathbf{v}_{N-r+1}, \cdots, \mathbf{v}_{N-1}, \mathbf{v}_0, \mathbf{v}_1, \cdots, \mathbf{v}_{N-r-1})^T, & r \in \{1, \cdots, N-1\}. \end{cases}$$

For NTRU, since $H\mathbf{r} + \mathbf{m} = \mathbf{c} \bmod q$, we have $H\mathbf{r}^{(i)} + \mathbf{m}^{(i)} = \mathbf{c}^{(i)} \bmod q$ for $i = 0, \cdots, N-1$. We take $\mathbf{r}$ as an example to show how we guess some of its bits. For example, we guess $\mathbf{r}_0 = 0, \mathbf{r}_2 = 0, \cdots, \mathbf{r}_{t-1} = 0$. Then, if there is an i-cycle of $\mathbf{r}$ such that, for j=0,...,t-1, $\mathbf{r}_j^{(i)} = \mathbf{r}_j$, we can use the corresponding equation $H\mathbf{r}^{(i)} + \mathbf{m}^{(i)} = \mathbf{c}^{(i)} \bmod q$ instead of the origin one for any recipient to continue the attack. Of course, we don't know what i is. However, we can commit the attack for every $i \in \{0, 1, \cdots, N-1\}$, then check whether we get the correct message.

By [19], we know that the probability that there is an i-cycle satisfying our need is approximately equal to $1 - (1 - \prod_{j=0}^{d_r-1} (1 - \frac{t}{N-j}))^N$ for $\mathcal{L}(r) = \mathcal{B}(d_r)$, which is very close to 1 for small t. For more analysis see [19].

With the Clue from the Parameters. We can easily get: $h(1)r(1) + m(1) = c(1) \bmod q$, when we take $h(x), r(x), m(x)$ and $c(x)$ as polynomials. Since we know $h(x), c(x)$, and by $\mathcal{L}_r$ we also know that $r(1) = 0$ in NTRU-1998 and $r(1) = d_r$ in NTRU-2001 and NTRU-2005, we have $\sum_{j=0}^{N-1} \mathbf{m}_j + h(1)r(1) - c(1) = 0 \bmod q$. Then we can eliminate a variable. For example, since $\mathbf{m}_0 = h(1)r(1) - c(1) - \sum_{j=1}^{N-1} \mathbf{m}_j \bmod q$, we can substitute $\mathbf{m}_0$ in every equation.

Use the XL or the Mutant XL Algorithms. Our attack can also be further improved with XL and MutantXL algorithms[2,4], or similar Gröbner basis type of algorithms. For instance, in case of NTRU 2001, if we do not have enough recipients to obtain enough ciphertext to generate enough quadratic equations in our attack, we could use the idea of the XL type of algorithm by multiplying the derived equations by monomial of degree one or higher, such that we may solve our system of equations at a higher degree. In this case, our attack could work with much fewer recipients but higher computation cost. The details of this part of work will be left to a subsequent paper.

4 Conclusion

In this paper, we present an algebraic broadcast attack against NTRU. Under a reasonable assumption, the attack can be completed in polynomial time and space. The experiments show that the attack succeed with probability very close to one. This is the first efficient successful broadcast attack against NTRU. However, it is still an open problem to find a more efficient broadcast attack with just a constant number of recipients.

Acknowledgments. The work of J.D. was supported by **Charles Phelps Taft Foundation** and the NNSF of China under the grant #60973131. The work of Y.D. and Y.P. was supported by the NNSF of China (Grants Nos. 11071285, 61121062) and 973 Project (2011CB302401).

References

1. Arora, S., Ge, R.: New Algorithm for Learning in Presence of Errors, `http://www.cs.princeton.edu/~rongge/LPSN.pdf`
2. Buchmann, J., Cabarcas, D., Ding, J., Mohamed, M.S.E.: Flexible Partial Enlargement to Accelerate Gröbner Basis Computation over $\mathbb{F}_2$. In: Bernstein, D.J., Lange, T. (eds.) AFRICACRYPT 2010. LNCS, vol. 6055, pp. 69–81. Springer, Heidelberg (2010)
3. Coppersmith, D., Shamir, A.: Lattice Attacks on NTRU. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 52–61. Springer, Heidelberg (1997)
4. Courtois, N.T., Klimov, A.B., Patarin, J., Shamir, A.: Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 392–407. Springer, Heidelberg (2000)
5. Ding, J.: Solving LWE problem with bounded errors in polynomial time. Cryptology ePrint Archive, Report 2010/558 (2010)
6. Ding, J.: Fast Algorithm to solve a family of SIS problem with l_∞ norm. Cryptology ePrint Archive, Report 2010/581 (2010)
7. Ding, J.: Algebraic solvers for certain lattice-related problems. In: 2011 IEEE Information Theory Workshop (ITW), pp. 405–409. IEEE Conference Publications (2011)
8. Gama, N., Nguyên, P.Q.: New Chosen-Ciphertext Attacks on NTRU. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 89–106. Springer, Heidelberg (2007)
9. Hästad, J.: Solving simultaneous modular equations of low degree. SIAM J. Comput. 17, 336–341 (1988)
10. Hoffstein, J., Silverman, J.H.: Implementation Notes for NTRU PKCS Multiple Transmissions, Report #6, NTRU Technical Reports, `http://www.securityinnovation.com/cryptolab/pdf/NTRUTech006.pdf`
11. Hoffstein, J., Silverman, J.H.: Optimizations for NTRU. Technical report, NTRU Cryptosystems (June 2000), `http://citeseer.ist.psu.edu/693057.html`
12. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A Ring-Based Public Key Cryptosystem. In: Buhler, J.P. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 267–288. Springer, Heidelberg (1998)
13. Howgrave-Graham, N.: A Hybrid Lattice-Reduction and Meet-in-the-Middle Attack Against NTRU. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 150–169. Springer, Heidelberg (2007)
14. Howgrave-Graham, N., Nguyên, P.Q., Pointcheval, D., Proos, J., Silverman, J.H., Singer, A., Whyte, W.: The Impact of Decryption Failures on the Security of NTRU Encryption. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 226–246. Springer, Heidelberg (2003)

15. Howgrave-Graham, N., Silverman, J.H., Whyte, W.: A Meet-In-The-Meddle Attack on an NTRU Private Key. Technical Report, http://www.ntru.com/cryptolab/technotes.htm#004
16. Howgrave-Graham, N., Silverman, J.H., Whyte, W.: Choosing Parameter Sets for NTRUEncrypt with NAEP and SVES-3. Technical Report, NTRU Cryptosystems (2005)
17. Hirschhorn, P.S., Hoffstein, J., Howgrave-Graham, N., Whyte, W.: Choosing NTRUEncrypt Parameters in Light of Combined Lattice Reduction and MITM Approaches. In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536, pp. 437–455. Springer, Heidelberg (2009)
18. IEEE. P1363.1 Public-Key Cryptographic Techniques Based on Hard Problems over Lattices. IEEE (June 2003), http://grouper.ieee.org/groups/1363/lattPK/index.html
19. May, A., Silverman, J.H.: Dimension Reduction Methods for Convolution Modular Lattices. In: Silverman, J.H. (ed.) CaLC 2001. LNCS, vol. 2146, pp. 110–125. Springer, Heidelberg (2001)
20. Mol, P., Yung, M.: Recovering NTRU Secret Key from Inversion Oracles. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 18–36. Springer, Heidelberg (2008)
21. Nguyên, P.Q., Pointcheval, D.: Analysis and Improvements of NTRU Encryption Paddings. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 210–225. Springer, Heidelberg (2002)
22. Plantard, T., Susilo, W.: Broadcast Attacks against Lattice-Based Cryptosystems. In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536, pp. 456–472. Springer, Heidelberg (2009)
23. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Johnson, D.S., Feige, U. (eds.) Proc. of 37th STOC, pp. 84–93. ACM (2005)
24. Shoup, V.: NTL: A library for doing number theory, http://www.shoup.net/ntl/

A Some Results for IEEE 1363.1 Standard

Since IEEE 1363.1 Standard is based on NTRU-1998, we give our estimation for the cost of our broadcast attack on the schemes without the padding.

Parameter Set	N	d	Recipients(at least)	Space(bits $\approx$)	Time ($\approx$)
ees401ep1	401	10747600	26802	2^{47}	2^{69}
ees541ep1	541	26391100	48782	2^{50}	2^{73}
ees659ep1	659	47699700	72382	2^{52}	2^{74}
ees449ep1	449	15087300	33602	2^{48}	2^{70}
ees613ep1	613	38392200	62630	2^{51}	2^{73}
ees761ep1	761	73453200	96522	2^{53}	2^{77}
ees677ep1	677	51716000	76390	2^{52}	2^{76}
ees887ep1	887	116312000	131130	2^{54}	2^{79}
ees1087ep1	1087	214063000	196930	2^{56}	2^{82}
ees1087ep2	1087	214063000	196930	2^{56}	2^{82}
ees1171ep1	1171	267623000	228542	2^{56}	2^{83}
ees1499ep1	1499	561378000	374502	2^{59}	2^{85}

Analysis of Indirect Message Injection for MAC Generation Using Stream Ciphers

Mufeed ALMashrafi[1,2], Harry Bartlett[1,2], Leonie Simpson[1,2], Ed Dawson[1,2], and Kenneth Koon-Ho Wong[1]

[1] Information Security Institute,
almashrafi@student.qut.edu.au
[2] Science and Engineering Faculty,
Queensland University of Technology,
GPO Box 2434, Brisbane Qld 4001, Australia
{h.bartlett,lr.simpson,e.dawson,kk.wong}@qut.edu.au

Abstract. This paper presents a model for generating a MAC tag with a stream cipher using the input message indirectly. Several recent proposals represent instances of this model with slightly different options. We investigate the security of this model for different options, and identify cases which permit forgery attacks. Based on this, we present a new forgery attack on version 1.4 of 128-EIA3. Design recommendations to enhance the security of proposals following this general model are given.

Keywords: MAC, Stream ciphers, Forgery attacks, Grain-128a, Sfinks, 128-EIA3.

1 Introduction

A Message Authentication Code (MAC) is used to provide assurance of the integrity of a message, and requires use of an algorithm along with a secret key. MACs are commonly formed using block ciphers in certain modes, and keyed hash functions. Recently algorithms to construct MACs using stream ciphers have been proposed. Some proposals using symmetric ciphers claim to provide both confidentiality and integrity assurance; these are referred to as Authenticated Encryption (AE).

There is extensive research in the existing literature on generating MAC tags using block ciphers, but much less on generating MAC tags using stream ciphers. This may be due to the existence of block cipher standards (DES, AES) with well known modes of operation. However, the potential for increased speed and a smaller footprint in hardware (and often software), makes stream cipher based MACs worth considering.

Lai *et al.* [10] proposed a cryptographic checksum algorithm based on stream ciphers. Golić also describes a mode of operation to generate a MAC using a stream cipher [8]. More recently, seven of the thirty-four stream cipher proposals submitted to eSTREAM [6] claimed to provide AE. The current NIST cryptographic hash competition received sixty-four submissions; seven of these use stream cipher algorithms to generate a hash value. Although flaws exist in these proposals, resulting in none of them being considered as finalists in the respective competitions, the proposals demonstrate increasing interest in MAC generation using stream ciphers.

Nakano *et al.* [12] proposed a general model for generating a MAC tag using stream ciphers by injecting the input message directly into the internal states of the ciphers. Their security analysis suggests that the message injection function is critical for achieving collision resistance for MACs which fit the model. This relates to [11], in which two methods for direct message injection into the internal states of the ciphers are examined.

In this paper we analyse proposals for MAC generation which use stream ciphers in a way that is not considered in the Nakano model [12]. Some existing stream cipher proposals do not incorporate the input message directly into the internal state of the cipher. Instead, they use the input message to control the compression of a bitstream from the keystream generator. We propose a general model for generating a MAC tag using stream ciphers in this manner, which we refer to as indirect message injection. We consider possible options at several stages of MAC generation under this model, and analyse the security implications associated with these options with respect to forgery attacks.

The paper is organized as follows. Section 2 outlines the phases in generating a MAC tag, and discusses the forgery attacks on this model. Section 3 describes our general model for generating a MAC using a stream cipher with indirect message injection. Three specific stream cipher algorithms that generate a MAC tag in this way (128-EIA3 [4,5], Grain-128a [1] and Sfinks [2]) are examined in section 4. Our security analysis is presented in section 5, and concluding remarks are contained in section 6.

2 MAC Generation

A MAC algorithm takes three inputs: an arbitrary length message M of length l bits, a k-bit secret key K and a v-bit initialisation vector IV, and produces a d-bit MAC tag.

2.1 MAC Generation Phases

The generation of a MAC tag usually involves three phases: preparation, accumulation and finalization. The **preparation phase** involves initializing the internal states of the integrity components of the device, and preparing the input message. Message preparation may involve adding padding bits to either end of the message M. The **accumulation phase** is where the input message is processed and values are accumulated in the internal states of the integrity components. The **finalization phase** completes the processing of the MAC tag, usually by combining the stored value at the end of the accumulation phase with a mask.

2.2 Forgery Attacks on MAC Generation

Consider the possibility of a forgery attack being conducted as follows. Suppose for a message M, a MAC tag $MAC_{K,IV}(M)$ is generated using key K and IV. The sender intends to transmit the message-MAC tag pair to a particular receiver. Assume a man in the middle attacker intercepts the message-MAC tag pair, and can modify M and possibly also $MAC_{K,IV}(M)$ to calculate a valid MAC tag $MAC_{K,IV}(M')$ for a modified message M'. The attacker then sends the new pair $(M', MAC_{K,IV}(M'))$ to the intended recipient. If it is possible to alter M to M' and provide a valid $MAC_{K,IV}(M')$ without any knowledge of the keystream sequences used to generate $MAC_{K,IV}(M)$, the forgery attack is successful. In this paper, we investigate the possibility of such forgeries for MACs formed by indirect message injection, for modifications involving flipping, deleting or inserting bits in the original message M.

3 MAC Generation Using Indirect Message Injection

Common structures for MAC algorithms use the message directly and accumulate message values in a component of the device. Some recent AE stream cipher proposals use a different strategy. These proposals use the input message to control the accumulation of a keystream sequence, with the accumulated value forming the basis of the MAC tag. In this section we describe explicitly a model for generating a MAC tag using stream ciphers in this manner.

3.1 General Structure for AE Algorithms

The keystream generator of a stream cipher takes as inputs a secret key K and a public IV, and generates a pseudorandom binary sequence. Usually these sequences are used as keystreams for binary additive stream ciphers to provide confidentiality for plaintext messages. However, they can also be used for integrity applications. Where authenticated encryption is required, the sequence used for the integrity application may be produced by the same generator as the sequence used for the confidentiality application, but a different keystream generator could also be used. Figure 1(a) shows a single keystream generator using K and an IV to produce two different binary sequences, z_t and y_t, used for confidentiality and integrity applications, respectively. If one generator is used to produce both sequences, then we assume that the two sequences are distinct. Examples of such algorithms include Sfinks [2] and Grain-128a [1]. Figure 1(b) shows an alternative case where the sequences z_t and y_t are generated by separate keystream generators, with distinct keys and IVs for each: K_C, IV_C, K_I and IV_I, respectively. An example of such algorithms is ZUC [4], used in 128-EIA3. In this paper our focus is on the integrity component of the designs; the part inside the dashed lines in Figure 1 (for both (a) and (b)). In this figure, we also use dashed lines from the output ciphertext to the integrity component of the algorithm to show that either plaintext or ciphetext could be used by this model to control the segment of keystream to be accumulated.

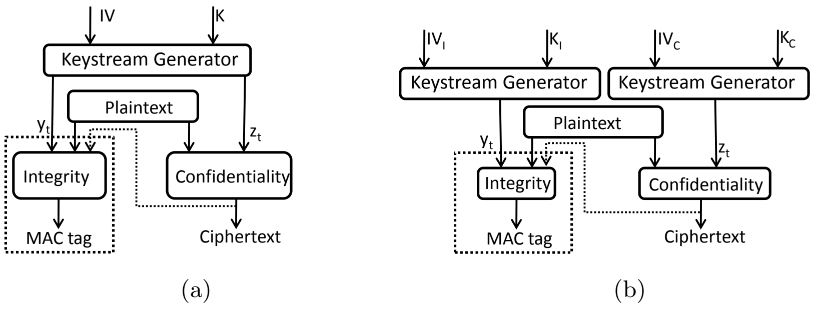


Fig. 1. General structure for AE algorithms

3.2 Structure of the Integrity Algorithm

The integrity component in Figure 1 consists of two registers, each of d binary stages. The relationship between these two registers, the keystream sequence and message M is shown in Figure 2.

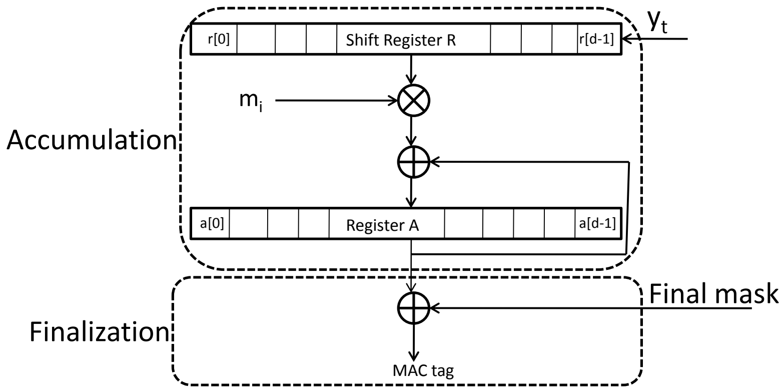


Fig. 2. MAC generation using indirect message injection

The first register is a binary shift register, denoted R. Let R_t denote the contents of R at time t, with $R_t = (r_t[0], \ldots, r_t[d-1])$ and initial contents $R_0 = (r_0[0], \ldots, r_0[d-1])$. At each time clock t, for $t > 0$, the contents of R are updated using the binary sequence y as follows:

$$r_t[i] = \begin{cases} r_{t-1}[i+1], & \text{for } i = 0, \ldots, d-2 \\ y_{t-1}, & \text{for } i = d-1 \end{cases} \tag{1}$$

Thus the contents of register R can be considered as a "sliding window" of length d on the sequence $R_0||y$, where $||$ denotes concatenation.

The second register is an accumulation register, denoted A. Let A_t denote the contents of A at time t, with $A_t = (a_t[0], \ldots, a_t[d-1])$ and initial contents

$A_0 = (a_0[0], \ldots, a_0[d-1])$. The contents of A are updated using the contents of R, conditional on the value of the input message bit m_t, according to Equation 2. If $m_t = 1$ then A is updated by XORing the current contents with the contents of R at time t; otherwise A remains unchanged:

$$A_t = \begin{cases} A_{t-1} \oplus R_{t-1}, & \text{if } m_t = 1 \\ A_{t-1}, & \text{otherwise} \end{cases} \tag{2}$$

The input message M controls which d-bit segments of the sequence $R_0||y$ are accumulated into register A. When the i^{th} message bit has been processed, the value in register A can be expressed as $A_i = A_0 \oplus T_i\, M_i$, where M_i consists of the first i bits of the message M and T_i is a $d \times i$ matrix such that the j^{th} column of T_i contains R_j, for $j = 0, \ldots, i-1$; that is:

$$\begin{aligned} A_i &= A_0 \oplus T_i\, M_i \\ &= \begin{pmatrix} a_0[0] \\ a_0[1] \\ \vdots \\ a_0[d-1] \end{pmatrix} \oplus \begin{pmatrix} r_0[0] & r_0[1] \ldots r_0[d-1] & y_0 & \ldots y_{i-d-1} \\ r_0[1] & r_0[2] \ldots \quad y_0 & y_1 & \ldots \quad y_{i-d} \\ \vdots & \vdots \quad \iddots \quad \vdots & \vdots & \iddots \quad \vdots \\ r_0[d-1] & y_0 \quad \ldots \quad y_{d-2} & y_{d-1} & \ldots \quad y_{i-2} \end{pmatrix} \begin{pmatrix} m_0 \\ m_1 \\ \vdots \\ m_{i-1} \end{pmatrix}. \end{aligned}$$

Note also that each row of T_i consists of i consecutive bits from the sequence $R_0||y$ and is closely related to both the preceding and following rows.

When all of the message bits have been processed, the MAC finalization phase begins. This involves combining the final contents of A with a masking value $F = (f[0], \ldots, f[d-1])$. Thus, for a message M of length l we have

$$MAC(M_l) = A_l \oplus F = A_0 \oplus T_l\, M_l \oplus F \tag{3}$$

3.3 Optional Processes

For the general model using indirect message injection in the accumulation phase, as shown in Figure 2, we consider several options for the preparation and finalization phases, respectively.

In the preparation phase, options relate to the initialization of the two registers R and A, and to preparation of the message. Either register could be initialized with fixed values, such as all zeroes; or with key dependent values, such as a segment from the keystream sequence y. The input message could be padded, by appending a specified sequence of bits at either the beginning, the end, or at both places. Alternatively no padding could be applied. If M_l is padded with n bits, we use M_p to denote the padded message, where $p = l + n$.

In the finalization phase, the final contents of accumulation register A are combined with a mask, as shown in Figure 2. The mask may be obtained from the sequence used for the accumulation phase, y, or from another sequence. For AE this may be z the sequence used for the confidentiality application. In some cases a null mask (all zero values) is used.

4 Current Proposals Using This Model

We examine the MAC generation process for algorithms that use the model presented in section 3.2, but with slight differences in the options applied. As each uses the same accumulation process, we describe only the options taken for the preparation and finalization phases for these ciphers.

4.1 128-EIA3 Version 1.4

Version 1.4 of the 128-EIA3 [3] integrity algorithm uses the ZUC stream cipher [5] as a keystream generator. ZUC is a word-based stream cipher with word size 32 bits, that uses 128-bit secret key and 128-bit IV. The 128-EIA3 MAC tag has length $d = 32$ bits.

Preparation Phase. 128-EIA3 does not use physical registers for R and A but they use instead variables k and T to represent these registers respectively. Register A is initialized with all zero values and register R is initialized with keystream sequence. Let $(y_{-32}, y_{-31}, \ldots, y_{-1})$ denote the first 32 bits of the keystream y produced; then

$$R_0 = \begin{pmatrix} y_{-32} \\ y_{-31} \\ \vdots \\ y_{-1} \end{pmatrix}.$$

The input message M_l is padded by adding one bit at the end of the message, so $M_p = M_l || 1 = (m_0, \ldots, m_{l-1}, 1)$.

Finalization Phase. After all $l+1$ bits of the padded message have been processed, the contents of A at time $l+1$ are combined with the final mask, a 32-bit segment from y starting at y_{l+32}. The final MAC tag is given by:

$$MAC(M_p) = \begin{pmatrix} y_{-32} & y_{-31} & \cdots & y_{-1} & y_0 & \cdots & y_{l-32} \\ y_{-31} & y_{-30} & \cdots & y_0 & y_1 & \cdots & y_{l-31} \\ \vdots & \vdots & .\cdot^{\cdot} & \vdots & \vdots & .\cdot^{\cdot} & \vdots \\ y_{-2} & y_{-1} & \cdots & y_{29} & y_{30} & \cdots & y_{l-2} \\ y_{-1} & y_0 & \cdots & y_{30} & y_{31} & \cdots & y_{l-1} \end{pmatrix} \begin{pmatrix} m_0 \\ m_1 \\ \vdots \\ m_{l-1} \\ 1 \end{pmatrix} \oplus \begin{pmatrix} y_{l+32} \\ y_{l+33} \\ \vdots \\ y_{l+62} \\ y_{l+63} \end{pmatrix}.$$

4.2 128-EIA3 Version 1.5

Version 1.5 of 128-EIA3 was proposed in response to a successful forgery attack [7] on version 1.4. We discuss this attack in more detail in section 5. This version is identical to version 1.4 except in the starting position of the final mask, obtained from the sequence y. Instead of starting at bit number $(l+32)$, the mask in version 1.5 starts at the beginning of the next 32-bit word of the bitstream. The final MAC tag is given by:

$$MAC(M_p) = \begin{pmatrix} y_{-32} & y_{-31} & \cdots & y_{-1} & y_0 & \cdots & y_{l-32} \\ y_{-31} & y_{-30} & \cdots & y_0 & y_1 & \cdots & y_{l-31} \\ \vdots & \vdots & \iddots & \vdots & \vdots & \iddots & \vdots \\ y_{-2} & y_{-1} & \cdots & y_{29} & y_{30} & \cdots & y_{l-2} \\ y_{-1} & y_0 & \cdots & y_{30} & y_{31} & \cdots & y_{l-1} \end{pmatrix} \begin{pmatrix} m_0 \\ m_1 \\ \vdots \\ m_{l-1} \\ 1 \end{pmatrix} \oplus \begin{pmatrix} y_{(\lceil l/32 \rceil+1)*32} \\ y_{((\lceil l/32 \rceil+1)*32)+1)} \\ \vdots \\ y_{((\lceil l/32 \rceil+1)*32)+30)} \\ y_{((\lceil l/32 \rceil+1)*32)+31)} \end{pmatrix}.$$

4.3 Grain-128a

Grain-128a [1] is a bit-based cipher from the Grain family [9] with an added authentication mechanism. Grain-128a uses a 128-bit secret key and a 96-bit IV, and generates two sequences, z_t and y_t, used for confidentiality and integrity applications, respectively. The MAC tag has length $d = 32$ bits.

Preparation Phase. Both R and A are initialized directly from the Grain-128a bitstreams. Let $(y_{-64}, y_{-63}, \ldots, y_{-33}, y_{-32}, \ldots, y_{-1})$ denote the first 64 bits of the keystream y produced; then

$$A_0 = \begin{pmatrix} y_{-64} \\ y_{-63} \\ \vdots \\ y_{-33} \end{pmatrix} \text{ and } R_0 = \begin{pmatrix} y_{-32} \\ y_{-31} \\ \vdots \\ y_{-1} \end{pmatrix}.$$

The input message is padded with a single bit of value 1, so $M_p = M_l||1$.

Finalization Phase. After all $l + 1$ bits of the padded message have been processed, the contents of A at time $l + 1$ represents the final MAC tag. That is, the final mask is a null mask.

$$MAC(M_p) = \begin{pmatrix} y_{-64} \\ y_{-63} \\ \vdots \\ y_{-33} \end{pmatrix} \oplus \begin{pmatrix} y_{-32} & y_{-31} & \cdots & y_{l-32} \\ y_{-31} & y_{-30} & \cdots & y_{l-31} \\ \vdots & \vdots & \iddots & \vdots \\ y_{-1} & y_0 & \cdots & y_{l-1} \end{pmatrix} \begin{pmatrix} m_0 \\ m_1 \\ \vdots \\ m_{l-1} \\ 1 \end{pmatrix}.$$

4.4 Sfinks

The Sfinks [2] stream cipher proposal, submitted to eSTREAM [6], includes an authentication mechanism. The bit-based keystream generator uses an 80-bit secret key and an 80-bit IV, to form an initial state for the 256 bits LFSR, which is the major component of the keystream generator. Nonlinear filters are applied to the contents of the LFSR to produce two different sequences z and y, used for confidentiality and integrity applications, respectively. In addition, two 64-bit registers are used in the authentication mechanism in the manner shown in Figure 2. The Sfinks MAC tag has length $d = 64$ bits.

Preparation Phase. Before processing the message, both R and A are set to all zero and an initialization algorithm is used to incorporate the first 128 bits of keystream $(y_{-128}, y_{-127}, \ldots, y_{-1})$ into the initial values of these registers. This algorithm consists of updating the two registers R and A 128 times according to Equations 1 and 2 but with $m_i = 1$ for $-127 \leq i \leq 0$. Note that this process is equivalent to padding the message at the beginning by concatenating with a sequence of 128 ones. That is $M_p = (1, 1, \ldots, 1)||(m_0, \ldots, m_{l-1})$.

Finalization Phase. The final contents of register A are combined (by XOR-ing) with a final mask that comprises 64 consecutive bits from the confidentiality sequence z, beginning immediately after the segment used to encrypt the input message. The final MAC tag is:

$$MAC(M_p) = \begin{pmatrix} 0 & 0 & \ldots & 0 & y_{-128} & \ldots & y_{l-65} \\ 0 & 0 & \ldots & y_{-128} & y_{-127} & \ldots & y_{l-64} \\ \vdots & \vdots & \iddots & \vdots & \vdots & \iddots & \vdots \\ 0 & 0 & \ldots & y_{-67} & y_{-66} & \ldots & y_{l-3} \\ 0 & y_{-128} & \ldots & y_{-66} & y_{-65} & \ldots & y_{l-2} \end{pmatrix} \begin{pmatrix} 1 \\ \vdots \\ 1 \\ m_0 \\ \vdots \\ m_{l-1} \end{pmatrix} \oplus \begin{pmatrix} z_l \\ z_{l+1} \\ z_{l+2} \\ \vdots \\ z_{l+62} \\ z_{l+63} \end{pmatrix}.$$

5 Forgery Attacks

Generating a MAC tag using the input message indirectly is simple and fast, as the accumulation phase makes repeated use of the XOR operation. In this section, we analyze the security provided by MACs generated using this model with respect to a man in the middle forgery attack as described in section 2.2. In our analysis we assume that the binary sequences produced by the keystream generators are pseudo-random and cannot be distinguished from truly random sequences.

Our analysis considers particularly the options for the preparation and finalization phases, and explores the security implications of particular choices. In most cases, modifying bits between the ends of a non-zero message changes which segments of the (unknown) pseudo-random sequence are accumulated into register A, so this should not lead directly to forgery attacks. Instead we consider forgeries related to the modification of bits at the ends of the message M. Recall from Equation 3 that the MAC tag $MAC_{K,IV}(M_l)$ takes the form $MAC_{K,IV}(M_l) = A_0 \oplus T_l\, M_l \oplus F = (T_l\, M_l) \oplus (A_0 \oplus F)$. Note that the final MAC tag is simply the XOR combination of the separate effects of the accumulation process $T_l\, M_l$ and the masking vector $A_0 \oplus F$. In our analysis, we first consider the security of the accumulation process $T_l\, M_l$, and then the effect of the masking elements $A_0 \oplus F$.

5.1 Security of the Accumulation Process

Consider a message M of length l, with no padding. Let the vector $X = (x_0, x_1, \ldots, x_{d-1})$ denote the output of the accumulation process. We represent the accumulation process in matrix form as follows:

$$X(M_l) = T_l\, M_l = \begin{pmatrix} r_0[0] & r_0[1] \dots r_0[d-1] & y_0 & \dots y_{l-d-1} \\ r_0[1] & r_0[2] \dots \quad y_0 & y_1 & \dots \ y_{l-d} \\ \vdots & \vdots \ \ \iddots \quad \vdots & \vdots & \iddots \quad \vdots \\ r_0[d-1] & y_0 \ \ \dots \ \ y_{d-2} & y_{d-1} & \dots \ y_{l-2} \end{pmatrix} \begin{pmatrix} m_0 \\ m_1 \\ \vdots \\ m_{l-1} \end{pmatrix}.$$

Where a message M is modified to obtain a new message M', we will use the notation $X' = (x'_0, x'_1, \dots, x'_{d-1})$ to refer to $X(M')$.

Bit Flipping Forgeries. Consider firstly the case where R is initialized with zero values. Then all elements in the first column of matrix T_l are zero. Now m_0, the first bit of M_l, has no effect on the value of $X(M_l)$. Thus it is possible to modify M_l by flipping m_0. Let $\overline{m}_i$ denote the complement of m_i. The output X' for the modified message $M' = (\overline{m}_0, \dots, m_{l-1})$ is exactly the same as $X(M_l)$. This collision clearly leads directly to a forgery; the attacker can provide a valid X' for M' with probability of 1.

Similarly, since all elements of the second column are zero except possibly y_0, it follows that message bit m_1 only affects bit x_{d-1} of $X(M_l)$. Modifying M_l by flipping m_1 requires the attacker to guess only x'_{d-1} to construct X' for the modified message. For the modified message $M' = (m_0, \overline{m}_1, \dots, m_{l-1})$, the probability that $X(M')$ will be exactly the same as $X(M_l)$ is therefore 0.5. Similarly, for a message M'' modified in the first two bit positions, $M'' = (\overline{m}_0, \overline{m}_1, \dots, m_{l-1})$, the probability that $X(M'')$ collides with $X(M_l)$ is also 0.5. In general, if we flip bit m_i and any of the bits up to m_i in the original message, for $0 \leq i \leq d-1$, then the probability of collision is 2^{-i}.

Consider now the case where R_0 is known and of the form $(1, 0, \dots, 0)$. Then all elements in the first column of matrix T_l are zero except $r_0[0] = 1$. Now m_0, the first bit of M_l, affects only bit x_0 of $X(M_l)$. Thus it is possible to modify M_l by flipping m_0 and to provide a valid X' for $M'_l = (\overline{m}_0, m_1, \dots, m_{l-1})$, by flipping x_0. That is, $X' = (\overline{x}_0, x_1, \dots, x_{d-1})$. Thus an attacker can produce a valid MAC for the forged message M' with probability of 1.

Similarly, for $R_0 = (1, 1, 0, \dots, 0)$, all elements of the first column are zero except $r_0[0]$ and $r_0[1]$, and in the second column all elements are zero except $r_0[1]$ and possibly y_0. Thus message bit m_0 affects positions x_0 and x_1 of $X(M_l)$, while message bit m_1 affects x_0 and possibly x_{d-1}. Modifying M_l by flipping both m_0 and m_1 requires the attacker to flip only x_1 to form x'_1 and to guess x'_{d-1}. Therefore, the probability that an attacker can construct a valid vector X' for $M'' = (\overline{m}_0, \overline{m}_1, \dots, m_{l-1})$ is 0.5.

In general, for any known R_0, if we flip bit m_i and any of the bits up to m_i in the original message, for $0 \leq i \leq d-1$, then we can construct a valid X' for this modified message M' by flipping the required bits in X and guessing the final i bits in vector X'. Therefore, the probability that an attacker can construct a valid vector X' for M' is 2^{-i}.

Resistance against this type of forgery attack can be provided in two ways. Firstly, we could initialize register R using key dependent values, such as a segment from the keystream sequence, y. Alternatively, the message M_l may be padded by concatenating with a segment of all ones, so that $M_p = (1, 1, \ldots, 1)||M_l$. The padding should consist of at least d ones so that all message bits affect all bits in X, and hence all bits in the final MAC tag, in an unpredictable manner. Note that a key dependent initialisation of R may be the more efficient approach, as padding the message increases both the length of the message and the size of the matrix T, requiring at least $(d + l)$ operations to generate the final MAC tag.

Bit Deletion Forgeries. Suppose we modify M by deleting m_0 to obtain $M'_{l-1} = (m_1, m_2, \ldots, m_{l-1})$. Then the matrix T_{l-1} for M'_{l-1} is just the matrix T_l for M_l without the last column of T_l. Note in Equation 1, for $1 \leq i \leq d-1$, that row i in matrix T is row $i-1$ shifted one position to the left, and with a new value as the final element in the row. Applying this to $X = T_l\ M_l$ and $X' = T_{l-1}\ M'_{l-1}$, it follows that $x_i = r_0[i]m_0 + x'_{i+1}$, for $0 \leq i \leq d-2$.

Now consider the case where R is initialized with zero values. Then, all elements in the first column of matrix T_l are zero and hence $x_i = x'_{i+1}$ for $i = 0, \ldots, d-2$; that is, $X' = (\beta, X \gg 1)$ for some unknown β. We call this the sliding property of the product $T_i\ M_i$. An attacker can guess β, and hence provide a valid X' for M' with probability of 0.5. Similarly, an attacker can form a message M'' by deleting i bits from the beginning of the message M_l, and obtain the new $X''_{l-i} = (\beta_0, \beta_1, \ldots, \beta_{i-1}, X \gg i)$, where the bits $\beta_0, \beta_1, \ldots, \beta_{i-1}$ must be guessed by the attacker. The attacker will provide a valid X for M''_{l-i} with probability 2^{-i}, for $1 \leq i \leq d$. Note that for $i = d$, this is effectively a brute force attack on X, so this attack is only effective for deletion of up to the first $d-1$ bits of the message. This attack can also be adapted for the case when R_0 is non-zero but known; all that is required is to flip appropriate bits of X, as described earlier in this section, before shifting X and guessing $\beta_0, \beta_1, \ldots, \beta_{i-1}$.

Suppose now that R_0 is unknown but that the first j bits of the message are known to be zeroes. We can again delete the first $i \leq j$ bits of the input message, shift X by i times and guess $\beta_0, \beta_1, \ldots, \beta_{i-1}$ to get a valid X' for the modified message. The attacker will provide a valid X' for M'_{l-i} with probability 2^{-i}, for $1 \leq i \leq d$.

The attacks discussed above can all be prevented by either padding the message at the start with at least d ones or by initialising R with unknown bits and padding the message at the start with a single one.

Now suppose that we try deleting bits from the end of a message. Assume $l > d$, if we delete the last bit of M, then $x_i = x'_i + y_{(l-d+i-1)}\ m_{l-1}$, for $0 \leq i \leq d-1$. If $m_{l-1} = 1$, the second term is unknown since it involves keystream bits, but if $m_{l-1} = 0$, then $X' = X$, giving rise to a potential MAC forgery. The same argument clearly applies for deleting any number of zeroes from the message. Such forgeries can be prevented by padding the message with a final one.

Bit Insertion Forgeries. Suppose we modify M_l by appending additional zero bits to the end of the message to obtain $M'_{l+n} = M||(0, \ldots, 0)$. In our matrix representation, adding n zeroes to the end of M_l requires adding n columns to matrix T. During the accumulation process, regardless of the values in these columns, multiplying by the additional message bits of value zero does not change the value of X. Hence $X(M'_{l+n}) = X(M_l)$, for any $n > 0$. Again, this collision leads to an obvious forgery attack which succeeds with probability 1.

Now consider the effects of modifying M_l by inserting an additional bit of value 1 at the end of the message. During the accumulation process, the additional column of T will be multiplied by the additional message bit of value 1 and this may change the values in X. An attacker must guess d elements in that additional column in order to calculate a valid X. So the probability of obtaining a valid X for this modified message is the same as the probability of brute force attack on the MAC. Therefore forgery attacks consisting of inserting zero bits at the end of the message can be prevented if we pad the message with a bit of value 1 at the end. An equivalent solution, which we discuss in the following section, is to use a masking term that depends on the message length.

Now suppose we modify M_l by adding a zero bit to the beginning of the message; that is, $M'_{l+1} = 0||M$. From the structure of T_l and T_{l+1}, it follows that $x'_i = r_0[i]0 + x_{i+1}$, for $0 \leq i \leq d-2$. That is $x'_i = x_{i+1}$ for $i = 0, \ldots, d-2$, so $X' = (X \lll 1, \alpha)$ for some unknown value α. An attacker can guess α, and hence provide a valid X' for M' with probability of 0.5. Similarly, an attacker can form a message M''_{l+i} by adding i zeroes to the beginning of message M_l, and obtain the new $X''_{l+i} = (X \lll i, \alpha_0, \alpha_1, \ldots, \alpha_{i-1})$, where the bits $\alpha_0, \alpha_1, \ldots, \alpha_{i-1}$ must be guessed by the attacker. The attacker will provide a valid X for M''_{l+i} with probability 2^{-i}, for $1 \leq i \leq d$. This is the basis of the previously reported attack on 128-EIA3 version 1.4 [7].

If R is initialised with zeroes, then this attack will work for inserted bits of either value, since the first i bits of M''_{l+i} are all multiplied by zeroes in the first $d - i + 1$ rows of T_{l+i}. Therefore, the inserted bits affect only the last $i - 1$ bits of X''_{l+i}, which are bits that must be guessed anyway. Further, this forgery can again be adapted to the case of R_0 known (but not necessarily zero), since the effects of any inserted bits of value 1 can be determined and allowed for in applying the attack.

From the above discussion, it follows that attacks involving the insertion of bits at the start of a message can be prevented by ensuring that R_0 is initialized with unknown values (keystream) and that the start of the message is padded with at least one bit of value 1.

5.2 Security Considerations for the Masking Vector $A_0 \oplus F$

The forgeries discussed in section 5.1 can all be prevented by suitable choices of R_0 and of message padding; specifically, by initialising R with keystream bits and padding the message with a bit of value 1 at both ends. The masking vector $A_0 \oplus F$ provides an alternative method of preventing many of these forgeries. We now discuss the security implications associated with various options for this term. If this term is to contribute to the security of the MAC, it is important that its contents are unknown to the attacker. Therefore at least one of A_0 and F must be sourced from keystream.

If R_0 is known and there is no message padding, the accumulation term $X = T_l\, M_l$ is vulnerable to attacks involving insertion or deletion of zeroes at the end of the message and to insertion or deletion of bits at the start of the message. (If R_0 is unknown, only the attacks involving insertion or deletion of zeroes apply.) Forgeries involving insertions or deletions at the start of a message rely on the sliding property of $T_l\, M_l$ described in section 5.1. These can be prevented by using an appropriate mask. It is important that changes in the length of the message do not result in corresponding changes in the position of the mask bits. Otherwise the sliding property in the accumulation process will apply to the MAC as a whole. The easiest way to satisfy this requirement is to initialise A with bits from a fixed position such as the start of the keystream sequence y.

Conversely, forgeries involving zeroes inserted or deleted at the end of the message rely on the fact that the additional bits have no effect on the accumulated value X. Such forgeries can be prevented by using an unknown mask that depends on the message length, for example by populating F with a sequence of keystream bits starting at a fixed distance from the last bit used in the accumulation process. Together, these choices for A_0 and F provide an effective alternative to message padding as a means of preventing bit insertion and deletion attacks. Note, however, that the masking term $A_0 \oplus F$ cannot prevent attacks based on flipping bits of the message.

5.3 Security Analysis of Existing Ciphers

In this section, we show how the previous attack on version 1.4 of 128-EIA3 works using our model and then we extend this attack to a new attack on the same version of this algorithm. Then we investigate the security provided by the existing ciphers that follow this model.

Previous Attack on 128-EIA3 Version 1.4 [7]. Recall that for 128-EIA3, $A_0 = (0, \ldots, 0)$, so the masking value is merely the value of F, that is, the 32 consecutive bits of y starting a fixed distance after the last bit used in the accumulation process. Any increase or decrease in the message length therefore causes a corresponding shift in the keystream bits from y used to form F.

Now consider inserting a zero at the start of the message. We noted in section 5.1 that the result X' of the accumulation process for this modified message is related to the original value of X by the sliding relationship $X' = (X \ll 1, \alpha)$. Since F also slides by one bit when a zero is appended to the message, the entire MAC has this sliding property, as noted in [7].

New Attack on 128-EIA3 Version 1.4. We extend the previous attack on version 1.4 of 128-EIA3, in the case where the message starts with one or more zeroes. As discussed in section 5.1, if we delete i of these zeroes, (where $i < d$), the result X' of the accumulation process for this modified message is related to the original value of X by the sliding relationship $X' = (\beta_0, \ldots, \beta_{i-1}, X \gg i)$. Since F also slides by i bits when these zeroes are deleted, the entire MAC has this property and a forgery results with probability of 2^{-i}.

128-EIA3 Version 1.5. The modification to the starting position of the 32-bit segment of y used to form F, introduced in version 1.5 of 128-EIA3, breaks the sliding property on F and provides effective resistance to the types of forgery just discussed. Resistance to bit flipping forgeries is provided by initialising R with keystream, and padding the end of the message prevents bit insertion or deletion forgeries at the end of the message.

Grain-128a. The cipher Grain-128a [1], described in section 4.3, initializes both registers R and A with keystream sequence. Initialising register R with keystream prevents forgery attacks due to bit flipping. Initializing register A with keystream prevents forgeries involving insertion or deletion of bits at the start of the message by breaking the sliding property in the accumulation phase. Padding at the end of the message by one bit of value 1 also prevents bit insertion and deletion forgeries at the end of the message.

Sfinks. For Sfinks stream cipher [2], described in section 4.4, padding is performed by prepending $2d$ ones to M_l to initialize both registers R and A with keystream sequence. However, padding by d bits is sufficient to ensure the register R is loaded with keystream bits before M_l is processed. Using this way of initialization prevents forgery attacks due to flipping, insertion or deletion of bits at the start of the message. Sfinks uses confidentiality sequence z started at z_l for 64-bits as a final mask F. Using a sequence related to the length of the input message prevents bit insertion or deletion forgeries at the end of the message.

6 Conclusion

In this paper, we describe a general model for generating MAC tags using stream ciphers by injecting the input message, either plaintext or ciphertext, indirectly. We outline the options available for various phases in this model and examine the MAC generation processes for three stream ciphers described by this model. These are 128-EIA3, Grain-128a and Sfinks.

The security analysis in section 5.1 highlights the importance of initialising register R with keystream bits. This prevents bit flipping forgeries and reduces the scope of the forgeries involving insertion or deletion of bits at the start of the message. Prepadding the message with at least d ones is a feasible but arguably less efficient alternative.

To prevent the remaining bit insertion and deletion forgeries, both of the following practices must also be adopted:

1. The message is padded with a 1 at the start and/or register A is initialised with keystream (from a fixed location in the keystream sequence).
2. The message is padded with a 1 at the end and/or the final mask F comprises a keystream sequence that depends on the length of the message.

Acknowledgments. The authors gratefully acknowledge the insightful comments provided by an anonymous reviewer of a previous version of this paper. Also, thanks to Chuan-Kun Wu for pointing out an inaccuracy in an earlier draft.

References

1. Ågren, M., Hell, M., Johansson, T., Meier, W.: Grain-128a: a new version of grain-128 with optional authentication. International Journal of Wireless and Mobile Computing 5(1), 48–59 (2011)
2. Braeken, A., Lano, J., Mentens, N., Preneel, B., Verbauwhede, I.: SFINKS: A Synchronous Stream Cipher for Restricted Hardware Environments. eSTREAM, ECRYPT Stream Cipher Project, Report 2005/026 (2005), http://www.ecrypt.eu.org/stream
3. ETSI/SAGE: Specification of the 3GPP Confidentiality and Integrity Algorithms 128-EEA3 & 128-EIA3. Document 4: Design and Evaluation Report. Tech. rep., ETSI (August 11, 2010)
4. ETSI/SAGE: Specification of the 3GPP Confidentiality and Integrity Algorithms 128-EEA3 & 128-EIA3. Document 1: 128-EEA3 and 128-EIA3 Specification. Tech. rep., ETSI (January 4, 2011), http://gsmworld.com/documents/EEA3_EIA3_specification_v1_5.pdf
5. ETSI/SAGE: Specification of the 3GPP Confidentiality and Integrity Algorithms 128-EEA3 & 128-EIA3. Document 2: ZUC Specification. Tech. rep., ETSI (January 4, 2011), http://gsmworld.com/documents/EEA3_EIA3_ZUC_v1_5.pdf
6. European Network of Excellence for Cryptology: The eSTREAM Project (2008), http://www.ecrypt.eu.org/stream/index.html
7. Fuhr, T., Gilbert, H., Reinhard, J.-R., Videau, M.: Analysis of the Initial and Modified Versions of the Candidate 3GPP Integrity Algorithm 128-EIA3. In: Miri, A., Vaudenay, S. (eds.) SAC 2011. LNCS, vol. 7118, pp. 230–242. Springer, Heidelberg (2012)
8. Golić, J.D.: Modes of Operation of Stream Ciphers. In: Stinson, D.R., Tavares, S. (eds.) SAC 2000. LNCS, vol. 2012, pp. 233–247. Springer, Heidelberg (2001)
9. Hell, M., Johansson, T., Meier, W.: Grain: a stream cipher for constrained environments. International Journal of Wireless and Mobile Computing 2(1), 86–93 (2007)
10. Lai, X., Rueppel, R., Woollven, J.: A Fast Cryptographic Checksum Algorithm Based on Stream Ciphers. In: Zheng, Y., Seberry, J. (eds.) AUSCRYPT 1992. LNCS, vol. 718, pp. 339–348. Springer, Heidelberg (1993)
11. Nakano, Y., Cid, C., Fukushima, K., Kiyomoto, S.: Analysis of Message Injection in Stream Cipher-Based Hash Functions. In: Lopez, J., Tsudik, G. (eds.) ACNS 2011. LNCS, vol. 6715, pp. 498–513. Springer, Heidelberg (2011)
12. Nakano, Y., Kurihara, J., Kiyomoto, S., Tanaka, T.: On a construction of stream-cipher-based hash functions. In: Proceedings of the 2010 International Conference on Security and Cryptography (SECRYPT), pp. 1–11. IEEE (2010)

Weimar-DM: A Highly Secure Double-Length Compression Function

Ewan Fleischmann, Christian Forler, Stefan Lucks, and Jakob Wenzel

Bauhaus-Universität Weimar, Germany
{ewan.fleischmann,christian.forler,stefan.lucks,
jakob.wenzel}@uni-weimar.de

Abstract. We present Weimar-DM, a double length compression function using two calls to a block cipher with $2n$-bit key and n-bit block size to compress a $3n$-bit string to a $2n$-bit one. For Weimar-DM, we show that for $n = 128$, no adversary asking less than $2^{n-1.77} = 2^{126.23}$ queries can find a collision with probability greater than 1/2. This is the highest collision security bound ever shown for such a compression function. Even more important, our security analysis is much simpler than that for comparable functions as, *e.g.*, Tandem-DM, Abreast-DM or Hirose-DM. We also give a preimage security analysis of Weimar-DM showing a near-optimal bound of $2^{2n-5} = 2^{251}$ queries. Our security bounds are asymptotically optimal.

Keywords: double length compression function, block cipher based, ideal cipher model, collision security, preimage security.

1 Introduction

A cryptographic hash function is a function which maps an input of arbitrary length to an output of fixed length. It should satisfy at least collision-, preimage- and second-preimage resistance and is one of the most important primitives in cryptography [26].

Block Cipher-Based Hash Functions. Since their initial design by Rivest, MD4-family hash functions (*e.g.*, MD4, MD5, RIPEMD, SHA-1, SHA2 [4,29,30,32,33]) have dominated cryptographic practice. But in recent years, a sequence of attacks on these type of functions [8,12,41,42] has led to a generalized sense of concern about the MD4-approach. The most natural place to look for an alternative is in block cipher-based constructions, which in fact predate the MD4-approach [25]. Another reason for the resurgence of interest in block cipher-based hash functions is due to the rise of size restricted devices such as RFID tags or smart cards: A hardware designer has to implement only a block cipher in order to obtain an encryption function as well as a hash function.

But since the output length of most practical encryption functions is far too short for a collision resistant hash function, *e.g.*, 128-bit for AES, one is mainly interested in sound design principles for *double block length* (DL) hash functions

Table 1. Comparison of double length compression function security results evaluated for $n = 128$ and a success probability of 1/2; for CYCLIC-DM $k > 1$, *i.e.*, the cycle length > 2 the value of k' is ≥ 2

compression function	**collision bound**	**preimage bound**
WEIMAR-DM	$2^{126.23}$ (this paper)	$2^{252.5}$ (this paper)
ABREAST-DM	$2^{124.42}$ [11,22]	2^{246} [1]
HIROSE-DM	$2^{124.55}$ [15]	2^{251} [1]
TANDEM-DM	$2^{120.87}$ [24]	2^{246} [1]
CYCLIC-DM (cycle length > 2)	2^{127-k} [11]	$\approx 2^{128}$ [11,22]
CYCLIC-DM (cycle length 2)	$2^{124.55}$ [11]	$\approx 2^{128}$ [11,22]
CUBE-DM	$2^{125.56}$ [11]	$\approx 2^{128}$ [11,22]
ADD/K-DM	$2^{127-k'}$ [11]	$\approx 2^{128}$ [11,22]
Lee/Kwon	$2^{125.0}$ [22]	$\approx 2^{128}$ [11,22]

[2]. A DL hash function uses a block cipher with n-bit output as the building block by which it maps possibly long strings to $2n$-bit ones. Usually, hash functions are built using compression functions only being able to compress a fixed length input into a (smaller) fixed-length output. These compression functions are iterated, *e.g.*, using the Merkle-Damgård [7,27] transform, in order to get a full-fledged hash function. Since these transforms are property preserving, this article focuses only on the compression function.

WEIMAR-DM. We define a new double length double call compression function as follows (cf. Figure 1).

Definition 1. *Let E be a block cipher taking an $2n$-bit key and an n-bit block size. The compression function $H^{\mathrm{WDM}} : \{0,1\}^n \times \{0,1\}^{2n} \rightarrow \{0,1\}^{2n}$ is defined as (cf. Figure 1)*

$$H^{\mathrm{WDM}}(M, U, \widehat{U}) = \left(E_{M\|U}(\widehat{U}) \oplus \widehat{U}, E_{\overline{M\|U}}(\widehat{U}) \oplus \widehat{U}\right),$$

where $\overline{M\|U}$ denotes the bit-by-bit complement of the bit-string $M\|U$.

In this paper we give very tight collision security and preimage security bounds for WEIMAR-DM. Table 1 gives an overview on known double length compression function designs using two calls to a block cipher with $2n$-bit key and n-bit block size inside. The results obtained in this paper for WEIMAR-DM have also been included.

Our Contribution. We present a new and surprisingly simple design of a double length double call compression function (WEIMAR-DM) and give a collision security bound as well as a preimage security bound. It has the best collision

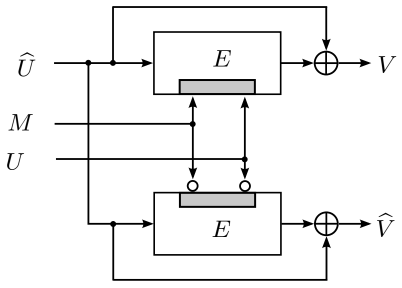


Fig. 1. WEIMAR-DM compression function H^{WDM}; the small circle '$\circ$' denotes a bit-by-bit complement

security bound of all known double length double call compression functions using a block cipher with $2n$-bit key and n-bit block size. Also, no compression function has a tighter preimage security bound, only for HIROSE-DM a comparable one is known. The collision security proof not only delivers an ultra-tight bound, but is also very short.

Outline. The paper is organized as follows: Section 2 gives formal notations and definitions. In Section 3, we prove that any adversary asking less than $2^{126.23}$ oracle queries has negligible advantage in finding a collision for the WEIMAR-DM compression function. Section 4 derives a near-optimal preimage bound for WEIMAR-DM. In Section 5 we discuss our results and conclude. Directly related publications have been mentioned in Table 1, a broader overview on block-cipher based hashing is provided in Appendix A.

2 Preliminaries

2.1 Basic Notions

Ideal Cipher Model. A (k, n) *block cipher* is a keyed family of permutations consisting of two paired algorithms $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ and $E^{-1} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, both accepting a key of size k bits and an input block of size n bits for some $k, n > 0$. For positive k, n, BLOCK(k, n) is the set of all (k, n) block ciphers. For any $E \in$ BLOCK(k, n) and any fixed key $K \in \{0,1\}^k$, decryption $E_K^{-1} := E^{-1}(K, \cdot)$ is the inverse function of encryption $E_K := E(K, \cdot)$, so that $E_K^{-1}(E_K(X)) = X$ holds for any admissible input $X \in \{0,1\}^n$.

Most of the attacks on hash functions based on block ciphers do not utilize the internal structure of the block ciphers. The security of such hash functions is usually analyzed in the *ideal cipher model* [2,9,18]. In this model, the underlying primitive, the block cipher E, is modeled as a family of random permutations $\{E_K\}$ whereas the random permutations are chosen independently for each key K, *i.e.* formally E is selected randomly from $\text{BC}(\mathcal{X}, \mathcal{K})$.

Block Cipher Based Compression Functions. Generally speaking, a *single length* (SL) block cipher based compression function is a compression function $H^{\mathrm{SL}} : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$ using a block cipher with n-bit block size inside. The idea was first discussed in literature by Rabin [25]. Most SL functions use a block cipher from $\textsc{Block}(n, n)$ and compress a $2n$ bit string to an n bit string. A popular example is the Davies-Meyer (DM) [43] mode

$$H(M, U) = E_M(U) \oplus U,$$

which is essentially used twice inside WEIMAR-DM. The $\oplus$ operation is usually called *feed-forward*. A double (block) length (DL) compression function is a compression function $H^{\mathrm{DL}} : \{0,1\}^{k-n} \times \{0,1\}^{2n} \rightarrow \{0,1\}^{2n}$ taking a $(k-n)$-bit message and a $2n$-bit chaining value and outputs a new $2n$-bit chaining value. It also uses a block cipher from $\textsc{Block}(k, n)$ inside. WEIMAR-DM as given in Definition 1 is an example of a double length compression function using exactly two calls to a block cipher from $\textsc{Block}(2n, n)$ in order to compute its output value.

2.2 Security Notions for Double Length Compression Functions

Security is quantified by the success probability of an optimal resource-bounded adversary. An adversary is a computationally unbounded but always-halting collision-finding algorithm $\mathcal{A}$ with resource-bounded access to an oracle $E \in \textsc{Block}(2n, n)$. We can assume (by standard arguments) that $\mathcal{A}$ is deterministic. The adversary may make *forward* queries $(X, K, ?)_{fwd}$ to discover the corresponding value $Y = E_K(X)$, or the adversary may make *backward* queries $(?, K, Y)_{bwd}$, so as to learn the corresponding value $X = E_K^{-1}(Y)$ for which $E_K(X) = Y$. Either way the result of the query is stored in a triple (X_i, K_i, Y_i). The *query history*, denoted by $\mathcal{Q}$, is the tuple $(Q_1, \ldots, Q_q)$ where $Q_i = (X_i, K_i, Y_i)$ is the result of the i-th query made by the adversary and where q is the total number of queries made by the adversary. Without loss of generality, it is assumed that $\mathcal{A}$ asks at most only once on a triplet of a key K_i, a plaintext X_i and a ciphertext Y_i obtained by a query and the corresponding reply.

As usual, we define the collision security of a hash function $\mathcal{H}$ by an experiment of an adversary $\mathcal{A}$ with a security parameter of $2n$, *i.e.* equal to the output bit-length of the compression function.

Experiment 1 (Collision-Finding Experiment $\texttt{Exp-Coll}_{\mathcal{A},H^{\mathrm{DL}}}(2n)$)

1. *The adversary $\mathcal{A}$ is given oracle access to a block cipher $E \in \textsc{Block}(k, n)$ and returns values $(M, U, \widehat{U}), (M', U', \widehat{U}') \in \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n$.*
2. *The output of the experiment is defined to be 1 iff $(M, U, \widehat{U}) \neq (M', U', \widehat{U}')$ and $H^{\mathrm{DL}}(M, U, \widehat{U}) = H^{\mathrm{DL}}(M', U', \widehat{U}')$. In such a case we say that $\mathcal{A}$ has found a collision for H^{DL}.*

The advantage of an adversary $\mathcal{A}$ finding such a collision of H^{DL} is given in the following definition.

Definition 2. $\mathbf{Adv}_{H^{\mathrm{DL}}}^{\mathrm{COLL}}(\mathcal{A}) = \Pr\left[\mathtt{Exp\text{-}Coll}_{\mathcal{A},H^{\mathrm{DL}}}(2n) = 1\right]$.

Since we only limit the adversary by the number of queries it is allowed to ask to the E oracle, *i.e.* it is explicitly given 'unlimited computing power', we write

$$\mathbf{Adv}_{H^{\mathrm{DL}}}^{\mathrm{COLL}}(q) := \max_{\mathcal{A}}\{\mathbf{Adv}_{H^{\mathrm{DL}}}^{\mathrm{COLL}}(\mathcal{A})\},$$

where the maximum is taken over all adversaries that ask at most q oracle queries in total.

There are several notions known that formalize *preimage security* [34]. We adopt *everywhere preimage resistance* (EPRE) in the information theoretic setting which essentially lets the adversary pre-commit to the hash value it likes to be challenged on *before* submitting any queries to the oracle. The corresponding preimage finding experiment is definied as follows.

Experiment 2 (Preimage-Finding Experiment $\mathtt{Exp\text{-}Epre}_{\mathcal{A},H^{\mathrm{DL}}}(2n)$)

1. *The adversary $\mathcal{A}$ is given oracle access to a block cipher $E \in \mathrm{Block}(k,n)$. $\mathcal{A}$ selects and announces a value $(V,\widehat{V}) \in \{0,1\}^n \times \{0,1\}^n$ before making any oracle queries. It outputs a value $(M,U,\widehat{U}) \in \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n$.*
2. *The output of the experiment is defined to be 1 iff $H^{\mathrm{DL}}(M,U,\widehat{U}) = (V,\widehat{V})$. In such a case we say that $\mathcal{A}$ has found a preimage of H^{DL}.*

Now, we let $\mathbf{Adv}_{H^{\mathrm{DL}}}^{\mathrm{EPRE}}(\mathcal{A})$ be the predicate that is true iff '1' is returned by the experiment $\mathtt{Exp\text{-}Epre}_{\mathcal{A},H^{\mathrm{DL}}}(2n)$. The pre-committed value $(V,\widehat{V})$ is an omitted parameter of $\mathbf{Adv}_{H^{\mathrm{DL}}}^{\mathrm{EPRE}}(\mathcal{A})$. Again, we define

$$\mathbf{Adv}_{H^{\mathrm{DL}}}^{\mathrm{EPRE}}(q) := \max_{\mathcal{A}}\{\mathbf{Adv}_{H^{\mathrm{DL}}}^{\mathrm{EPRE}}(\mathcal{A})\},$$

where the maximum is taken over all adversaries that ask at most q oracle queries in total.

3 Collision Security Analysis of Weimar-DM

3.1 Security Results

It is easy to see that H^{WDM} is of type Cyclic-DM with a cycle length of 2, *i.e.*, we directly have a collision security bound of $2^{124.55}$ (cf. Table 1). So we are done with our analysis. But we do not use this generic proof technique but rather use a specialized one delivering us a number of benefits. First, our proof is way simpler than the generic proof for Cyclic-DM. And, second, our new collision security bound is much better by virtually halving the gap between the theoretically optimal bound known before (via Cyclic-DM) and the best bound theoretically possible ($\approx 2^{127}$). Our main collision security result is stated in the following theorem.

Theorem 1. *Let $N = 2^n$. Then,* $\mathbf{Adv}^{\mathrm{COLL}}_{H^{\mathrm{WDM}}}(q) \leq \frac{q(q+1)}{(N-2q)^2}$.

In numerical terms, *e.g.*, for $n = 128$ and $\mathbf{Adv}^{\mathrm{COLL}}_{H^{\mathrm{WDM}}}(q) = 1/2$, we have $q = 2^{126.23}$. Using simple calculus, it is easy to see that for $\alpha = N(1 - \frac{1}{\sqrt{2}}) = 2^{n-1.77}$ we have

$$\mathbf{Adv}^{\mathrm{COLL}}_{H^{\mathrm{WDM}}}(\alpha) = \frac{1}{2} + o(1),$$

where the term $o(1) \to 0$ for $n \to \infty$. Neglecting constant factors, our security bound reads as an asymptotically optimal bound of $O(2^n)$ for a compression function with $2n$-bit output.

3.2 Proof of Theorem 1

We assume that the adversary has made any relevant query to E to come up with a collision – which is reasonable in the ideal cipher model. Another standard assumption made in ideal cipher proofs is that "the adversary never makes a query to which it already knows the answer". By this it is meant, for example, that one can assume that the adversary never makes a query $E_K(X)$, obtaining an answer Y, and then makes the query $E_K^{-1}(Y)$ (which will necessarily be answered by X). We start by considering an arbitrary q-query collision finding adversary $\mathcal{A}$. We then construct an adversary $\mathcal{A}'$ which simulates $\mathcal{A}$ but does sometimes ask an additional query to the E oracle under certain circumstances.

Since $\mathcal{A}'$ is more powerful than $\mathcal{A}$, it suffices to upper bound the success probability of A'. We now give a detailed description of $\mathcal{A}'$ by simultaneously upper bounding its chances of success. We say that an adversary is *successful* if its query history $\mathcal{Q}$ contains the means of computing a collision. This is discussed more thoroughly in the following case analysis.

Description of A'. The adversary $\mathcal{A}'$ maintains an initially empty list $\mathcal{L}$ representing any possible input/output of the compression function H^{WDM} that can be computed by the adversary $\mathcal{A}$. An entry $L \in \mathcal{L}$ is a 4-tuple $(K, X, Y, Y') \in \{0,1\}^{5n}$ where $K \in \{0,1\}^{2n}$, $X \in \{0,1\}^n$ is the $3n$-bit input to the compression function such that $(M, U) = K$ and $\widehat{U} = X$. The n-bit values Y, Y' are given by $Y = E_K(X)$, $Y' = E_{\overline{K}}(X)$.

The list is now built as follows. Say that the adversary $\mathcal{A}$ mounts its i-th query to E or E^{-1}, $1 \leq i \leq q$. In the case of a forward query, the adversary gets hold of the tuple (K, X, Y) where $Y = E_K(X)$. In the case of a backward query, the adversary gets also hold of the tuple (K, X, Y), but in this case $X = E_K^{-1}(Y)$. In either case, the value $X \oplus Y$ is randomly determined by the output of the query.

Now, $\mathcal{A}'$ checks if an entry $L = (K, X, *, *)$ or $L' = (\overline{K}, X, *, *)$ is contained in $\mathcal{L}$ where $'*'$ denotes an arbitrary value. We now analyze the two possible cases $\mathcal{A}'$ might be confronted with and upper bound their success probabilities separately.

Case 1. Neither L nor L' are in $\mathcal{L}$. Then $\mathcal{A}'$ mounts an additional forward query $Y' = E_{\overline{K}}(X)$. Note that $Y' \oplus X$, the result of the 'bottom row' of the compression function, is always uniformly distributed since $K \neq \overline{K}$ always, *i.e.*, the results of the first query asked by the adversary $\mathcal{A}$ and the second query asked additionally by the adversary $\mathcal{A}'$ are always independently distributed. Set $L_i := (K, X, Y, Y')$. We append L_i to the list $\mathcal{L}$.

We now define what we mean by a collision in the list. Fix two integers r, s with $r \neq s$, such that $L_r = (K_r, X_r, Y_r, Y'_r)$ represents the r-th entry in $\mathcal{L}$ and $L_s = (K_s, X_s, Y_s, Y'_s)$ the s-th entry in $\mathcal{L}$ and both entries exist. We say that L_s and L_r collide if a collision of the compression functions occurs that can be computed using the query results given in L_r and L_s. This is the case if at least one of the following two conditions is met:

1. $Y_r \oplus X_r = Y_s \oplus X_s$ and $Y'_r \oplus X_r = Y'_s \oplus X_s$ or
2. $Y_r \oplus X_r = Y'_s \oplus X_s$ and $Y'_r \oplus X_r = Y_s \oplus X_s$.

So for the i-th query, there are at most $i - 1$ entries in the list $\mathcal{L}$ that might collide with L_i. We can upper bound the probability of success of the i-th query by

$$\sum_{j=1}^{i-1} \frac{2}{(N-2q)(N-2q)} \leq \frac{2i}{(N-2q)(N-2q)}$$

As the adversary can ask at most q queries, the list $\mathcal{L}$ cannot contain more than q entries since for any adversary query at most one additional entry is added to the list $\mathcal{L}$ of $\mathcal{A}'$. So the total chance of success for q queries is

$$\leq \sum_{i=1}^{q} \frac{2i}{(N-2q)(N-2q)} = \frac{q(q+1)}{(N-2q)^2}.$$

In case of a collision in $\mathcal{L}$ we give the attack to the adversary.

Case 2. It is clear that, by design, it cannot happen that exactly one of the values L or L' is already in $\mathcal{L}$. So now assume that both values L, L' are already in $\mathcal{L}$. Then $\mathcal{A}'$ ignores this query, since we know that $\mathcal{A}$ has zero chance of winning since otherwise we would have given the attack to the adversary before. □

4 Preimage Security Analysis of Weimar-DM

4.1 Security Results

Preimage security results for double length compression function have 'historically' been limited by the birthday bound, mainly due to technical reasons. At Asiacrypt 2011 a new breakthrough result by Armknecht *et al.* [1] gave new techniques that enable preimage security results for double length compression function way beyond the birthday-barrier. For our preimage security proof of Weimar-DM, we adopt these methods. More precise, we show the following Theorem.

Theorem 2. *Let $N = 2^n$. Then,* $\mathbf{Adv}^{\mathrm{EPRE}}_{H^{\mathrm{WDM}}}(q) \leq 16q/N^2$.

It is easy to see that $\mathbf{Adv}^{\mathrm{EPRE}}_{H^{\mathrm{WDM}}}(2^{2n-5}) = 1/2$ and therefore our bound is asymptotically optimal for a $2n$-bit compression function.

4.2 Proof of Theorem 2

Parts of the proof closely follow the proofs of [1, Theorems 1 and 2]. Our security proof uses the notion of *free* queries. Formally, these can be modeled as queries which the adversary is *forced* to query (under certain conditions), but for which the adversary is not charged: they do not count towards the maximum of q queries which the adversary is allowed. However, these queries become part of the adversary's query history, just like other queries. In particular, the adversary is not allowed, later, to remake these queries "on its own" (due to the previously discussed assumption that the adversary never makes a query which it already owns).

Similar to our collision security analysis, we say the attacker *succeeds* or *finds a preimage* if its query history $\mathcal{Q}$ contains the means of computing a preimage of C, in the sense that there exist values $B \in \{0,1\}^{3n}$, $K_1, K_2 \in \{0,1\}^{2n}$ and $X_1, X_2, Y_1, Y_2 \in \{0,1\}^n$ such that both (X_1, K_1, Y_1) and (X_2, K_2, Y_2) are in the query history $\mathcal{Q}$, $H^{\mathrm{WDM}}(B) = C$ and the two queries used to evaluate $H^{\mathrm{WDM}}(B)$ are precisely $E_{K_1}(X_1)$ and $E_{K_2}(X_2)$. In this case, we also say $\mathcal{Q}$ *contains a preimage* of C. In the current context, where we consider adversaries making 2^n queries or more, the assumption that the adversary never makes a query where it knows the answer to, should be more precisely restated as "the adversary never makes a query that will result in a triple (X, K, Y) which is already present in the query history". (This latter assumption can be made without loss of generality using the fact that $E_K(\cdot)$ is a permutation.) Indeed, if an adversary has made $2^n - 1$ queries under a key K, the result of the last query under that key is predetermined, and thus the adversary "already knows" the answer to this query. However, one should not forbid the adversary from making this query, since the query may be necessary to complete the attack.

Let $(V, \widehat{V}) \in \{0,1\}^n \times \{0,1\}^n$ be the point to invert (chosen by the adversary before it makes any queries to E). We upper bound the probability that, in q queries, the adversary finds a point $(M, U, \widehat{U}) \in (\{0,1\}^n)^3$ such that $H^{\mathrm{WDM}}(M, U, \widehat{U}) = (V, \widehat{V})$.

When the adversary makes a (normal) *forward query* $E_{M\|U}(\widehat{U})$ we give it for free, also, the answer to the query $E_{\overline{M\|U}}(\widehat{U})$. Moreover when the adversary makes a (normal) *backward query* $E^{-1}_{M\|U}(R)$, resulting in an answer $\widehat{U} = E^{-1}_{M\|U}(R)$, we give it for free the answer to the forward query $E_{\overline{M\|U}}(\widehat{U})$. As discussed, we assume that the adversary never makes a query to which it knows the answer. Thus the elements of the adversary's query history $\mathcal{Q}$ can be paired into adjacent pairs of the form $(M\|U, \widehat{U}, R), (\overline{M\|U}, \widehat{U}, S)$. We call such a pair an *adjacent query pair.*

We now give further free queries to the adversary, in the fashion described next. After each adjacent query pair has been completed (namely, after the adversary has received the response to both its query and its associated free query, and after these have been placed in the query history), we check whether the key prefix used for the latest query is such that the (current) query history contains exactly $N/2$ adjacent query pairs with this key prefix. If so, we give *all* remaining adjacent query pairs under this key for free to the adversary. There will be exactly $N/2$ such query pairs. We insert these $N/2$ free query pairs into the query history pair-by-pair (to maintain, mostly for conceptual simplicity, the adjacent pair structure of the query history). We note that, after these free queries have been inserted into the query history, the adversary cannot make any more queries under this key prefix, since, the adversary is assumed never to make a query to which it knows the answer. When $N/2$ free query pairs are given to the adversary in the fashion just described, we say that a *super query* occurs. This can be summed up as follows.

Super Query. Given $N/2$ adjacent query pairs to E all using the same key $K \in \{0,1\}^{2n}$, all the remaining $N/2$ queries using the same key K and the remaining $N/2$ queries using key $\overline{K}$ are given for free.

We say that an adjacent query pair $(M\|U, \widehat{U}, R), (\overline{M\|U}, \widehat{U}, S)$ is *successful*, if $\widehat{U} \oplus R = V$ and $\widehat{U} \oplus S = \widehat{V}$, or if $\widehat{U} \oplus R = \widehat{V}$ and $\widehat{U} \oplus S = V$. Thus the adversary obtains a preimage of $(V, \widehat{V})$ precisely if it obtains a successful adjacent query pair. This can occur in one of two ways: either the winning query pair is part of a super query, or not. We let $\mathsf{SuperQueryWin}(\mathcal{Q})$ denote the event that the adversary obtains a winning query pair that is part of a super query, and $\mathsf{NormalQueryWin}(\mathcal{Q})$ the event that the adversary obtains a winning query pair of normal queries (either forward or backward). It thus suffices to upper bound

$$\Pr[\mathsf{SuperQueryWin}(\mathcal{Q})] + \Pr[\mathsf{NormalQueryWin}(\mathcal{Q})].$$

Here probabilities are taken (as usual) over the adversary's randomness (if any) and over the randomness of the ideal cipher.

We first upper bound $\Pr[\mathsf{NormalQueryWin}(\mathcal{Q})]$. Note that when the adversary makes, say, a forward query $E_{M\|U}(\widehat{U})$, at most $N/2-2$ queries (counting free queries) have been previously answered with the key $M\|U$, since otherwise a super query for the key $M\|U$ would have occurred. Thus the value $R = E_{M\|U}(\widehat{U})$ comes uniformly at random from a set of size at least $N/2+2 \geq N/2$, and there is chance at most $2/(N/2) = 4/N$ that either $\widehat{U} \oplus R = V$ or $\widehat{U} \oplus R = \widehat{V}$ (this is also true if $V = \widehat{V}$). If, say, $\widehat{U} \oplus R = V$, there is further chance at most $1/(N/2) = 2/N$ that the free query $E_{\overline{M\|U}}(\widehat{U})$ returns $\widehat{U} \oplus \widehat{V}$, since the answer to the free query comes uniformly at random from a set of size at least $N/2+1 \leq N/2$. Other cases (*e.g.* when $\widehat{U} \oplus R = \widehat{V}$, and when the adversary makes a backward query $E^{-1}_{M\|U}(R)$) are similarly analyzed, showing that the adversary's chance of triggering the event $\mathsf{NormalQueryWin}(\mathcal{Q})$ at any given query is at most $(4/N)(2/N) = 8/N^2$. Since the adversary makes q queries total, we therefore have

$$\Pr[\mathsf{NormalQueryWin}(\mathcal{Q})] \leq 8q/N^2. \tag{1}$$

We now bound $\Pr[\mathsf{SuperQueryWin}(\mathcal{Q})]$. Assume that a super query is about to occur on keys $M\|U$ and $\overline{M\|U}$ meaning that the value of $E_{M\|U}(\cdot)$ and $E_{\overline{M\|U}}(\cdot)$ are already known on exactly $N/2$ points. Let us denote this set of points by $\mathcal{X}$ and let $\mathcal{Y} = E_{M\|U}(\mathcal{X})$ and $\mathcal{Y}' = E_{\overline{M\|U}}(\mathcal{X})$. Further let $\mathcal{R} = \{0,1\}^n \backslash \mathcal{X}$, $\mathcal{S} = \{0,1\}^n \backslash \mathcal{Y}$ and $\mathcal{S}' = \{0,1\}^n \backslash \mathcal{Y}'$. Clearly, $|\mathcal{X}| = |\mathcal{Y}| = |\mathcal{Y}'| = |\mathcal{R}| = |\mathcal{S}| = |\mathcal{S}'|$.

Now fix a point $R \in \mathcal{R}$ in the domain of the super query. We now estimate the probability that this point R induces a successful pair. This can only be the case if

1. $R \oplus V \in \mathcal{S}$ and $R \oplus \widehat{V} \in \mathcal{S}'$ or
2. $R \oplus \widehat{V} \in \mathcal{S}$ and $R \oplus \mathcal{V} \in \mathcal{S}'$.

The probability that $E_{M\|U}(R) = R \oplus V$ and $E_{M\|U}(R) = R \oplus \widehat{V}$ equals $1/(N/2)^2$. The same is true for the probability that $E_{M\|U}(R) = R \oplus \widehat{V}$ and $E_{M\|U}(R) = R \oplus V$. Thus the total probability to be successful in a super query is at most

$$2 \cdot N/2 \cdot \left(\frac{1}{N/2}\right)^2 = \frac{2}{N/2}.$$

Since at most $q/(N/2)$ super queries can ever occur, we have

$$\Pr[\mathsf{SuperQueryWin}(\mathcal{Q})] \leq 8q/N^2. \tag{2}$$

The sum of (1) and (2) gives our claim. □

5 Discussion and Conclusion

In this paper, we have presented WEIMAR-DM, a double length compression function. We have shown very tight collision security bounds and preimage security bounds. The collision security bound is currently the best known bound for any such compression functions known in literature. Also, no compression function with a tighter preimage security bound is known – only HIROSE-DM has a numerically similar bound. For our security benefits, we have to pay the price of two key-scheduler runs per compression function.

Although a lot of progress has been made in recent years in the field of double length hashing, a lot of open questions remain. Related to our analysis, it would be interesting to investigate if our techniques in the collision security proof can be generalized, *e.g.*, to a subclass of CYCLIC-DM. Another open problem is the design of conveniently secure compression functions only using a block cipher from BLOCK(n, n).

References

1. Armknecht, F., Fleischmann, E., Krause, M., Lee, J., Stam, M., Steinberger, J.: The Preimage Security of Double-Block-Length Compression Functions. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 233–251. Springer, Heidelberg (2011)
2. Black, J.A., Rogaway, P., Shrimpton, T.: Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 320–335. Springer, Heidelberg (2002)
3. Bos, J.W., Özen, O., Stam, M.: Efficient Hashing Using the AES Instruction Set. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 507–522. Springer, Heidelberg (2011)
4. Bosselaers, A., Preneel, B. (eds.): RIPE 1992. LNCS, vol. 1007. Springer, Heidelberg (1995)
5. Brachtl, B., Coppersmith, D., Hyden, M.M., Meyer, C.H., Matyas, S.M., Oseas, J., Pilpel, S., Schilling, M.: Data authentication using modification detection codes based on a public one way encryption function. U.S. Patent No. 4,908,861, March 13 (1990)
6. Brassard, G. (ed.): CRYPTO 1989. LNCS, vol. 435. Springer, Heidelberg (1990)
7. Damgård, I.: A design principle for hash functions. In: Brassard [6], pp. 416–427
8. den Boer, B., Bosselaers, A.: Collisions for the Compression Function of MD-5. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 293–304. Springer, Heidelberg (1994)
9. Even, S., Mansour, Y.: A Construction of a Cipher From a Single Pseudorandom Permutation. In: Matsumoto, T., Imai, H., Rivest, R.L. (eds.) ASIACRYPT 1991. LNCS, vol. 739, pp. 210–224. Springer, Heidelberg (1993)
10. Fleischmann, E., Gorski, M., Lucks, S.: On the Security of Tandem-DM. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 84–103. Springer, Heidelberg (2009)
11. Fleischmann, E., Gorski, M., Lucks, S.: Security of Cyclic Double Block Length Hash Functions. In: Parker, M.G. (ed.) Cryptography and Coding 2009. LNCS, vol. 5921, pp. 153–175. Springer, Heidelberg (2009)
12. Dobbertin, H.: The status of MD5 after a recent attack (1996)
13. Hattori, M., Hirose, S., Yoshida, S.: Analysis of Double Block Length Hash Functions. In: Paterson, K.G. (ed.) Cryptography and Coding 2003. LNCS, vol. 2898, pp. 290–302. Springer, Heidelberg (2003)
14. Hirose, S.: Provably Secure Double-Block-Length Hash Functions in a Black-Box Model. In: Park, C., Chee, S. (eds.) ICISC 2004. LNCS, vol. 3506, pp. 330–342. Springer, Heidelberg (2005)
15. Hirose, S.: Some Plausible Constructions of Double-Block-Length Hash Functions. In: Robshaw, M. (ed.) FSE 2006. LNCS, vol. 4047, pp. 210–225. Springer, Heidelberg (2006)
16. Hohl, W., Lai, X., Meier, T., Waldvogel, C.: Security of Iterated Hash Functions Based on Block Ciphers. In: Stinson [40], pp. 379–390
17. ISO/IEC. ISO DIS 10118-2: Information technology - Security techniques - Hash-functions, Part 2: Hash-functions using an n-bit block cipher algorithm. First released in 1992 (2000)
18. Kilian, J., Rogaway, P.: How to Protect DES against Exhaustive Key Search. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 252–267. Springer, Heidelberg (1996)

19. Knudsen, L.R., Lai, X., Preneel, B.: Attacks on Fast Double Block Length Hash Functions. J. Cryptology 11(1), 59–72 (1998)
20. Knudsen, L.R., Muller, F.: Some Attacks Against a Double Length Hash Proposal. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 462–473. Springer, Heidelberg (2005)
21. Lee, J., Kwon, D.: The security of abreast-dm in the ideal cipher model. Cryptology ePrint Archive, Report 2009/225 (2009), http://eprint.iacr.org/
22. Lee, J., Kwon, D.: The Security of Abreast-DM in the Ideal Cipher Model. IACR Cryptology ePrint Archive, 2009, 225 (2009)
23. Lee, J., Stam, M.: MJH: A Faster Alternative to MDC-2. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 213–236. Springer, Heidelberg (2011)
24. Lee, J., Stam, M., Steinberger, J.: The Collision Security of Tandem-DM in the Ideal Cipher Model. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 561–577. Springer, Heidelberg (2011)
25. Rabin, M.: Digitalized Signatures (1978)
26. Menezes, A., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press (1996)
27. Merkle, R.C.: One Way Hash Functions and DES. In: Brassard [6], pp. 428–446
28. Nandi, M., Lee, W.I., Sakurai, K., Lee, S.: Security Analysis of a 2/3-Rate Double Length Compression Function in the Black-Box Model. In: Gilbert, H., Handschuh, H. (eds.) FSE 2005. LNCS, vol. 3557, pp. 243–254. Springer, Heidelberg (2005)
29. NIST National Institute of Standards and Technology. FIPS 180-1: Secure Hash Standard (April 1995), http://csrc.nist.gov
30. NIST National Institute of Standards and Technology. FIPS 180-2: Secure Hash Standard (April 1995), http://csrc.nist.gov
31. Preneel, B., Govaerts, R., Vandewalle, J.: Hash Functions Based on Block Ciphers: A Synthetic Approach. In: Stinson [40], pp. 368–378
32. Rivest, R.L.: RFC 1321: The MD5 Message-Digest Algorithm. Internet Activities Board (April 1992)
33. Rivest, R.L.: The MD4 Message Digest Algorithm. In: Menezes, A., Vanstone, S.A. (eds.) CRYPTO 1990. LNCS, vol. 537, pp. 303–311. Springer, Heidelberg (1991)
34. Rogaway, P., Shrimpton, T.: Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. In: Roy, B., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 371–388. Springer, Heidelberg (2004)
35. Rogaway, P., Steinberger, J.P.: Constructing Cryptographic Hash Functions from Fixed-Key Blockciphers. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 433–450. Springer, Heidelberg (2008)
36. Rogaway, P., Steinberger, J.P.: Security/Efficiency Tradeoffs for Permutation-Based Hashing. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 220–236. Springer, Heidelberg (2008)
37. Satoh, Haga, Kurosawa: Towards secure and fast hash functions. TIEICE: IEICE Transactions on Communications/Electronics/Information and Systems (1999)
38. Stam, M.: Blockcipher-Based Hashing Revisited. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 67–83. Springer, Heidelberg (2009)
39. Steinberger, J.P.: The Collision Intractability of MDC-2 in the Ideal-Cipher Model. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 34–51. Springer, Heidelberg (2007)
40. Stinson, D.R. (ed.): CRYPTO 1993. LNCS, vol. 773. Springer, Heidelberg (1994)

41. Wang, X., Lai, X., Feng, D., Chen, H., Yu, X.: Cryptanalysis of the Hash Functions MD4 and RIPEMD. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 1–18. Springer, Heidelberg (2005)
42. Wang, X., Yin, Y.L., Yu, H.: Finding Collisions in the Full SHA-1. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 17–36. Springer, Heidelberg (2005)
43. Winternitz, R.S.: A Secure One-Way Hash Function Built from DES. In: IEEE Symposium on Security and Privacy, pp. 88–90 (1984)

A Related Work

Schemes with NonOoptimal or Unknown Collision Security. Preneel *et al.* [31] discussed the security of single (block)length hash functions against several generic attacks. They concluded that 12 out of 64 hash functions are secure against these attacks. However, formal proofs were first given by Black *et al.* [2] about 10 years later. Their most important result is that 20 hash functions – including the 12 mentioned above – are optimally collision resistant. Knudsen *et al.* [19] discussed the insecurity of DBL hash functions with rate 1 composed of (n, n) block ciphers. Hohl *et al.* [16] analyzed the security of DBL compression functions with rate 1 and 1/2. Satoh *et al.* [37] and Hattoris *et al.* [13] discussed DBL hash functions with rate 1 composed of $(2n, n)$ block ciphers. MDC-2 and MDC-4 [5,17] are (n, n) block cipher based DBL hash functions with rates 1/2 and 1/4, respectively. Steinberger [39] proved that for MDC-2 instantiated with, *e.g.*, AES-128 no adversary asking less than $2^{74.9}$ can usually find a collision. Nandi *et al.* [28] proposed a construction with rate 2/3 but it is not optimally collision resistant. In [20], Knudsen and Muller presented some attacks against it. At EUROCRYPT'08 and CRYPTO'08, Steinberger [35,36] proved some security bounds for fixed-key (n, n) block cipher based hash functions, *i.e.*, permutation based hash functions, that all have small rates and low security guarantees. None of these schemes/techniques mentioned so far are known to have birthday-type collision resistance. Lee and Stam [23] gave a scheme similar to MDC-2, called MJH. It uses finite field multiplications to offer a collision security bound in the iteration of $O(2^{2n/3-\log n})$.

Schemes with Birthday-Type Collision Security. Merkle [27] presented three DBL hash functions composed of DES with rates of at most 0.276. They are optimally collision resistant in the ideal cipher model. Hirose [14] presented a class of DBL hash functions with rate 1/2 which are composed of two different and independent $(2n, n)$ block ciphers that have birthday-type collision resistance. At FSE'06, Hirose [15] presented a rate 1/2 and $(2n, n)$ block cipher based DBL hash function that has birthday-type collision resistance. He stated that for his compression function, no adversary can find a collision with probability greater than 1/2 if no more than $2^{124.55}$ queries are asked (see [10, App. B] for details on this). For Tandem-DM, the best known collision security bound is $2^{120.87}$ queries [24]. Fleischmann *et al.* [11] as well as Lee and Kwon [21] independently provided a security bound for Abreast-DM of $2^{124.42}$. In [11] a lot of variants are also discussed, *e.g.*, Cyclic-DM, Cube-DM or Add/k-DM. Bos *et al.* [3]

provided practical performance figures for some double length hash functions using the AES-NI instruction set.

Preimage Security Results. For single length compression functions, tight security results are known [2,38]. For double length compression functions, some birthday-type preimage results are also known [22,24], essentially stating that any adversary asking less 2^n queries has only a negligible chance of finding a preimage. For ABREAST-DM, TANDEM-DM and HIROSE-DM there are better bounds known [1] (cf. also Table 1).

An Efficient IND-CCA2 Secure Variant of the Niederreiter Encryption Scheme in the Standard Model*

Preetha Mathew K.[1], Sachin Vasant[2], and Sridhar Venkatesan[2], and C. Pandu Rangan[1]

[1] Theoretical Computer Science Lab,
Department of Computer Science and Engineering,
Indian Institute of Technology Madras, India
{kpreetha,prangan}@cse.iitm.ac.in

[2] Department of Mathematics and Computer Applications,
PSG College of Technology, Coimbatore, India
{sachin.tcs2k7,vsridhar1729}@gmail.com

Abstract. In this paper, we propose an IND-CCA2 secure code based encryption scheme in the standard model, built on the Niederreiter encryption scheme. The security of the scheme is based on the hardness of the *Syndrome Decoding* problem and the *Goppa Code Distinguishability* problem. The system is developed according to the construction similar to IND-CCA2 secure encryption scheme by Peikert and Waters using the lossy trapdoor functions. Compared to the existing IND-CCA2 secure variants due to Dowsley et.al. and Freeman et. al. (using the κ repetition paradigm initiated by Rosen and Segev), our scheme is more efficient as it avoids κ repetitions. This can be considered as the first practical code-based encryption scheme that is IND-CCA2 secure in the standard model.

Keywords: Standard Model, CCA-2 security, Neiderreiter Cryptosystem, Syndrome Decoding, Code Indistinguishability.

1 Introduction

The important notions in public key cryptography are that of security under chosen ciphertext attack (CCA security) and trapdoor functions (TDFs) [7,17,20]. Trapdoor functions, which are hard to invert unless one possesses some secret trapdoor information was conceptualized by Diffie and Hellman [6] and were realized by the RSA implementation by Rivest, Shamir and Adleman [21]. The security notions have been maturing since then and currently indistinguishability against a chosen ciphertext attack by an adversary who has a brief access even to a decryption oracle (IND-CCA2) is considered to be the strongest security notion for encryption schemes.

* This material is based upon work supported by the Summer Fellowship from Indian Statistical Institute Chennai.

Most of the efficient IND-CCA2 secure schemes were proven secure in the random oracle model. But Canetti et. al. [4] observed that instantiating random oracles in the real scenario using hash functions may lead to insecure implementation of the scheme. Hence, construction of schemes that are IND-CCA2 secure in the standard model (without the use of random oracles) is preferred. There are some compromises in the efficiency, due to much stronger security requirements.

In [19], the authors presented a black box construction of IND-CCA2 secure encryption scheme based on lossy TDFs and all-but-one trapdoor functions, with a witness recovering decryption algorithm. The decryption first recovers the randomness that was used to create the ciphertext, and then tests the validity of the ciphertext by re-encrypting the message using retrieved randomness. The same method is adapted by Freeman et. al. [12] and Rosen and Segev [23]. This paper investigates, how code-based assumptions can be used for obtaining an IND-CCA2 secure encryption scheme in the standard model using witness recovery.

Code-based cryptography was initiated by the seminal paper due to McEliece [22]. The security of the McEliece encryption scheme is based on the hardness of the problems of *Bounded Decoding* and *Goppa Code Distinguishability.* Initially, the scheme did not gain sufficient acclaim, due to the large key-sizes, hence, Niederreiter proposed a cryptosystem, that is dual of the McEliece cryptosystem [13]. Unlike number-theoretic schemes that are weak against the attack due to Shor [24], McEliece and Niederreiter cryptosystems have resisted such attacks (when using Goppa codes), thus making them strong candidates for *Post-Quantum Cryptography.* Also, in comparison with number-theoretic encryption schemes, code-based schemes are computationally attractive, as the underlying operations are vector-matrix multiplication and vector additions, allowing even parallel processing in practical contexts.

Related Work. Li et. al. [16] showed the equivalence of the security of McEliece and Niederreiter cryptosystems. Berson [3] showed that the McEliece cryptosystem is not CPA secure. Kobara and Imai [15] gave conversions for the McEliece PKE that were proved CCA secure in the random oracle model. Nojima et. al. [18] presented randomised variants of the McEliece and Niederreiter schemes, that were IND-CPA secure in the standard model.

Recently Dowsley et al. [8] proposed a CCA2 secure scheme in the standard model using the randomized version of the CPA secure McEliece cryptosystem [18], by the method of κ-repetition. Rosen et al. [23] initiated the study of the one-wayness under correlated products (κ-repetition) and Freeman et al. [12] proposed instantiation of lossy trapdoor functions and correlation-secure trapdoor functions. They proposed a correlation-secure trapdoor functions based on the hardness of syndrome decoding to obtain a CCA-2 secure encryption scheme in the standard model.

Motivation. The existing variants of the Niederreiter and McEliece cryptosystem that are IND-CCA2 secure in the standard model [8,12] are all based on the κ repetition paradigm [23]. Such cryptosystems lead to extremely large keys and ciphertexts, and thus incurring a huge encrypting cost. The scheme by Freeman et al. [12] not only requires the hardness of the $[n, k]$*Syndrome Decoding* problem, but also requires the hardness of $[n, \kappa k - (\kappa - 1)n]$ *Syndrome Decoding* problem, where κ is the security parameter. The parameters chosen in [12] are for the deterministic version of Niederreiter, which are vulnerable to an attack proposed by Håstad, [14]. Hence, the parameters that are generally used for the above construction, requires a large (n, k) resulting in large key-sizes. Therefore, a cryptosystem that does not follow the κ-repetition paradigm is desirable.

Our Contributions. Our scheme uses ideas from [19,18,5]. We use strongly unforgeable one-time signature (OTS) to handle malleability related issues as in [19,23,5]. However we use two injective functions on the verification key and the message. This novel approach leads to the elimination of the κ-repetition. Note that the instantiation of the protocol in [19] in a direct way leads to the scheme similar to [8] that involves κ-repetition. Thus, for practical code-based cryptosystems such a $\kappa-$ repetition paradigm needs to be avoided.
Our contributions in this paper are:

(i) Proposal of efficient variant of the Niederreiter scheme, that are not based on the κ repetition paradigm, and
(ii) formal argument of their security against IND-CCA2 adversary in the standard model.

An analogous idea for selective provision of trapdoor was also used by Agrawal et. al. [1] in the lattice-based setup, for simulation of the key-extraction phase in their proof of CPA security of a (H)IBE in the standard model. They use lattices built from two parts called right and left lattices. A trapdoor for the left lattices is used as the master secret in the real system and enables one to generate private keys for all identities. A trapdoor for the right lattice is only used in the proof of selective security and enables the simulator to generate private keys for all identities except for one. They used right lattices to achieve the targetted ID method of proving, where the key extraction simulation extracts private keys for all IDs except the targetted ID. In our case, f_1, f_2 are the two injective functions that achieve the same purpose, but the details and computations are entirely different from [1].

Organization of the Paper. Section 2 lists the preliminaries which include the security notions and the hardness assumptions used in the paper. Section 3 gives the proposed scheme, the proof of security, the secure parameters for the cryptosystems, and the comparison with existing schemes. Concluding remarks are offered in section 4.

2 Preliminaries

2.1 Notation

If x is a string, then $|x|$ denotes its length, while $|S|$ represents the cardinality of the set S. If $\kappa \in \mathbb{N}$ then 1^κ denotes the string of length κ. $s \in_R S$ denotes the operation of choosing an element s from a set S uniformly at random.

$w \leftarrow \mathcal{A}(x, y, ...)$ represents the running of algorithm $\mathcal{A}$ with inputs $x, y, ...$ and producing output w. We write $w \leftarrow \mathcal{A}^{\mathcal{O}}(x, y, ...)$ for representing an algorithm $\mathcal{A}$ having access to oracle $\mathcal{O}$. We denote by $\Pr[E]$ the probability that the event E occurs. For a matrix M, its transpose is represented by M^T and its inverse (if it exists) is represented by M^{-1}. If a and b are two strings of bits, we denote their bitwise XOR by $a \oplus b$. Let A be a $m \times n_1$ matrix and B be a $m \times n_2$ matrix, the $C = [A|B]$ is a $m \times (n_1 + n_2)$ matrix, where each row i of C is the concatenation of the i^{th} row of A with that of B.

Since, the proposed cryptosystems are code-based, a few notations regarding coding theory are introduced. A binary linear-error correcting code of length n and dimension k or a $[n, k]$- code is a *k-dimensional* subspace of $\mathbb{F}_2^n$. The rate of a code can be calculated as $\frac{k}{n}$. A code is high-rate if $\frac{k}{n} \rightarrow 1$. If the minimum hamming distance between any two codewords is d, then the code is a $[n, k, d]$ code. The hamming weight of a codeword x, $\mathsf{wt}(x)$, is the number of non-zero bits in the codeword. For $t \leq \lfloor \frac{d-1}{2} \rfloor$, the code is said to be t-error correcting if it detects and corrects errors of weight at most t. Hence, the code can also be represented as a $[n, k, 2t + 1]$ code. The generator matrix $G \in \mathbb{F}_2^{k \times n}$ of a $[n, k]$ linear code C is a matrix of rank k whose rows span the code C. The parity-check matrix $H \in \mathbb{F}_2^{n-k \times n}$ of a $[n, k]$ code C is a matrix satisfying $HG^T = 0$. Hence, code C can be defined as $\{mG : \forall m \in \mathbb{F}_2^k\}$ or $\{c : Hc^T = 0\}$.

2.2 Definition of the Security Notions

The IND-CCA2 security for any Public-Key Encryption Scheme (PKE) is defined as follows:

Definition 1 IND-CCA2 Security. *For a two -stage adversary $\mathcal{A}$ = ($\mathcal{A}_1$,$\mathcal{A}_2$) against* PKE*, with security parameter κ, we associate the following experiment* $\mathsf{Exp}^{cca2}_{\mathsf{PKE},\mathcal{A}}(\kappa)$*:*

$(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^\kappa)$
$(\mathsf{m}_0, \mathsf{m}_1, \mathsf{state}) \leftarrow \mathcal{A}_1^{\mathsf{Dec}(\mathsf{sk},\cdot)}(\mathsf{pk})$ *s.t.* $|\mathsf{m}_0| = |\mathsf{m}_1|$
$b \in_R (0,1)$
$\mathsf{c}^* \leftarrow \mathsf{Enc}(\mathsf{pk},\mathsf{m}_b)$
$b' \leftarrow \mathcal{A}_2^{\mathsf{Dec}(\mathsf{sk},\cdot)}(\mathsf{c}^*, state)$
if $b=b'$ *return 1 else return 0*

The adversary $\mathcal{A}_2$ is not allowed to query $\mathsf{Dec}(\mathsf{sk},\cdot)$ *with* c^**. We define the advantage of $\mathcal{A}$ in the experiment as*

$$\mathsf{Adv}^{cca2}_{\mathsf{PKE},\mathcal{A}}(\kappa) = |Pr[\mathsf{Exp}^{cca2}_{PKE,\mathcal{A}}(\kappa) = 1] - \tfrac{1}{2}|$$

We say that PKE *is indistinguishable against adaptive chosen-ciphertext attacks (IND-CCA2) if for all probabilistic polynomial time (PPT) adversaries $\mathcal{A}$ = ($\mathcal{A}_1$,$\mathcal{A}_2$) that makes a polynomial number of oracle queries the advantage of $\mathcal{A}$ in the experiment is a negligible function of κ (the security parameter).*

The security notion of One-time strongly unforgeable, or one-time existentially unforgeable under chosen message attack (EUF-1CMA) is as follows (based on [2]):

Definition 2 EUF-1CMA. *A signature scheme is said to secure under EUF-1CMA, if there exists no PPT algorithm $\mathcal{A}$, which has knowledge of only the verification key* vk *and the public parameters and access for just one query to the signature oracle to obtain a tuple(m', σ') , outputs a valid signature $(m, \sigma) \neq (m', \sigma')$ with a non-negligible probability.*

The probability that any PPT adversary $\mathcal{A}$ wins the EUF-1CMA game for a one-time signature $\mathcal{OS}$, given the verification key vk is denoted by $\mathsf{Succ}_{\mathcal{A}}^{\mathcal{OS}}(\mathsf{vk})$.

2.3 Security Assumptions

The following are some of the hard problems on which the security of the proposed cryptosystems is based.

Definition 3 Syndrome Decoding Problem. *For some parameters $[n, k, 2t+1]$ given an $a \in \mathbb{F}_2^{n-k}$ and a matrix $H \in \mathbb{F}_2^{n-k \times n}$, find a vector $e \in \mathbb{F}_2^n$ with weight $\mathsf{wt}(e) \leq t$ such that $He^T = a$.*

The advantage of a PPT algortihm $\mathcal{D}$ of solving the problem is denoted by $\mathsf{Adv}_{\mathcal{D}}^{\mathsf{SD}}(n, k)$.

Assumption 1. *For any probabilistic polynomial time algorithm $\mathcal{F}$, $\mathsf{Adv}_{\mathcal{F}}^{\mathsf{SD}}(C) < \epsilon_1(n, k)$ where $\epsilon_1(n, k)$ is a negligible value with respect to n and k.*

For Goppa codes, there is a polynomial time bounded decoding/syndrome decoding algorithm. Thus, there is a preference for most code-based cryptosystems to use the Goppa code as a trapdoor.

Definition 4 Goppa code-distinguishability. *For parameters $[n, k, 2t + 1]$ given a matrix $H \in \mathbb{F}_2^{n-k \times n}$, output 1 if H is a parity check matrix of a Goppa code, 0 if H is not a parity check matrix of any Goppa code.*

The advantage of a PPT algorithm $\mathcal{D}$ of solving the problem is denoted by $\mathsf{Adv}_{\mathcal{D}}^{\mathsf{CD}}(n, k)$.

Assumption 2. *For any probabilistic polynomial time distinguisher $\mathcal{D}$, $\mathsf{Adv}_{\mathcal{D}}^{\mathsf{CD}}(n, k) < \epsilon_2(n, k)$ where $\epsilon_2(n, k)$ is a negligible function if it is not a high rate goppa code, [9].*

$$|Pr[\mathcal{D}(H) = 1] - Pr[\mathcal{D}(M) = 1]| < \epsilon_2(n, k)$$

where H is the parity check matrix of the Goppa code and $M \in_R \mathbb{F}_2^{n-k \times n}$.

3 Variant of the Niederreiter Cryptosystem

3.1 Proposed Scheme

The proposed system uses the randomized Niederreiter encryption approach given in [18] along with the construction by Peikert and Waters [19] (which uses lossy trapdoors and all-but-one trapdoor functions) to obtain a CCA-2 secure encryption scheme in the standard model.

The use of a one-time signature for non-malleability of the ciphertext is a paradigm initiated by Dolev et al. [7]. Thus our scheme uses the following:

– Any one-time strongly unforgeable signature scheme $\mathcal{OS}(\mathsf{KeyGen}_{\mathcal{OS}}, \mathsf{Sign}_{\mathcal{OS}}, \mathsf{Verfiy}_{\mathcal{OS}})$, for the security parameter κ such that
  - $\mathsf{KeyGen}_{\mathcal{OS}}(\kappa)$ outputs the key-pair $(\mathsf{vk}, \mathsf{sk})$, where vk is the verifcation key, and sk is signing key. The size of the domain of the keys $|\mathsf{vk}|, |\mathsf{sk}| = \kappa$
  - $\mathsf{Sign}_{\mathcal{OS}}(\mathsf{sk}, M)$ outputs a signature σ on a message M using the signing key sk. In this case, the message M is of size $2(n-k)$.
  - $\mathsf{Verfiy}_{\mathcal{OS}}(\mathsf{vk}, M, \sigma)$, verifies the signature σ on message M using the verification key vk. If the signature is valid then the verification algorithm outputs VALID, else it outputs INVALID.

– The injective functions f_1, f_2, map the verification key, to the set of all full-rank $n-k \times n$ binary matrices and the set of all $n \times n$ permutation matrices, respectively. With the use of such functions, we are in possession of a tool that makes the ciphertext dependent on the verification key also, but without compromise in security. Such a setup, does away with most malleability issues.

– The encryption scheme uses two public keys $\widetilde{H}_1$, $\widetilde{H}_2$. The decoding trapdoor associated with $\widetilde{H}_1$ is used in the decryption phase to retreive the encrypted message. The trapdoor with regards to $\widetilde{H}_2$ is used in the decryption oracle, and plays a role analogous to all-but-one trapdoor in [19], which makes use of the property $HG^T = 0$ (where H is a parity check matrix and G is among the corresponding generator matrices).

A formal description of the scheme is as follows:

System Parameters. The system paramters are as follows: Let D_{vk} denotes the domain of the signature and verification keys.

– Parameters of the code n, k, t for any $[n, k, 2t+1]$ linear code, with n, k determined by the security parameter κ, and $t = \frac{n-k}{\log_2 n}$. Also, select n_1, n_2 such that $n = n_1 + n_2$ and $n_1 = bn$, where b is a positive rational number and $b < 1$. The message to be encrypted is of length n_2, and weight $t_2 = \lfloor \frac{n_2 t}{n_1 + n_2} \rfloor$. We make use of ephmeral keys (randomness) r of length n_1 with weight $t_1 = \lceil \frac{n_1 t}{n_1 + n_2} \rceil$.

– A one-time strongly unforgeable signature scheme $\mathcal{OS}(\mathsf{KeyGen}_{\mathcal{OS}}, \mathsf{Sign}_{\mathcal{OS}}, \mathsf{Verfiy}_{\mathcal{OS}})$, for the security parameter κ.

– An injective function $f_1 : D_{\mathsf{vk}} \rightarrow \mathbb{F}_2^{n-k \times n}$ which takes verification key as input, and gives the random matrix (full-rank) H_{vk} .
– An injective function $f_2 : D_{\mathsf{vk}} \rightarrow \mathcal{P}_{n \times n}$ which takes the verification key as input and output a $n \times n$ permutation matrix P_{vk}, for the Niederreiter cryptosystem.

Key Generation. For the security parameter 1^κ, the KeyGen is as follows:

– Randomly select two distinct $[n, k, 2t + 1]$Goppa codes with parity check matrices H_1, H_2 respectively
– Randomly select an invertible matrix $Q_1 \in_R \mathbb{F}_2^{n-k \times n-k}$, two full-rank matrices $G_1, G_2 \in_R \mathbb{F}_2^{n-k \times n}$ and a $n \times n$ permutation matrices P_1, P_2.
– Define $\widetilde{H}_1 = Q_1 H_1 P_1$ and $\widetilde{H}_2 = G_1^T H_2 P_2 \oplus G_2^T H_1 P_1$.

Thus, we have :

– **Public Keys:** $\widetilde{H}_1$ & $\widetilde{H}_2$
– **Secret Keys:** $H_1,\ H_2,\ Q_1,\ P_1,\ P_2,\ G_1,\ G_2$

Encryption. On an message $m \in \mathbb{F}_2^{n_2}$ with $wt(m) = t_2$, the following steps constitute the encryption algorithm:

– Generate $r \in_R \mathbb{F}_2^{n_1}$, with $wt(r) = t_1$.
– $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}_{\mathcal{OS}}(1^\kappa)$, and compute $H_{\mathsf{vk}} = f_1(\mathsf{vk})$ and $P_{\mathsf{vk}} = f_2(\mathsf{vk})$.
– Define $K_1 = \widetilde{H}_1 P_{\mathsf{vk}}$ and $K_2 = H_{\mathsf{vk}} \widetilde{H}_2 P_{\mathsf{vk}}$.
– Define $c_1 = K_1[r|m]^T$, and $c_2 = K_2[r|m]^T$.
– Compute $\sigma = \mathsf{Sign}_{\mathcal{OS}}(\mathsf{sk}, (c_1, c_2))$, i.e., the one-time signature on (c_1, c_2) (where (c_1, c_2) is denoted as M) using the signing key sk.

The ciphertext that is sent is $c = (\mathsf{vk}, c_1, c_2, \sigma)$.

Decryption. The decryption on the ciphertext $c = (\mathsf{vk}, c_1, c_2, \sigma)$ is done as follows:

```
if ($\mathsf{Verify}_{\mathcal{OS}}(\mathsf{vk}, (c_1, c_2), \sigma) \rightarrow$ INVALID))
    return $\perp$
else
    Compute, $H_{\mathsf{vk}} \leftarrow f_1(\mathsf{vk})$, $P_{\mathsf{vk}} \leftarrow f_2(\mathsf{vk})$.
   if ($\mathsf{Decode}_{H_1}(Q_1^{-1} c_1) \rightarrow \perp$)
       return $\perp$.
    else
       $[r'|m'] \leftarrow P_{\mathsf{vk}}^T P_1^T \mathsf{Decode}_{H_1}(Q_1^{-1} c_1)$
       if($c_2 \neq H_{\mathsf{vk}} \widetilde{H}_2 P_{\mathsf{vk}} [r'|m']^T$)
           return $\perp$.
       else
           return $m'$
end
```

Correctness. The reciever on getting the ciphertext can verify the signature as the the verification key is attached along with the ciphertext components, then decode c_1 to obtain r and m. The receiver can verify the consistency of the retrieved randomness using c_2. Since, r and m maintain the weight constraints of the code used, we obtain the correct m (as long as the decoding algorithm is correct).

3.2 Proof for the Security of the System

The proof of security is based on the proof in [18]. It is claimed that every adversary has only a negligible advantage in the CCA-2 games under the standard model, provided the *Computational Syndrome Decoding* problem and *Goppa Code Distinguishability* are hard to solve, and the signature is one-time strongly unforgeable. $\widetilde{H}_{11}$, $\widetilde{H}_{12}$ be $n-k \times n_1$ and $n-k \times n_2$ matrices such that $\widetilde{H}_1 = [\widetilde{H}_{11}|\widetilde{H}_{12}]$. Similarly we define $\widetilde{H}_{21}$ & $\widetilde{H}_{22}$ such that $\widetilde{H}_2 = [\widetilde{H}_{21}|\widetilde{H}_{22}]$. It is seen that $c_1 = \widetilde{H}_{11}r^T \oplus \widetilde{H}_{12}m^T$ and $c_2 = \widetilde{H}_{21}r^T \oplus \widetilde{H}_{22}m^T$. Fischer and Stern [11] had proved that a generated syndrome is computationally indistinguishable from a randomly generated vector. The reduction along with the reduction cost was given by Nojima et. al. [18].

Theorem 1. *[18] If there exists an algorithm $\mathcal{D}$ which runs in time τ, such that*

$$Pr[r \in_R \mathbb{F}_2^{n_1},\ wt(r) = t_1,\ R \in_R \mathbb{F}_2^{n-k\times n_1} | \mathcal{D}(Rr^T, R) = 1] - Pr[s \in_R \mathbb{F}_2^{n-k},\ R \in_R \mathbb{F}_2^{n-k\times n_1} | \mathcal{D}(s, R) = 1] \ \geq\ \delta$$

then one can construct an algorithm $\mathcal{D}'$ running in time $\tau' = O(n_1^2(\tau + n_1^2)/\delta^2)$, such that $4\sqrt[3]{n_1 \cdot \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{SD}}(n_1, k)} \geq \delta$.

The following corollary is deduced from the above theorem.

Corollary 1. *[18] If there exists an algorithm $\mathcal{D}$ running in time τ for any $m \in_R \mathbb{F}_2^{n_2}$, $wt(m) = t_2$, such that*

$$Pr[r \in_R \mathbb{F}_2^{n_1},\ wt(r) = t_1,\ R \in_R \mathbb{F}_2^{n-k\times n} | \mathcal{D}(R_1 r^T \oplus R_2 m^T, R) = 1 \ \&\ R = [R_1|R_2]] - Pr[s \in_R \mathbb{F}_2^{n-k},\ R \in_R \mathbb{F}_2^{n-k\times n} | \mathcal{D}(s \oplus R_2 m^T, R) = 1 \ \&\ R = [R_1|R_2]] \ \geq\ \delta$$

then one can construct an algorithm $\mathcal{D}'$ running in time $\tau' = O(n_1^2(\tau + n_1^2)/\delta^2)$, and satisfying the inequality, $4\sqrt[3]{n_1 \cdot \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{SD}}(n_1, k)} \geq \delta$.

It also follows that if presented with two distinct $n-k \times n$ matrices R_1, R_2 as in the scheme, a syndrome decoding on the $2(n-k) \times n$ matrix $R^T = [R_1^T|R_2^T]$ (which is a parity check matrix of a $[n, 2k-n]$ code) is also possible.

Corollary 2. *If there exists an algorithm $\mathcal{D}$ running in time τ for any $m \in_R \mathbb{F}_2^{n_2}$, $wt(m) = t_2$, such that*

$$Pr\left[\begin{array}{c|r} r \in_R \mathbb{F}_2^{n_1},\ wt(r) = t_1 & \mathcal{D}((R_{11}r^T \oplus R_{12}m^T, R_1), \\ R_1 \in_R \mathbb{F}_2^{n-k\times n}\ \&\ R_2 \in_R \mathbb{F}_2^{n-k\times n} & (R_{21}r^T \oplus R_{22}m^T, R_2)) = 1 \\ & \&\ R_1 = [R_{11}|R_{12}]\ \&\ R_2 = [R_{21}|R_{22}] \end{array}\right] -$$

$$Pr\left[\begin{array}{c|r} s_1, s_2 \in_R \mathbb{F}_2^{n-k} & \mathcal{D}((s_1 \oplus R_{12}m^T, R_1), \\ R_1 \in_R \mathbb{F}_2^{n-k\times n}\ \&\ R_2 \in_R \mathbb{F}_2^{n-k\times n} & (s_2 \oplus R_{22}m^T, R_2)) = 1 \\ & \&\ R_1 = [R_{11}|R_{12}]\ \&\ R_2 = [R_{21}|R_{22}] \end{array}\right] \geq \delta$$

then one can construct an algorithm $\mathcal{D}'$ running in time $\tau' = O(n_1^2(\tau + n_1^2)/\delta^2)$, and satisfying the inequality, $4(\sqrt[3]{n_1 \cdot \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{SD}}(n_1, k)} + \sqrt[3]{n_1 \cdot \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{SD}}(n_1, 2k - n)}) \geq \delta$.

Next we formally argue the security of the protocol in the IND-CCA2 setting without the use of random oracles. The argument is essentially an extension of the reduction in [18] to the IND-CCA2 setting.

Theorem 2. *If there exists a PPT adversary $\mathcal{A}$ such that $Adv_{PKE,\mathcal{A}}^{cca2}(\kappa) \geq \delta$, there exists an algorithm $\mathcal{D}'$ such that $\mathsf{Adv}_{\mathcal{D}'}^{\mathsf{CD}}(n) + \mathsf{Succ}_{\mathcal{A}}^{\mathcal{OS}}(\mathsf{vk}^*) +$ $4(\sqrt[3]{n_1 \cdot \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{SD}}(n_1, k)} + \sqrt[3]{n_1 \cdot \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{SD}}(n_1, 2k - n)}) \geq \delta$.*

Proof: We construct such a challenger $\mathcal{D}'$ which simulates the cryptosystem (giving the adversary $\mathcal{A}$ the public parameters of the system and access to the decryption oracle) and uses the adversary $\mathcal{A}$ by giving it an appropriate challenge ciphertext. The simulation by $\mathcal{D}'$ is as follows:

Key Generation. For the parameters $[n, k, t]$ (also n_1, n_2, t_1, t_2), the parity matrix R of a random $[n, k, t]$ linear code, the injective functions f_1, f_2, the one-time signature scheme $\mathcal{OS}(\mathsf{KeyGen}_{\mathcal{OS}}, \mathsf{Sign}_{\mathcal{OS}}, \mathsf{Verfiy}_{\mathcal{OS}})$, the challenger generates the keys as follows:
- $(\mathsf{vk}^*, \mathsf{sk}^*) \leftarrow \mathsf{KeyGen}_{\mathcal{OS}}(1^\kappa)$.
- Randomly selects a $[n, k, 2t + 1]$Goppa code with parity check matrix H_2 respectively
- Randomly selects an invertible matrix $Q_1 \in_R \mathbb{F}_2^{n-k\times n-k}$, the full-rank matrix $G_2 \in_R \mathbb{F}_2^{n-k\times n}$ and a $n \times n$ permutation matrices P_1, P_2.
- Computes $H_{\mathsf{vk}^*} = f_1(\mathsf{vk}^*)$ and generates $G_1 \in \mathbb{F}_2^{n-k\times n}$ such that $H_{\mathsf{vk}^*} G_1^T = 0$. To generate such a G_1, one has to find a generator matrix G corresponding to H_{vk^*}, and select randomly $Q \in_R \mathbb{F}_2^{n-k\times k}$ and $G_1 = QG$. Hence G_1 contains $n-k$ code-words from the code generated by G, since on multiplying any k (dimension of code)length vector with the corresponding generator matrix G gives a code-word. Therefore $HG_1^T = 0$.
- Define $\widetilde{H}_1 = Q_1 R P_1$ and $\widetilde{H}_2 = G_1^T H_2 P_2 \oplus G_2^T R P_1$.

Decryption Oracle. The challenger has to simulate the decryption oracle, as it does not possess the trapdoor with respect to $\widetilde{H}_1$. The decryption oracle uses the trapdoor related to the key $\widetilde{H}_2$. The simulation is as follows:

Input: The ciphertext $c = (\mathsf{vk}, c_1, c_2, \sigma)$
Output: The message m.

```
if $(\mathsf{Verify}_{\mathcal{OS}}(\mathsf{vk}, c_1, c_2) == FALSE)$
  Return $\perp$
else
  if$(\mathsf{vk} == \mathsf{vk}^*)$                    // $\mathcal{D}'$ does not posses a trapdoor for $\mathsf{vk}^*$
    ABORT                                          // since $H_{\mathsf{vk}^*}G_1^T = 0$
  else
    $H_{\mathsf{vk}} \leftarrow f_1(\mathsf{vk})$, $P_{\mathsf{vk}} \leftarrow f_2(\mathsf{vk})$
    Compute $c_1' = Q_1^{-1}c_1$
    Compute $c_1'' = H_{\mathsf{vk}}G_2^T c_1'$
    Compute $y = c_2 \oplus c_1''$
    Compute $Q' = H_{\mathsf{vk}}G_1^T$
    if(Q' is not invertible)
      ABORT
    else
      if$(\mathsf{Decode}_{H_2}(Q'^{-1}y) \rightarrow \perp)$                 // Invalid ciphertext
        Return $\perp$
      else
        Compute $[r|m'] = P_{\mathsf{vk}}^T P_2^T \mathsf{Decode}_{H_2}(Q'^{-1}y)$
        if$(c_1 \neq \widetilde{H}_1 P_{\mathsf{vk}}[r|m']^T$ OR $c_2 \neq H_{\mathsf{vk}}\widetilde{H}_2 P_{\mathsf{vk}}[r|m']^T)$
          Return $\perp$                             // Ciphertext inconsistent
        else
          Return $m'$.
end
```

Challenge Ciphertext. Given the messages (m_0, m_1) each of length n_2 and weight t_2 challenger selects $(\mathsf{vk}^*, \mathsf{sk}^*)$ as the verification-signing key-pair for the one-time signature. The challenge ciphertext is setup as follows:

– Compute $H_{\mathsf{vk}^*} = f_1(\mathsf{vk}^*)$ and $P_{\mathsf{vk}^*} = f_2(\mathsf{vk}^*)$.
– Define $K_1 = \widetilde{H}_1 P_{\mathsf{vk}^*}$ and $K_2 = H_{\mathsf{vk}^*}\widetilde{H}_2 P_{\mathsf{vk}^*}$. Since $H_{\mathsf{vk}^*}G_1^T = 0$, the key $K_2 = H_{\mathsf{vk}^*}G_2 R P_1 P_{\mathsf{vk}^*}$. Also, we see that $K_1 = H_1 R P_1 P_{\mathsf{vk}^*}$. Hence for both the schemes, we find that it reduces to syndrome decoding on R.
– Randomly select b, i.e.,$b \in_R \{0, 1\}$.

Let $K_1 = [K_{11}|K_{12}]$ and $K_2 = [K_{21}|K_{22}]$ of appropriate sizes $n - k \times n_1$ and $n - k \times n_2$. Now to generate the challenge ciphertext:

```
if $(b ==1)$
    $r \in_R \mathbb{F}_2^{n_1}$ such that $wt(r) \leq t_1$. Compute $c_1 = K_1[r|m_1]^T$ and $c_2 = K_2[r|m_1]^T$.
else
    $s \in_R \mathbb{F}_2^{n-k}$.Compute $c_1 = Q_1 s \oplus K_{12}m_0^T$ and $c_2 = H_{\mathsf{vk}^*}G_2 s \oplus K_{22}m_0^T$.
```

The challenger computes $\sigma = \mathsf{Sign}_{\mathcal{OS}}(\mathsf{sk}^*, (c_1, c_2))$, and sends $(\mathsf{vk}^*, c_1, c_2, \sigma)$ as the challenge ciphertext. The adversary return b'. If $b == b'$ then $\mathcal{D}'$ outputs 1 else outputs 0.

Let Real denote the actual encryption scheme, and Sim denote the simulated version by $\mathcal{D}'$. It is noted that in Sim the key generation differs in the sense that the Goppa matrix H_1 is replaced by R. Also, the decryption oracle aborts in certain cases. We note that the ABORT scenario in the case that Q' is not invertible is negligible. This is so because, for every H_{vk} there are exactly 2^k elements in $\mathbb{F}_2^n$ that give 0 on multiplication. This is the code corresponding to H_{vk}, and is hence the a vector space. Thus it follows that every vector $v \in \mathbb{F}_2^{n-k}$ can be generate by exactly 2^k vectors in $\mathbb{F}_2^n$. Hence the probability that a vector $v \in \mathbb{F}_2^{n-k}$ is generated is $\frac{2^k}{2^n}$ i.e., $\frac{1}{2^{n-k}}$. It is noted that the probability distribution is exactly similar to when we select a vector $v \in \mathbb{F}_2^{n-k}$ uniformly at random. Hence, the probability of matrix Q' being non-invertible is the same as when generated completely at random,viz. $\mathsf{negl}(n)$. The other scenario to abort is the case at which $\mathsf{vk} = \mathsf{vk}^*$. In the first phase of the game, the probability of its occurrence is negligible in the size of the verification key space. But, after the challenge ciphertext is generated the adversary can query on altered ciphertexts with signature σ' verifiable using vk^*. Hence, it is seen that

$$Pr[b' == b|\mathsf{Real}] - Pr[b' == b|\mathsf{Sim}] \leq \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{CD}}(n) + \mathsf{Succ}_{\mathcal{A}}^{\mathcal{OS}}(\mathsf{vk}^*) \qquad (1)$$

From the corollary 2 it can derived that

$$Pr[b' == b|\mathsf{Sim}] - \frac{1}{2} \leq 4(\sqrt[3]{n_1 \cdot \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{SD}}(n_1, k)} + \sqrt[3]{n_1 \cdot \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{SD}}(n_1, 2k-n)}) \quad (2)$$

We know that $Pr[b == b'|\mathsf{Real}]$ is the probability of success of the adversary in the actual encryption scheme.

$$\mathsf{Adv}_{\mathsf{PKE},\mathcal{A}}^{cca2}(\kappa) = Pr[b == b'|\mathsf{Real}] - \frac{1}{2}$$

By equation 1,

$$\mathsf{Adv}_{\mathsf{PKE},\mathcal{A}}^{cca2}(\kappa) \leq \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{CD}}(n) + \mathsf{Succ}_{\mathcal{A}}^{\mathcal{OS}}(\mathsf{vk}^*) + Pr[b == b'|\mathsf{Sim}] - \frac{1}{2}$$

From equation 2

$$\begin{aligned}\mathsf{Adv}_{\mathsf{PKE},\mathcal{A}}^{cca2}(\kappa) \leq \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{CD}}(n) + \mathsf{Succ}_{\mathcal{A}}^{\mathcal{OS}}(\mathsf{vk}^*) + 4(\sqrt[3]{n_1 \cdot \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{SD}}(n_1, k)} +\\ \sqrt[3]{n_1 \cdot \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{SD}}(n_1, 2k-n)})\end{aligned}$$

Hence using algorithm $\mathcal{D}'$ we get

$$\mathsf{Adv}_{\mathcal{D}'}^{\mathsf{CD}}(n,k) + \mathsf{Succ}_{\mathcal{A}}^{\mathcal{OS}}(\mathsf{vk}^*) + 4(\sqrt[3]{n_1 \cdot \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{SD}}(n_1, k)} + \sqrt[3]{n_1 \cdot \mathsf{Adv}_{\mathcal{D}'}^{\mathsf{SD}}(n_1, 2k-n)}) \geq \delta \quad \square$$

From the result, we obtain that the advantage of the adversary, depends on the advantage of solving the *Goppa Code distinguishability* problem and *Syndrome Decoding* problem. For, parameters (n, k) with the appropriate n_1, n_2, t_1, t_2 for which $\mathsf{CD}(n, k)$, $\mathsf{SD}(n_1, k)$, & $\mathsf{SD}(n_1, 2k - n)$ are hard, the advantage for the adversary is negligible.

Validity of conditions for f_1 and f_2. A requirement for the security of the scheme is that, f_1 & f_2 are injective. The range of the function f_1 is of size $2^{n-k\times n}$ and the range of the function f_2 is of size $n!$. Clearly, the range of the function is of size comparable to that of the domain, if not much larger (since, size of domain must be $O(2^{\kappa})$). Hence, injective mappings can be realised for such domains and ranges.

3.3 Parameters

From, the previous section, we have seen that the selection of parameters is important in defining the negligible advantage an adversary has in solving the syndrome decoding problem. Clearly, for $\mathsf{SD}(n_1, 2k - n)$ to be hard, we have to select a $k > \frac{n}{2}$. Since, the required codes need not have a very high rate, the distinguisher attack [9] does not hold. Hence, the parameters that are generally used for Niederreiter encryption scheme, can be used for the proposed scheme too. The table 1, presents the (n_1, k) parameters along with the appropriate n_2 as message size, and the binary work factor for syndrome decoding for (n_1, k) and $(n_1, 2k - n)$. Here binary work factor, is $\log_2(\textit{time taken})$. The work factors are *estimated* according to the lower bound complexity given in [10]. For, the given parameters, Goppa codes are *indistinguishable* [9]. The security of the scheme for the given parameters increases with decrease in n_2.

Table 1. Parameters for the given scheme, and corresponding work factors for solution of Syndrome Decoding problem, estimated according to [10]

(n, k)	(n_1, n_2)	**Security factor for** (n_1, k)	**Security factor for** $(n_1, 2k - n)$
(4096,3604)	(4046,50)	128.60	90.08
	(3996,100)	124.96	87.31
	(3896,200)	120.65	83.50
	(3796,300)	116.29	80.44

3.4 Comparison with Other Schemes

It is seen that the proposed scheme is IND-CCA2 secure in the standard model, without much change in the parameters. The comparison of the proposed schemes with existing schemes are presented in table 2.To the best of our knowledge, this is the first Niederreiter variant that is IND-CCA2 secure in the standard model without κ repetition.

Table 2. Comparison with other code-based CCA-2 cryptosystems

Scheme	**Public-key (bits)**	**Secret key (bits)**	**Ciphertext**	**Encrypt Complexity**	**Decrypt complexity**
Dowsley et al.[8]	$2\kappa\times$MP	$2\kappa\times$MS	$\kappa\times$ MC	$\kappa\times$ME	1 MD + $\kappa\times$ME
Freeman et al. [12]	$2\kappa\times$NP	$2\kappa\times$NS	$\kappa\times$NC	$\kappa\times$NE	1 ND + $\kappa\times$ NE
Proposed Scheme	1 NP + $1(n\times n)$ Matrix	$2\times$NS	$2\times$NC	$2\times$NE + 1 MM	1 ND + $2\times$NE + 1 MM

(MP,MS)- McEliece (Public Key, Secret Key), (NP,NS) - Niederreiter (Public Key, Secret Key), (ME,MD) - McEliece (Encryption Complexity, Decryption Complexity), (NE,ND) - Niederreiter (Encryption Complexity, Decryption Complexity), MC - McEliece ciphertext size, NC -Niederreiter ciphertext size MM - Matrix Multiplication.

4 Conclusion

In the paper, we propose an efficient IND-CCA2 secure code-based encryption scheme in the standard model. The scheme is the first such scheme, that does not use the κ repetition paradigm [23]. Thus, the scheme has avoided the inherent costs incurred by the existing schemes [8,12] and is more efficient, because it requires at most two repetitions of the underlying Niederreiter encryption scheme and any one-time strongly unforgeable signature.

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient Lattice (H)IBE in the Standard Model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010)
2. Barreto, P.S.L.M., Misoczki, R., Simplício Jr., M.A.: One-time signature scheme from syndrome decoding over generic error-correcting codes. Journal of Systems and Software 84(2), 198–204 (2011)
3. Berson, T.A.: Failure of the McEliece Public-Key Cryptosystem under Message-Resend and Related-Message Attack. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 213–220. Springer, Heidelberg (1997)
4. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. J. ACM 51(4), 557–594 (2004)
5. Canetti, R., Halevi, S., Katz, J.: Chosen-Ciphertext Security from Identity-Based Encryption. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 207–222. Springer, Heidelberg (2004)
6. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Transactions on Information Theory 22, 644–654 (1976)
7. Dolev, D., Dwork, C., Naor, M.: Nonmalleable cryptography. SIAM J. Comput. 30(2), 391–437 (2000)

8. Dowsley, R., Müller-Quade, J., Nascimento, A.C.A.: A CCA2 Secure Public Key Encryption Scheme Based on the McEliece Assumptions in the Standard Model. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 240–251. Springer, Heidelberg (2009)
9. Faugére, J.-C., Otmani, A., Perret, L., Tillich, J.-P.: Algebraic Cryptanalysis of McEliece variants with compact keys – toward a complexity analysis. In: SCC 2010: Proceedings of the 2nd International Conference on Symbolic Computation and Cryptography, pp. 45–55. RHUL (June 2010)
10. Finiasz, M., Sendrier, N.: Security Bounds for the Design of Code-Based Cryptosystems. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 88–105. Springer, Heidelberg (2009)
11. Fischer, J.-B., Stern, J.: An Efficient Pseudo-random Generator Provably as Secure as Syndrome Decoding. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 245–255. Springer, Heidelberg (1996)
12. Freeman, D.M., Goldreich, O., Kiltz, E., Rosen, A., Segev, G.: More Constructions of Lossy and Correlation-Secure Trapdoor Functions. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 279–295. Springer, Heidelberg (2010)
13. Niederreiter, H.: Knapsack-type cryptosystems and algebraic coding theory. Prob. Contr. Inform. Theor. 15, 159–166 (1986)
14. Håstad, J.: Solving simultaneous modular equations of low degree. SIAM J. Comput. 17(2), 336–341 (1988)
15. Kobara, K., Imai, H.: Semantically Secure McEliece Public-Key Cryptosystems-Conversions for McEliece PKC. In: Kim, K. (ed.) PKC 2001. LNCS, vol. 1992, pp. 19–35. Springer, Heidelberg (2001)
16. Li, Y.X., Deng, R.H., Wang, X.M.: On the equivalence of McEliece's and Niederreiter's public-key cryptosystems. IEEE Transactions on Information Theory 40(1), 271–273 (1994)
17. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: STOC, pp. 427–437. ACM (1990)
18. Nojima, R., Imai, H., Kobara, K., Morozov, K.: Semantic security for the McEliece cryptosystem without random oracles. Des. Codes Cryptography 49(1-3), 289–305 (2008)
19. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. In: Dwork, C. (ed.) STOC, pp. 187–196. ACM (2008)
20. Rackoff, C., Simon, D.R.: Non-interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 433–444. Springer, Heidelberg (1992)
21. Rivest, R.L., Shamir, A., Adleman, L.M.: A method for obtaining digital signatures and public-key cryptosystems (reprint). Commun. ACM 26(1), 96–99 (1983)
22. McEliece, R.J.: A public-key cryptosystem based on algebraic coding theory. JPL DSN Progress Report, 114–116 (1978)
23. Rosen, A., Segev, G.: Chosen-Ciphertext Security via Correlated Products. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 419–436. Springer, Heidelberg (2009)
24. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. 26(5), 1484–1509 (1997)

Zero-Knowledge Protocols for the McEliece Encryption

Kirill Morozov and Tsuyoshi Takagi

Institute of Mathematics for Industry, Kyushu University, Japan
{morozov,takagi}@imi.kyushu-u.ac.jp

Abstract. We present two zero-knowledge protocols for the code-based McEliece public key encryption scheme in the standard model. Consider a prover who encrypted a plaintext m into a ciphertext c under the public key pk. The first protocol is a proof of plaintext knowledge (PPK), where the prover convinces a polynomially bounded verifier on a joint input (c, pk) that he knows m without actually revealing it. This construction uses code-based Véron's zero-knowledge identification scheme. The second protocol, which builds on the first one, is a verifiable McEliece encryption, were the prover convinces a polynomially bounded verifier on a joint input (c, pk, m) that c is a valid encryption of m, without performing decryption. These protocols are the first PPK and the first verifiable encryption for code-based cryptosystems.

1 Introduction

The McEliece public key encryption (PKE) scheme [26] is the first code-based cryptosystem. It uses the error-correcting codes by Goppa [20,25]. Security of the McEliece PKE is based on hardness of the problems related to general decoding [5,31]. Breaking of the McEliece PKE is believed to be infeasible for properly chosen parameters [12,13,7], even for adversaries equipped with quantum computers [6]. The later fact makes this cryptosystem a prospective candidate for the postquantum world. In fact, it is also argued by Bernstein et al [7, App. A] that the McEliece PKE is a prospective cryptosystem due to its good asymptotic performance.

Informally, a proof of plaintext knowledge (PPK) for an encryption scheme with public key pk, allows a prover P to prove knowledge of the plaintext m, corresponding to the ciphertext $c = Enc_{pk}(m)$, to a verifier V on the public inputs pk and c. Moreover, if such the proof is zero-knowledge (ZK), it will not reveal any additional information on m.

Informally, a verifiable encryption with respect to some binary relation R on the plaintexts is a zero-knowledge proof on public inputs pk, c, and δ that allows P to convince V that c is a ciphertext of m under pk such that $(m, \delta) \in R$.

1.1 Our Contributions

– We present a computational zero-knowledge PPK for the McEliece PKE using Véron's ZK identification scheme [35].

– Using this PPK, we also construct a verifiable IND-CPA McEliece encryption for equality relation by introducing a computational ZK proof of the statement "ciphertext c decrypts to the plaintext m".

1.2 Related Works

Proof of Plaintext Knowledge. PPK were introduced by Aumann and Rabin [1] (as attributed in [23]), and later studied by Katz [23], who presented PPK for RSA, Rabin, ElGamal and Paillier cryptosystems. The first PPK for a lattice-based Ajtai-Dwork PKE is due to Goldwasser and Kharchenko [19]. Xagawa et al [36] presented PPK for the two variants of the lattice-based Regev's cryptosystem. Xagawa and Tanaka presented that for NTRU [37] using a modification of Stern's code-based ZK identification scheme [34]. Bendlin and Damgård [3] presented a PPK for a variant of Regev's cryptosystem. Compared to the previous lattice-based constructions (as well as to our protocols) the latter scheme is constant-round that is achieved using the "multiparty computation in the head" paradigm of Ishai et al [21].

Stern's scheme [34] was used by Kobara et al [24] for enforcing correct behavior of a sender in code-based oblivious transfer. They even suggested verifiable encryption as a possible application for their technique, but no formal treatment of this subject was made in their work.

Verifiable Encryption. Verifiable encryption was introduced by Stadler [33] in the context of publicly verifiable secret sharing, and later generalized by Asokan et al [2] with application to fair exchange of digital signatures. Developments on this topic include further generalizations by Camenisch and Damgård [8] and Camenisch and Shoup [9].

We emphasize that none of the previous works on the above topics considered code-based PKE.

Note that assuming that one-way functions exist, one could achieve the results presented in this work using general zero-knowledge proofs for NP-statements [18], however such constructions would be prohibitively inefficient.

1.3 Discussion of Our Contributions

We present a computational zero-knowledge PPK for the McEliece PKE by showing that Véron's ZK identification scheme [35] (that is, in a sense, a dual of Stern's scheme [34]) can be directly used as PPK for the McEliece encryption. The witness in this proof is both the plaintext and the (random) error vector. Using Véron's scheme rather than Stern's (as in [24]), we avoid pre-computation on the public data.

An immediate application of this result is the interactive chosen-ciphertext secure encryption. Here, the sender uses an IND-CPA secure PKE to encrypt a message for the receiver, who must be online. Along with transmitting the ciphertext, the sender also uses the interactive PPK to convince the receiver that he knows the message. According to the observation by Katz [23], this construction results in an interactive IND-CCA1 PKE [15,16]. Combined with the IND-CPA secure McEliece encryption by Nojima et al [29], this yields the first code-based interactive IND-CCA1 PKE in the standard model.

Using the above mentioned PPK, we also construct a verifiable IND-CPA McEliece encryption for equality relation. Note that although the original McEliece encryption is not deterministic, given pk and $c = Enc_{pk}(m)$, it is trivial to check whether or not c is a ciphertext of m. Therefore, for verifiable encryption, we use an IND-CPA secure McEliece encryption [29].

It is interesting to note that in the lattice-based constructions [19,36], one first constructs a verifiable encryption for equality relation, and then use it as a building block for PPK, while in our case it works the other way around.

In our constructions, we assume that both the prover and the verifier are assured that the public key pk is valid. This assumption will require a trusted third party who generates public keys – this can be, for instance, an entity in the public key infrastructure.

The proofs of Stern's [34] and Véron's schemes [35] are in the random oracle model. In order to avoid such the strong assumption, we employ the later scheme with (efficient) computationally hiding and statistically binding commitment scheme based on hardness of syndrome decoding, as presented in [11].

2 Preliminaries

Let us fix some notation. Denote by "$\oplus$" the bitwise exclusive-or. For an ordered subset $\{j_i, \ldots, j_m\} = J \subseteq \{1, \ldots, n\}$, we denote the vector $(x_{j_1}, \ldots, x_{j_m}) \in \mathbb{F}_2^m$ by x_J. Similarly, we denote by M_J the submatrix of a $(k \times n)$ matrix M consisting of the columns corresponding to the indexes of J. A concatenation of vectors x and y is written as $(x|y)$. We denote by $x \overset{\$}{\leftarrow} \mathcal{X}$ a uniformly random selection of an element from its domain $\mathcal{X}$. A set of $(n \times n)$ permutation matrices is denoted by $\mathcal{S}_n$.

We denote by $\langle A(a), B(b) \rangle(c)$ a random variable representing the output of a Turing machine B following an execution of an interactive two-party protocol between a Turing machine A with private input a and B with private input b on joint input c, where A and B have uniformly distributed random tapes. If a party, say A, has no input, then we omit the input by writing just A (instead of $A(a)$) in the above notation.

In our two-party protocols, we will denote an honest prover by P and an honest verifier by V, while a dishonest party will be denoted by $\widetilde{\mathsf{P}}$ and $\widetilde{\mathsf{V}}$, respectively.

We call a function $\epsilon(n)$ *negligible in* n, if $\epsilon(n) = 2^{-\omega(\log n)}$. We call a probability $1 - \epsilon(n)$ *overwhelming*, when $\epsilon(n)$ is negligible.

Occasionally, we omit the mentioning of a security parameter. In these cases, by saying that a quantity is negligible (overwhelming), we mean that it is *negligible (overwhelming) in the security parameter.*

For the relevant topics in coding theory we refer the reader to [30,25].

2.1 Security Assumptions

Definition 1 (Syndrome Decoding (SD) Problem).
Input: $H \xleftarrow{\$} \mathbb{F}_2^{(n-k)\times n}$, $y \xleftarrow{\$} \mathbb{F}_2^{n-k}$ *and* $0 < t \in \mathbb{N}$.
Output: $s \in \mathbb{F}_2^n$ *such that* $w_H(s) \leq t$, $Hs^T = y$.

This problem was shown to be NP-complete by Berlekamp et al [5]. Its equivalent dual version can be formulated as follows.

Definition 2 (General Decoding (G-SD) Problem).
Input: $G \xleftarrow{\$} \mathbb{F}_2^{k\times n}$, $y \xleftarrow{\$} \mathbb{F}_2^{n}$ *and* $0 < t \in \mathbb{N}$.
Output: $x \in \mathbb{F}_2^k$, $e \in \mathbb{F}_2^n$ *s.t.* $w_H(e) \leq t$, $xG \oplus e = y$.

The following two problems use the quantities defined in the next subsection. No polynomial-time algorithm is known for these problems [12,13,7].

Definition 3 (McEliece Problem).
Input: A McEliece public key (G^{pub}, t), *where*
$G^{pub} \in \mathbb{F}_2^{k\times n}$, $0 < t \in \mathbb{N}$; *and a McEliece ciphertext* $c \in \mathbb{F}_2^n$.
Output: $m \in \mathbb{F}_2^k$ *such that* $d_H(mG^{pub}, c) = t$.

Definition 4 (Goppa Code Distinguishing (GD) Problem).
Input: $R \in \mathbb{F}_2^{k\times n}$.
Decide: Is R *a generator matrix of an* (n,k) *irreducible Goppa code, or of a random* (n,k)*-code?*

2.2 McEliece Cryptosystem

For a survey on code-based PKE and related schemes we refer the reader to the work by Engelbert et al [12].

The McEliece PKE consists of the following triplet of algorithms $(\mathcal{K}, \mathcal{E}, \mathcal{D})$:

- Security parameters: $n, t \in \mathbb{N}$.
- Key generation algorithm $\mathcal{K}$: On input n, t, generate the following matrices:
  - $G \in \mathbb{F}_2^{k\times n}$ – the generator matrix of an irreducible binary Goppa code correcting up to t errors. Its decoding algorithm is denoted as Dec.
  - $S \in \mathbb{F}_2^{k\times k}$ – a random non-singular matrix.
  - $P \in \mathbb{F}_2^{n\times n}$ – a random permutation matrix (of size n).
  - $G^{pub} = SGP \in \mathbb{F}_2^{k\times n}$.

  Output the public key $pk = (G^{pub}, t)$ and the secret key $sk = (S, G, P, \mathsf{Dec})$.
- Encryption algorithm $\mathcal{E}$: On input a plaintext $m \in \mathbb{F}_2^k$ and the public key pk, choose a vector $e \in \mathbb{F}_2^n$ of weight t at random, and output the ciphertext

$$c = mG^{pub} \oplus e.$$

- Decryption algorithm $\mathcal{D}$: On input c and the secret key sk, calculate:

- $cP^{-1} = (mS)G \oplus eP^{-1}$.
- $mSG = \mathsf{Dec}(cP^{-1})$.
- Let $J \subseteq \{1, \ldots, n\}$ be s.t. G_J is invertible.
  Output $m = (mSG)_J(G_J)^{-1}S^{-1}$.

It is easy to check that the decryption algorithm correctly recovers the plaintext: Since in the first step of decryption, the permuted error vector $\mathbf{e}P^{-1}$ is again of weight t, the decoding algorithm Dec successfully corrects these errors in the next step.

Randomized McEliece Encryption. In the standard model, Nojima et al [29] show that the McEliece encryption with a random padding of the plaintext (which is multi-bit) is IND-CPA secure under hardness of the learning parities with noise (LPN) problem[1] and GD problem.

A little more formally, the Randomized McEliece encryption is constructed in the same way as described above, except that the ciphertext $c = (r|m)G^{pub} \oplus e$, where $r \overset{\$}{\leftarrow} \{0,1\}^{k_0}$, $m \in \{0,1\}^{k_1}$, $k = k_0 + k_1$. A particular choice of k_0 and k_1 is discussed in [29].

2.3 Proof of Plaintext Knowledge

In this subsection, we closely follow the presentation of [23]. For a public key cryptosystem $(\mathcal{K}, \mathcal{E}, \mathcal{D})$, denote by $c = \mathcal{E}_{pk}(m; R)$ an encryption of a plaintext m under public key pk using randomness R. We will call (m, R) a *witness* to the decryption of c under pk. Informally, in a PPK protocol, a sender P proves to a receiver V the knowledge of a witness to the decryption for some ciphertext c under the known public key pk.

Definition 5. *Let $\Pi = (\mathsf{P}, \mathsf{V})$ be a tuple of PPT algorithms. Π is a* proof of plaintext knowledge *for encryption scheme $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ if the following conditions hold:*

(Completeness) *For all pk output by $\mathcal{K}(1^n)$ and all c with witness w to the decryption of c under pk, we have that* $\Pr[\langle \mathsf{P}(w), \mathsf{V}\rangle(pk, c) = 1]$. *(When V outputs 1 we say it* accepts.*)*

(Soundness) *For all pk output by $\mathcal{K}(1^n)$, all c produced under pk, and for any $\widetilde{\mathsf{P}}$, we have that* $\Pr[\langle \widetilde{\mathsf{P}}, \mathsf{V}\rangle(pk, c) = 1]$ *is negligible.*

(Zero-knowledge) *There exists a PPT Turing machine $\mathcal{SIM}$ (called a* simulator*) such that, for all pk output by $\mathcal{K}(1^n)$, all PPT $\widetilde{\mathsf{V}}$, and all w, the following distributions are computationally indistinguishable:*

$$\{c = \mathcal{E}_{pk}(m; R) : \langle \mathsf{P}(w), \widetilde{\mathsf{V}}\rangle(pk, c)\},$$

$$\{c = \mathcal{E}_{pk}(m; R) : \langle \mathcal{SIM}, \widetilde{\mathsf{V}}\rangle(pk, c)\}.$$

[1] See e.g. [29] for a formal definition of LPN problem – it is similar to G-SD problem except that in the error vector e, each bit has Bernoulli distribution with fixed p, $0 < p < 0.5$.

2.4 Verifiable Encryption

We adapt the following definition from [8].

Definition 6. *Let* $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ *be a public key encryption scheme, let* R *be a binary relation and let* $L_R = \{x | \exists w : (x, w) \in R\}$. *A* secure verifiable encryption *scheme for a relation* R *consists of a two-party protocol between* P *and* V *s.t. the following conditions hold:*
(Completeness) *For all* pk *output by* $\mathcal{K}(1^n)$ *and all* $x \in L_R$*, we have* $\Pr[\langle \mathsf{P}(x), \mathsf{V}\rangle(pk) = 1]$. *(When* V *outputs 1 we say it* accepts.*)*
(Soundness) *For all* pk *output by* $\mathcal{K}(1^n)$*, all* $x' \notin L_R$*, and for any* $\widetilde{\mathsf{P}}$*,* $\Pr[\langle \widetilde{\mathsf{P}}(x'), \mathsf{V}\rangle(pk, c) = 1]$ *is negligible.*
(Zero-knowledge) *There exists a PPT simulator* $\mathcal{SIM}$ *such that, for all* pk *output by* $\mathcal{K}(1^n)$*, all PPT* $\widetilde{\mathsf{V}}$*, and all* $x \in L_R$*, the following distributions are computationally indistinguishable:*

$$\{x \in L_R : \langle \mathsf{P}(x), \widetilde{\mathsf{V}}\rangle(pk)\},\ \{x \in L_R : \langle \mathcal{SIM}, \widetilde{\mathsf{V}}\rangle(pk)\}.$$

Note that this definition captures only the properties related to verifiability. We implicitly assume that a scheme in question is indeed a public-key encryption scheme. For a formal definition of the latter, see e.g. [17, Ch. 5].

2.5 Commitments

Zero-knowledge proof systems use commitments as a building block. A commitment scheme consists of two stages. In the first one, called *committing*, the sender P provides the receiver V with an evidence about his data b. The cheating receiver $\widetilde{\mathsf{V}}$ cannot learn b before the second stage, called *opening*, when P reveals b to V. The cheating sender $\widetilde{\mathsf{P}}$ cannot successfully open anything other than b. Let us denote by $[\mathsf{P}, \mathsf{V}]_{A,st}$ the *view* of the party $A \in \{\mathsf{P}, \mathsf{V}\}$ at the stage st, which is a concatenation of all the messages sent and received by A, along with its local randomness.

We adapt the following definition from [11].

Definition 7. *A protocol is said to* securely implement string commitment*, if at the end of its execution by PPT Turing machines* P *(with input* $b \in \mathbb{F}_2^l$*,* $l \in \mathbb{N}$*) and* V*, the following properties hold:*
(Correctness) $\Pr[\langle \mathsf{P}(b), \mathsf{V}\rangle = 1]$ *with overwhelming probability.*
(Hiding) *For any PPT* $\widetilde{\mathsf{V}}$*, any* $l \in \mathbb{N}$*, any* $b \in \mathbb{F}_2^l$ *and* $b' \in \mathbb{F}_2^l$ *such that* $b' \neq b$*, after the committing stage, but before the opening stage, the distributions*

$$[\mathsf{P}(b), \widetilde{\mathsf{V}}]_{\widetilde{\mathsf{V}},Commit} \quad \textit{and} \quad [\mathsf{P}(b'), \widetilde{\mathsf{V}}]_{\widetilde{\mathsf{V}},Commit}$$

are computationally indistinguishable.
(Binding) *For any* $\widetilde{\mathsf{P}}$*, any* $l \in \mathbb{N}$*, and* $b' \in \mathbb{F}_2^l$ *there exists* $b \in \mathbb{F}_2^l$ *which can be computed by* P *after the committing stage, such that the probability*

$$\Pr[\langle \widetilde{\mathsf{P}}(b'), \mathsf{V}\rangle = 1]$$

is negligible.

In the random oracle model, a string commitment which is both computationally hiding and binding can be implemented using (idealized) cryptographic hash functions. We avoid this additional strong assumption by employing a computationally hiding and statistically binding commitment based on syndrome decoding, which was suggested by Dowsley et al [11]. They proposed to use Naor's bit commitment scheme [27] based on pseudorandom generator, which, in turn, can be constructed assuming hardness of SD problem, as proved by Fischer and Stern [14].

3 PPK for McEliece Encryption

Our proof of knowledge for the McEliece encryption is based on Véron's zero-knowledge identification scheme [35]. We make the following modifications to it – instead of a generator matrix of the random code, we use that of the irreducible (n, k) Goppa code as described in Section 2.2, and set a weight of the error vector to be exactly t.

Our main observation is that the security proof of Véron's scheme [35] is valid for any code, for which G-SD problem is hard, not just a random one. Therefore, replacing a random code with the McEliece public key, and an assumption on the hardness of G-SD problem with that on the hardness of the McEliece problem, we preserve the validity of the original proof.

Remark 1. Note that we do not need to assume hardness of the Goppa Distinguishing problem for the proof itself.

Remark 2. In the following protocol, the probability for $\widetilde{\mathsf{P}}$ to break soundness (i.e. to make V accept the proof without knowledge of the witness (m, e)) is $2/3$. It can be reduced to an arbitrary small value $(2/3)^s$ by iterating the protocol s times.

Witness: (m, e), $m \in \mathbb{F}_2^k$, $e \in \mathbb{F}_2^n$, $w_H(e) = t$, where the parameters n, k, t are described in Section 2.2.

Common data: $(G^{pub} \in \mathbb{F}_2^{k\times n}, t)$ – the McEliece public key, and $c = mG^{pub} \oplus e$ – the McEliece PKE ciphertext (as described in Section 2.2).

Protocol 1 (McEliece PPK).

1. P computes $u \overset{\$}{\leftarrow} \mathbb{F}_2^k$, $T \overset{\$}{\leftarrow} \mathcal{S}_n$ and sends three commitments:
   – $C_1 = com(T)$,
   – $C_2 = com((u \oplus m)G^{pub}T)$,
   – $C_3 = com((uG^{pub} \oplus c)T)$.
2. V sends $b \overset{\$}{\leftarrow} \{0, 1, 2\}$.
3. In this step, V checks the validity of the quantities presented by P, and rejects if it does not hold:

– If $b = 0$,
   – P sends T, $u \oplus m$, and opens C_1, C_2.
   – V checks validity of C_1 and C_2 (using G^{pub}).

– If $b = 1$,
  – P sends $(u \oplus m)G^{pub}T$, eT, and opens C_2, C_3.
  – V checks that $w_H(eT) = t$ and validity of C_2, C_3
  (using that $(u \oplus m)G^{pub}T \oplus eT = (uG^{pub} \oplus c)T$).
– If $b = 2$,
  – P sends T, u, and opens C_1, C_3.
  – V checks the validity of C_1, C_3.

Denote a protocol consisting of s independent iterations of Protocol 1 by $\mathsf{PPK}(G^{pub}, c; m, e)$, with some appropriately chosen s.

Theorem 1. *Protocol* $\mathsf{PPK}(G^{pub}, c; m, e)$ *is a proof of plaintext knowledge for the McEliece cryptosystem according to Definition 5 assuming hardness of the McEliece problem.*

Proof. We closely follow the proof in [35].

Completeness. It is easy to check that P knowing a valid (m, e) for G^{pub} can answer any of the queries correctly. Hence, we have $\Pr[\langle \mathsf{P}(w), \mathsf{V}\rangle(pk, c) = 1]$.

Soundness. First, we prove the following lemma.

Lemma 1. *If* V *accepts* $\widetilde{\mathsf{P}}$*'s proof with probability at least* $(\frac{2}{3})^s + \epsilon$*, then there exists a PPT algorithm* M *which, with overwhelming probability, computes a witness* (m, e)*.*

Proof. Let $\mathcal{T}$ be an execution tree of the protocol $(\widetilde{\mathsf{P}}, \mathsf{V})$ corresponding to all possible questions of V, when $\widetilde{\mathsf{P}}$ has a random tape RA. V may ask 3 possible questions at each stage. First, we show that as long as the binding property of the commitment holds, a witness (m, e) can be computed from a vertex with 3 descendants. Next, we show that a PPT M can find such a vertex in $\mathcal{T}$ with overwhelming probability.

Let v be a vertex with 3 descendants. This corresponds to a situation, where 3 commitments C_1, C_2, C_3 have been made and where the three queries were correctly answered.

Let T' and $u' \oplus m'$ be the answers to the query $b = 0$, y'', e'' – to the query $b = 1$, T''', u''' – to the query $b = 2$.

We have $w_H(e'') = t$, $T' = open(C_1) = T'''$,
$(u' \oplus m')G^{pub}T' = open(C_2) = y''$,
$y'' \oplus e'' = open(C_3) = (u'''G^{pub} \oplus c)T'''$.

Therefore, either $\widetilde{\mathsf{P}}$ was able to violate the binding property of the commitment, or we have $c = (u' \oplus m' \oplus u''')G^{pub} \oplus e''(T')^{-1}$, where $e''(T')^{-1}$ is a word of length n and weight t. Therefore, $(u' \oplus m' \oplus u''', e''(T')^{-1})$ is a valid witness.

Next, we show that the probability for $\mathcal{T}$ to have a vertex with 3 descendants is at least ϵ. Let us consider the random tape RA of $\widetilde{\mathsf{P}}$ as a set of μ elements, from which $\widetilde{\mathsf{P}}$ randomly picks its values and let $Q = \{1, 2, 3\}$. These two sets are considered as probability spaces, both of them with uniform distribution.

A pair $(a, b) \in (RA \times Q)^s$ represents the commitments, queries and answers communicated between $\widetilde{\mathsf{P}}$ and V in the protocol. We will call (a, b) a *valid* pair, if the execution of $(\widetilde{\mathsf{P}}, \mathsf{V})$ leads to the success state.

Let V be the subset of $(RA \times Q)^s$ composed of all the valid pairs. By the hypothesis of the lemma,

$$\frac{|V|}{|(RA \times Q)^s|} \geq \left(\frac{2}{3}\right)^s + \epsilon.$$

Let $\Omega_s \subset RA^s$ such that:

– If $a \in \Omega_s$, then $2^s + 1 \leq |\{b : (a, b) \text{ are valid}\}| \leq 3^s$,
– If $a \in RA^s \setminus \Omega_s$, then $0 \leq |\{b : (a, b) \text{ are valid}\}| \leq 2^s$.

Then, we write $V = \{\text{valid } (a, b), a \in \Omega_s\} \cup \{\text{valid } (a, b), a \in RA^s \setminus \Omega_s\}$, therefore $|V| \leq |\Omega_s| \cdot 3^s + (\mu^s - |\Omega_s|) \cdot 2^s$, so by noting that $|RA^s| = \mu^s$ and $|Q^s| = 3^s$ it follows that

$$\frac{|V|}{|(RA \times Q)^s|} \leq \left(\frac{|\Omega_s|}{|RA^s|} + 2^s \left(3^{-s} - \frac{|\Omega_s|}{|RA \times Q)^s|}\right)\right) \leq \frac{|\Omega_s|}{|RA^s|} + \left(\frac{2}{3}\right)^s, \quad (1)$$

and therefore $|\Omega_s|/|RA^s| \geq \epsilon$. This shows that the probability that $\widetilde{\mathsf{P}}$ answers to (at least) $2^s + 1$ V's queries, by choosing random values, is bigger than ϵ.

Now, if more than $2^s + 1$ queries are correctly answered by $\widetilde{\mathsf{P}}$, $\mathcal{T}(RA)$ has at least $2^s + 1$ leaves, i.e. $\mathcal{T}(RA)$ has at least one vertex with 3 descendants.

Therefore, by rewinding $\widetilde{\mathsf{P}}$ $1/\epsilon$ times, it is possible to find an execution tree with a vertex having 3 descendants with probability arbitrary close to 1. This concludes the proof of the lemma. □

Unless the binding property of the commitment was violated, the conclusion of this lemma contradicts hardness of the McEliece problem. It follows that $\Pr[\langle \widetilde{\mathsf{P}}, \mathsf{V} \rangle (pk, c) = 1] \leq (2/3)^s + \epsilon$, which is negligible in n and s.

Zero-Knowledge. Let us denote by $\mathcal{R}_{\mathsf{P},\mathsf{V}}$ the communication tape for P and V, that is a concatenation of all bits they exchanged during the protocol. We consider the probability distributions on $\mathcal{R}_{\mathsf{P},\mathsf{V}}$.

Proposition 1. *Protocol 1 is zero-knowledge according to Definition 5 assuming hardness of the McEliece problem.*

Proof. In order to simulate $\widetilde{\mathsf{V}}$, we have to assume that it will choose a particular cheating strategy depending on the information received from P. Let us denote this strategy by $St(C_1, C_2, C_3) \in \{0, 1, 2\}$.

Consider the following two functions: $\phi_m : \mathbb{F}_2^k \to \mathbb{F}_2^k$, $\phi_m(u) = u \oplus m$, which is an automorphism of $\mathbb{F}_2^k$ and $\psi : \mathbb{F}_2^k \to \mathbb{F}_2^n$, $\psi(u) = uG^{pub}$, which is an isomorphism of $\mathbb{F}_2^k$ into the code generated by G^{pub}.

The following PPT algorithm $\mathcal{SIM}$ produces a communication tape, whose probability distribution is indistinguishable from that of a communication tape produced by the honest parties.

1. $\mathcal{SIM}$ randomly picks a query $b \in \{0,1,2\}$.
   - If $b = 0$, $\mathcal{SIM}$ chooses $y \overset{\$}{\leftarrow} \mathbb{F}_2^k$, $T \overset{\$}{\leftarrow} \mathcal{S}_n$, computes $C_1 = com(T)$, $C_2 = com(yG^{pub}T)$ and sets C_3 to be a random binary vector of appropriate length.
     Let $COM = (C_1|C_2|C_3)$ and $Ans = (y|T)$, here we assume a representation of $T \in \mathcal{S}_n$ as a binary vector by concatenating its rows. Note that y and $u \oplus m$ have the same probability distribution, since for some $z \in \mathbb{F}_2^k$, and $u \overset{\$}{\leftarrow} \mathbb{F}_2^k$, we have $\Pr[u \oplus m = z] = \Pr[u = \phi_m^{-1}(z)] = 2^{-k}$ $= \Pr[y = z]$.
   - If $b = 1$, $\mathcal{SIM}$ chooses $T \overset{\$}{\leftarrow} \mathcal{S}_n$, $y \overset{\$}{\leftarrow} \mathcal{C}$ (where $\mathcal{C}$ is a code generated by G^{pub}), $e' \overset{\$}{\leftarrow} W_2^{n,t}$ (where $W_2^{n,t} = \{x \in \mathbb{F}_2^n | w_H(x) = t\}$), computes $C_2 = com(yT)$, $C_3 = com((y \oplus e')T)$, and sets C_1 to be a random binary vector of appropriate length.
     Let $COM = (C_1|C_2|C_3)$ and $Ans = (yT|e'T)$. Then, $e'T$ has the same probability distribution as eT, moreover for some $z \in \mathcal{C}$,
     $\Pr[(u \oplus m)G^{pub} = z] = \Pr[u = \phi_m^{-1}(\psi^{-1}(z))] = 2^{-k} = \Pr[y = z]$.
   - If $b = 2$, $\mathcal{SIM}$ chooses $y \overset{\$}{\leftarrow} \mathbb{F}_2^k$, $T \overset{\$}{\leftarrow} \mathcal{S}_n$, computes $C_1 = com(T)$, $C_3 = com((yG^{pub} \oplus c)T)$ and sets C_2 to be a random binary vector of appropriate length.
     Let $COM = (C_1|C_2|C_3)$ and $Ans = (y|T)$.
2. $\mathcal{SIM}$ computes $b' = St(COM)$.
3. If $b = b'$, then $\mathcal{SIM}$ writes on the tape $\mathcal{R}$ the quantities H, b, and Ans, otherwise $\mathcal{SIM}$ goes to Step 1.

Thus, in $3s$ rounds on average, the simulator $\mathcal{SIM}$ produces a communication tape $\mathcal{R}$ computationally indistinguishable from a communication tape $\mathcal{R}_{\mathsf{P},\mathsf{V}}$ produced by the honest parties running s rounds of Protocol 1. Therefore, we have that $\langle \mathsf{P}(m,e), \widetilde{\mathsf{V}}\rangle(pk,c)$ and $\langle \mathcal{SIM}, \widetilde{\mathsf{V}}\rangle(pk,c)$ are computationally indistinguishable. Note that computational indistinguishability is due to the fact that the bit commitment scheme is computationally hiding according to Definition 7. This completes the proof of the proposition. □

The above arguments of completeness, soundness and zero-knowledge conclude the proof of the theorem. □

By inspecting the construction of the randomized McEliece PKC in Sec. 2.2, the next Corollary follows immediately by replacing m with $(r|m)$ in Theorem 1.

Corollary 1. *Protocol McEliece PPK is a proof of plaintext knowledge for the Randomized McEliece PKE of [29] assuming hardness of the McEliece problem.*

3.1 Extensions

Similarly to the above construction, PPK for the Niederreiter PKE [28] (the dual of the McEliece PKE), or its semantically secure variant [29], can be constructed in a straight forward manner using Stern's zero-knowledge identification scheme [34] (the dual of Véron's scheme [35]).

We believe but do not prove formally that the result of this section can also be extended to provide PPK for the q-ary variants of the McEliece encryption [22,4] using the identification scheme by Cayrel et al [10], which is based on q-ary codes.

4 Verifiable McEliece Encryption

Let us denote by 0^l an all-zero vector of length $l \in \mathbb{N}$.

In this section, we present the verifiable IND-CPA McEliece encryption for the equality relation $R_{eq} = \{(m, m')|m = m'\}$, i.e. that a given ciphertext c is an encryption of a given plaintext m under public key G^{pub}.

Let the parameters n, k, k_0, k_1, t be as described in Section 2.2, in particular, $k = k_0 + k_1$ and $m \in \mathbb{F}_2^{k_1}$.

Witness: (r, e), where $r \in \mathbb{F}_2^{k_0}$, $e \in \mathbb{F}_2^n$, $w_H(e) = t$.

Common data: $(G^{pub} \in \mathbb{F}_2^{k \times n}, t)$ – the McEliece public key, and $c = (r|m)G^{pub} \oplus e$ – the Randomized McEliece PKE ciphertext (as described in Section 2.2).

Remark 3. For the ciphertext c as defined above, we have that $c \oplus (0^{k_0}|m)G^{pub} = (r|0^{k_1})G^{pub} \oplus e = rG_r^{pub} \oplus e$, where $G_r^{pub} \in \mathbb{F}_2^{k_0 \times n}$ is a restriction of G^{pub} to its first k_0 rows.

Protocol 2 (Verifiable McEliece PKE).

1. P and V execute $\mathsf{PPK}(G^{pub}, c; (r|m), e)$.
   If PPK was rejected, then V rejects.
2. P and V each compute:
   $c_r = c \oplus (0^{k_0}|m)G^{pub} = rG_r^{pub} \oplus e$.
3. P and V execute $\mathsf{PPK}(G_r^{pub}, c_r; r, e)$.
   If PPK was rejected, then V rejects,
   otherwise V accepts.

Proposition 2. *Protocol 2 is a verifiable McEliece encryption for the relation R_{eq} under hardness of G-SD, LPN and GD problems.*

Proof (Sketch). We need to argue completeness, zero-knowledge, and soundness. The first two properties follow easily using the proof of Theorem 1.

As for soundness, Step 1 ensures that c is indeed of the form $(r'|m')G^{pub} \oplus e$ with $w_H(e) = t$ for some $r' \in \mathbb{F}_2^{k_0}$ and $m' \in \mathbb{F}_2^{k_1}$. Now, suppose $m \neq m'$, then we have $c'_r = r'G_r^{pub} \oplus e \oplus (m \oplus m')G_m^{pub}$, where $G_m^{pub} \in \mathbb{F}_2^{k_1 \times n}$ is a restriction of G^{pub} to its last k_1 rows. Note that $(m \oplus m')G_m^{pub}$ is not in a code generated by G_r^{pub}, since the rows of G^{pub} are linearly independent. However, since $(m \oplus m')G_m^{pub}$ is a codeword of the code generated by G^{pub}, its weight is at least $d \geq 2t + 1$. Therefore, the weight of $e \oplus (m \oplus m')G_m^{pub}$ is at least $t + 1$. This implies that if $\widetilde{\mathsf{P}}$ was accepted by V, he necessarily used an error vector of weight larger than t, that would contradict to soundness of Protocol 1 established by Theorem 1.

We note that although the above protocol does not reveal any information on the witness (r, e), the verifier learns the plaintext m and hence she will be able to construct a valid ciphertext of Randomized McEliece encryption with randomness (r, e) for any plaintext. This attack can be prevented using standard message integrity techniques, such as message authentication codes. We leave this issue for our future study.

5 Conclusion

We presented the first proof of plaintext knowledge and the first verifiable encryption for an equality relation for the McEliece PKE. Our constructions are proved secure in the standard model, under hardness of the McEliece assumptions related to coding theory. An important open question is to upgrade our scheme to non-malleable security. According to [23], this will allow us to construct password-based authentication and key exchange, as well as deniable authentication based on coding. Another important open question is to extend our verifiable encryption to more general relations and to verifiable decryption. This would, for instance, yield code-based constructions for key escrow and optimistic fair exchange, according to [9].

Acknowledgments. The authors would like to thank anonymous reviewers of ACISP 2012 for their helpful comments.

The first author was supported by a *kakenhi* Grant-in-Aid for Young Scientists (B) 24700013 from Japan Society for the Promotion of Science.

References

1. Aumann, Y., Rabin, M.O.: A Proof of Plaintext Knowledge Protocol and Applications. Manuscript (June 2001), Available as slides from 1998 IACR Distinguished Lecture by M.O. Rabin, `http://www.iacr.org/publications/dl/rabin98/rabin98slides.ps`
2. Asokan, N., Shoup, V., Waidner, M.: Optimistic Fair Exchange of Digital Signatures (Extended Abstract). In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 591–606. Springer, Heidelberg (1998)
3. Bendlin, R., Damgård, I.: Threshold Decryption and Zero-Knowledge Proofs for Lattice-Based Cryptosystems. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 201–218. Springer, Heidelberg (2010)
4. Bernstein, D.J., Lange, T., Peters, C.: Wild McEliece. In: Biryukov, A., Gong, G., Stinson, D.R. (eds.) SAC 2010. LNCS, vol. 6544, pp. 143–158. Springer, Heidelberg (2011)
5. Berlekamp, E., McEliece, R., van Tilborg, H.: On the inherent intractability of certain coding problems. IEEE Trans. on Inf. Theory 24, 384–386 (1978)
6. Bernstein, D.J.: Grover vs. McEliece. In: Sendrier, N. (ed.) PQCrypto 2010. LNCS, vol. 6061, pp. 73–80. Springer, Heidelberg (2010)
7. Bernstein, D.J., Lange, T., Peters, C.: Smaller Decoding Exponents: Ball-Collision Decoding. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 743–760. Springer, Heidelberg (2011)

8. Camenisch, J., Damgård, I.: Verifiable Encryption, Group Encryption, and Their Applications to Separable Group Signatures and Signature Sharing Schemes. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 331–345. Springer, Heidelberg (2000)
9. Camenisch, J.L., Shoup, V.: Practical Verifiable Encryption and Decryption of Discrete Logarithms. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 126–144. Springer, Heidelberg (2003)
10. Cayrel, P.-L., Véron, P., El Yousfi Alaoui, S.M.: A Zero-Knowledge Identification Scheme Based on the q-ary Syndrome Decoding Problem. In: Biryukov, A., Gong, G., Stinson, D.R. (eds.) SAC 2010. LNCS, vol. 6544, pp. 171–186. Springer, Heidelberg (2011)
11. Dowsley, R., van de Graaf, J., Müller-Quade, J., Nascimento, A.C.A.: Oblivious Transfer Based on the McEliece Assumptions. In: Safavi-Naini, R. (ed.) ICITS 2008. LNCS, vol. 5155, pp. 107–117. Springer, Heidelberg (2008)
12. Engelbert, D., Overbeck, R., Schmidt, A.: A Summary of McEliece-Type Cryptosystems and their Security. Journal of Mathematical Cryptology 1, 151–199 (2007), Walter de Gruyter
13. Finiasz, M., Sendrier, N.: Security Bounds for the Design of Code-Based Cryptosystems. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 88–105. Springer, Heidelberg (2009)
14. Fischer, J.-B., Stern, J.: An Efficient Pseudo-random Generator Provably as Secure as Syndrome Decoding. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 245–255. Springer, Heidelberg (1996)
15. Galil, Z., Haber, S., Yung, M.: Symmetric Public-Key Encryption. In: Williams, H.C. (ed.) CRYPTO 1985. LNCS, vol. 218, pp. 128–137. Springer, Heidelberg (1986)
16. Goldreich, O.: Foundations of Cryptography I: Basic Tools. Cambridge University Press (2001)
17. Goldreich, O.: Foundations of Cryptography II: Basic Applications. Cambridge University Press (2004)
18. Goldreich, O., Micali, S., Wigderson, A.: Proofs that Yield Nothing But Their Validity for All Languages in NP Have Zero-Knowledge Proof Systems. J. ACM 38(3), 691–729 (1991)
19. Goldwasser, S., Kharchenko, D.: Proof of Plaintext Knowledge for the Ajtai-Dwork Cryptosystem. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 529–555. Springer, Heidelberg (2005)
20. Goppa, V.D.: A new class of linear error-correcting code. Probl. Peredach. Inform. 6, 24–30 (1970) (in Russian)
21. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-knowledge from secure multiparty computation. In: STOC, pp. 21–30 (2007)
22. Janwa, H., Moreno, O.: McEliece Public Key Cryptosystems Using Algebraic-Geometric Codes. Des. Codes Cryptography 8(3), 293–307 (1996)
23. Katz, J.: Efficient and Non-Malleable Proofs of Plaintext Knowledge and Applications. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 211–228. Springer, Heidelberg (2003)
24. Kobara, K., Morozov, K., Overbeck, R.: Coding-Based Oblivious Transfer. In: MMICS, pp. 142–156 (2008)
25. MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error-Correcting Codes. North-Holland, Amsterdam (1992)
26. McEliece, R.J.: A Public-Key Cryptosystem Based on Algebraic Coding Theory. Deep Space Network Progress Rep. (1978)

27. Naor, M.: Bit Commitment Using Pseudo-Randomness (Extended Abstract). In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 128–136. Springer, Heidelberg (1990)
28. Niederreiter, H.: Knapsack-type Cryptosystems and Algebraic Coding Theory. Prob. of Control and Inf. Theory 15(2), 159–166 (1986)
29. Nojima, R., Imai, H., Kobara, K., Morozov, K.: Semantic security for the McEliece cryptosystem without random oracles. Des. Codes Cryptography 49(1-3), 289–305 (2008)
30. Roth, R.: Introduction to coding theory. Cambridge University Press (2006)
31. Sendrier, N.: On the security of the McEliece public-key cryptosystem. In: Information, Coding and Mathematics – Proceedings of Workshop honoring Prof. Bob McEliece on his 60th Birthday, pp. 141–163. Kluwer (2002)
32. Shor, P.W.: Polynominal Time Algorithms for Discrete Logarithms and Factoring on a Quantum Computer. In: Huang, M.-D.A., Adleman, L.M. (eds.) ANTS 1994. LNCS, vol. 877, p. 289. Springer, Heidelberg (1994)
33. Stadler, M.: Publicly Verifiable Secret Sharing. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 190–199. Springer, Heidelberg (1996)
34. Stern, J.: A New Identification Scheme Based on Syndrome Decoding. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 13–21. Springer, Heidelberg (1994)
35. Véron, P.: Improved identification schemes based on error-correcting codes. Appl. Algebra Eng. Commun. Comput. 8(1), 57–69 (1996)
36. Xagawa, K., Kawachi, A., Tanaka, K.: Proof of Plaintext Knowledge for the Regev Cryptosystems. Tech.rep. C-236, Tokyo Inst. of Technology (2007)
37. Xagawa, K., Tanaka, K.: Zero-Knowledge Protocols for NTRU: Application to Identification and Proof of Plaintext Knowledge. In: Pieprzyk, J., Zhang, F. (eds.) ProvSec 2009. LNCS, vol. 5848, pp. 198–213. Springer, Heidelberg (2009)

Effort-Release Public-Key Encryption from Cryptographic Puzzles

Jothi Rangasamy, Douglas Stebila, Colin Boyd, Juan Gonzalez-Nieto, and Lakshmi Kuppusamy

Information Security Institute, Queensland University of Technology,
GPO Box 2434, Brisbane, Queensland 4001, Australia
{j.rangasamy,stebila,c.boyd,j.gonzaleznieto,l.kuppusamy}@qut.edu.au

Abstract. Timed-release cryptography addresses the problem of "sending messages into the future": a message is encrypted so that it can only be decrypted after a certain amount of time, either (a) with the help of a trusted third party time server, or (b) after a party performs the required number of sequential operations. We generalise the latter case to what we call *effort-release public key encryption (ER-PKE)*, where only the party holding the private key corresponding to the public key can decrypt, and only after performing a certain amount of computation which may or may not be parallelisable. Effort-release PKE generalises both the sequential-operation-based timed-release encryption of Rivest, Shamir, and Wagner, and also the encapsulated key escrow techniques of Bellare and Goldwasser. We give a generic construction for ER-PKE based on the use of moderately hard computational problems called puzzles. Our approach extends the KEM/DEM framework for public key encryption by introducing a difficulty notion for KEMs which results in effort-release PKE. When the puzzle used in our generic construction is non-parallelisable, we recover timed-release cryptography, with the addition that only the designated receiver (in the PKE setting) can decrypt.

Keywords: puzzles, difficulty, timed-release encryption, key escrow.

1 Introduction

Until 1992, only the *hard* problems of computational complexity were considered as the foundation of cryptography. Dwork and Naor introduced the notion of *moderately hard* problems in 1992 [9]. Since then, moderately hard problems have shown a great deal of promise and have emerged as an important pillar of cryptography. A moderately hard problem is defined as a cryptographic problem such that solving it is not computationally infeasible but also not easy. Moderately hard problems have found their main application in guarding against resource exhaustion attacks such as denial-of-service (DoS) and spam [9, 10]. In these applications, they are called *client puzzles* or *proofs of work*; in the case of DoS attacks, a defending server can force its clients to commit some of its own resources by solving a puzzle, before being granted access to a resource. In this work, we call all such moderately hard problems *cryptographic puzzles* regardless of who is the solver.

Timed-release cryptography. Rivest *et al.* used a class of cryptographic puzzles to enable "future decryption": encrypting a message so that the decryption is possible only after a certain amount of time has elapsed [14]. They called this idea timed-release cryptography (TRC) and also proposed an alternative way to accomplish TRC with the help of a trusted third party time server. The class of cryptographic puzzles being used for this task is called *time-lock puzzles* and has the following properties: (i) solving them is an intrinsically sequential process (puzzle solving is a non-parallellisable task) and (ii) the puzzle generator may hold a trapdoor allowing an easy (short-cut) way to find solutions. Timed-release encryption has been identified to have many applications in practice. Examples given by Rivest *et al.* include e-voting where opening votes needs to be delayed and sealed-bid auctions where the bids must not be opened until the end of the bidding period.

Motivation. Since Rivest *et al.*'s work, there has been a lot of work on TRC but primarily focused on the second approach of using a trusted third-party server [6, 4, 7]. Moreover the puzzle-based TRC of Rivest *et al.* does not provide confidentiality which is as important as delaying the decryption since anyone can get the message after solving the associated puzzle. Achieving both the confidentiality and the delayed decryption properties is vital in many applications where timed-release encryption is useful. For example, bid-privacy in e-auctions and vote-privacy in e-voting schemes require the addition of confidentiality.

Another interesting scenario is the encapsulated key escrow techniques of Bellare and Goldwasser [1, 2] where both the confidentiality and the delayed decryption properties are essential, but the puzzles need not be non-parallelisable. In particular an Internet service provider may need to escrow (session) keys of its customers to the government law-enforcement agency. To prevent the agency from engaging in massive wire-tapping, puzzles are used to delay the key recovery process. However we observe that puzzle-based TRC has not been treated in a formal way, thus a more formal and thorough approach is desirable.

Contributions. We propose the notion of *effort-release PKE* (ER-PKE) in which *only* the intended recipient can get the message, and that too after a certain amount of computational effort which need not be a sequential process. Our notion generalises both the encapsulated key escrow techniques of Bellare and Goldwasser and the puzzle-based timed-release encryption of Rivest, Shamir, and Wagner in the PKE setting.

Moreover our notion generalises timed-release cryptography in two ways as effort-release cryptography considers not only non-parallellisable puzzles but also parallelisable ones and achieves confidentiality. In particular, the receiver can decrypt the message only after solving the puzzle correctly but the solving process may or may not be parallelisable. Since time-lock puzzles are mainly non-parallelisable puzzles, restricting effort-release cryptography only to non-parallelisable puzzles recovers timed-release PKE.

In our approach, we adapt the KEM/DEM approach to obtain a generic construction of effort-release PKE. In particular, following this strictly modular approach, we first introduce a difficulty notion for KEMs and quantify the effort required to release the encrypted message by extending the difficulty notion for puzzles by Chen *et al*[5]. We then give a generic construction of difficult KEM as a composition of a PKE and a difficult puzzle. Finally, we define effort-release PKE analogous to difficult KEM, show that difficulty of the KEM carries over to the KEM/DEM hybrid PKE, and provide a concrete construction of ER-PKE.

Paper Outline. Section 2 considers the definition and security notions for puzzles. Section 3 presents the difficulty notion for KEMs by adapting the framework for puzzle schemes. Section 4 is dedicated to effort-release PKE and effort-release hybrid PKE and Section 5 concludes the paper.

Notation. If n is an integer, $|n|$ denotes its length in bits and if S is a set, $|S|$ denotes its cardinality. $a \overset{\$}{\leftarrow} S$ means choosing a from the set S at random and if $a = (a_1, \ldots, a_n)$ then $(a_1, \ldots, a_n) \leftarrow a$ means a is parsed as shown. By $y \leftarrow A(x)$, we mean that the output of an algorithm A with the input x is assigned to y; $y \overset{\$}{\leftarrow} A(x)$ denotes the similar running of a probabilistic algorithm. PPT means probabilistic polynomial time. $[A(x_1, x_2, \ldots)]$ denotes the set of all possible outputs of A on inputs $x_1, x_2, \ldots$. We use $\text{negl}(\ell)$ to denote an arbitrary function which is negligible as a function of ℓ.

2 Cryptographic Puzzles

The functions that we often use in cryptography are either easy to compute or intractable. In this section we look at a special kind of functions or problems that are moderately hard to compute: cryptographic puzzles.

A cryptographic puzzle scheme CPuz is a tuple $(\mathsf{Setup}, \mathsf{GenPuz}, \mathsf{GetSoln}, \mathsf{FindSoln}, \mathsf{Vrfy})$ of algorithms defined as follows:

$\mathsf{Setup}(1^\ell)$**:** The PPT algorithm that accepts the security parameter ℓ as input and returns output as follows:

- Selects the key space sSpace, the difficulty space QSpace, the string space $\mathsf{strSpace}$, the puzzle instance space $\mathsf{puzSpace}$ and puzzle solution space $\mathsf{solnSpace}$.
- Selects the long-term puzzle secret $s \overset{\$}{\leftarrow} \mathsf{sSpace}$.
- Selects the puzzle parameters $\mathsf{params} \leftarrow (\mathsf{sSpace}, \mathsf{puzSpace}, \mathsf{solnSpace}, \mathsf{QSpace})$ required for the client puzzle.
- Returns (params, s)

$\mathsf{GenPuz}(\mathsf{params}, s, Q, str)$**:** Given params, the puzzle secret $s \in \mathsf{sSpace}$, $Q \in \mathsf{QSpace}$ and $str \in \mathsf{strSpace}$ the probabilistic algorithm outputs a puzzle instance $puz \in \mathsf{puzSpace}$.

$\mathsf{GetSoln}(\mathsf{params}, s, puz)$**:** Given params, the puzzle secret $s \in \mathsf{sSpace}$, and a puzzle $puz \in \mathsf{puzSpace}$, the algorithm outputs a solution $soln \in \mathsf{solnSpace}$.

$\mathsf{FindSoln}(\mathsf{params}, puz, \tau)$**:** Given params, a puzzle $puz \in \mathsf{puzSpace}$ and a run time $\tau \in \mathbb{N}$, the algorithm outputs a potential solution $soln \in \mathsf{solnSpace}$ after running for at most τ clock cycles of execution.

$\mathsf{Vrfy}(\mathsf{params}, puz, soln)$**:** is a deterministic algorithm taking as inputs params, a puzzle $puz \in \mathsf{puzSpace}$ and a potential solution $soln \in \mathsf{puzSpace}$ and returns a true or false.

Correctness. We say a puzzle scheme $\mathsf{CPuz} = (\mathsf{Setup}, \mathsf{GenPuz}, \mathsf{GetSoln}, \mathsf{FindSoln}, \mathsf{Vrfy})$ is correct if for all $(\mathsf{params}, s) \in [\mathsf{Setup}(1^k)]$, all $Q \in \mathsf{QSpace}$, all $str \in \mathsf{strSpace}$ and all $puz \in [\mathsf{GenPuz}(\mathsf{params}, s, Q, str)]$, there exists a $\tau \in \mathbb{N}$ such that $soln \leftarrow \mathsf{FindSoln}(\mathsf{params}, puz, \tau)$, and $\mathsf{true} \leftarrow \mathsf{Vrfy}(\mathsf{params}, puz, soln)$.

Remark 1. The GetSoln algorithm taking the trapdoor puzzle secret s as an input and will be used by the puzzle generator to find a solution faster than the FindSoln algorithm. Hence the difficulty of finding solution applies only to the solver running the FindSoln without the trapdoor s. In this work we are interested in puzzles for which GetSoln algorithms exist and there exist unique solution for each puzzle instance.

Chen *et al.* [5] were the first to study computational puzzles in a rigorous manner and they introduced two necessary security properties for a puzzle scheme to be effective against DoS attackers. In particular they defined two security notions of puzzles, namely, unforgeability and difficulty. The unforgeability property requires that only the puzzle generator who holds the long-term puzzle secret can generate genuine puzzles. The difficulty property requires that solving a puzzle requires a certain amount of computational work.

In the context of DoS defense the unforgeability property is quite important as argued by Chen *et al.*. In contrast, in the context of ER-PKE a sender generates a puzzle and encrypts it under the receiver's public key so that the adversary does not see puzzles. Hence we require puzzles only to be difficult enough for the intended recipient.

We now describe the puzzle-difficulty game of Chen *et al.* using the code-based game-playing approach due to Bellare and Rogaway [3]. The difficulty of CPuz is defined by the game executed between a challenger and an adversary $\mathcal{A}$ in Figure 1. The advantage of $\mathcal{A}$ playing the difficulty game is defined as

$$\mathbf{Adv}^{Q,\mathsf{Diff}}_{\mathcal{A},\mathsf{CPuz}}(\ell) = \Pr\left[\mathbf{Exec}^{Q,\mathsf{Diff}}_{\mathcal{A},\mathsf{CPuz}}(\ell) = 1\right].$$

Definition 1 (Puzzle-difficulty). *Let $\epsilon_{\ell,Q}(\tau)$ be a family of functions monotonically increasing in τ, where ℓ is a security parameter and Q is a difficulty parameter. Fix $\ell \geq 0$ and $Q \geq 0$. Then, a client puzzle* CPuz *is $\epsilon_{\ell,Q}(\cdot)$-difficult if*

$$\mathbf{Adv}^{Q,\mathsf{Diff}}_{\mathcal{A},\mathsf{CPuz}}(\ell) \leq \epsilon_{\ell,Q}(\tau),$$

for all $\mathcal{A}$ running in time at most τ ($\tau \in \mathbb{N}$).

```
$\mathbf{Exec}^{Q,\mathsf{Diff}}_{\mathcal{A},\mathsf{CPuz}}(\ell)$ :

1. $(\mathsf{params}, s) \xleftarrow{\$} \mathsf{Setup}(1^\ell); List \leftarrow \emptyset$
2. $(state, str^*) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}}(\mathsf{params})$
   • If $\mathcal{A}$ queries $\mathcal{O}(str)$:
     (a) $puz \xleftarrow{\$} \mathsf{GenPuz}(\mathsf{params}, s, str)$
     (b) $soln \leftarrow \mathsf{GetSoln}(\mathsf{params}, s, str, puz)$
     (c) Append $(str, puz, soln)$ to $List$
     (d) Answer $\mathcal{A}$ with $(puz, soln)$.
3. $puz^* \xleftarrow{\$} \mathsf{GenPuz}(\mathsf{params}, s, str^*)$
4. $soln^* \leftarrow \mathsf{GetSoln}(\mathsf{params}, s, str^*, puz^*)$
5. $soln' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}}(state, puz^*)$
   • Answer $\mathcal{O}$ queries as above
6. If $(str^*, puz^*, soln')$ is in the $List$, return $\perp$
7. Return 1 if $soln' = soln^*$, else return 0.
```

Fig. 1. Difficulty experiment for puzzles

Remark 2. Let Puz be an $\epsilon_{\ell,Q}(t)$-difficult puzzle such that each instance of it requires about Q basic steps to solve. Then $\epsilon_{\ell,Q}(t)$ might take the form $t/Q + \text{negl}(\ell)$. However for puzzles we usually have $\ell \geq Q$ and ℓ can be chosen according to the difficulty we aim to achieve. When $\ell \geq Q$, we have that $\epsilon_Q(t) := t/Q + \text{negl}(Q) \geq t/Q + \text{negl}(\ell) = \epsilon_{\ell,Q}(t)$ and therefore an $\epsilon_{\ell,Q}(t)$-difficult puzzle Puz is also $\epsilon_Q(t)$-difficult. Hence, for ease of notation, we set the difficulty parameter Q to be the puzzle security parameter.

In the sections that follow we combine difficult puzzles with public-key primitives to delay the decryption process for a certain amount of time.

3 Difficult Key Encapsulation Mechanism

The basic idea behind the work of Rivest *et al.* on TRC is that the symmetric key used for encrypting the message should not be available immediately for the recipient but only after a certain period of time. Analogous to this approach, the natural extension to a designated solver case is to delay the decryption of the ciphertext encapsulating the symmetric key. This leads us to seek a new class of KEMs and we call a KEM satisfying this goal a *difficult* KEM.

3.1 Definition: Difficult KEM

We now propose the notion of difficult key encapsulation mechanism (DKEM) which will lead to a PKE achieving confidentiality as well as delayed decryption. A DKEM works very similar to a KEM scheme, except that the encapsulation algorithm takes in addition to the regular inputs a secret generated by the parameter generation algorithm. A DKEM is a tuple (KEM.PG,KEM.KG,KEM.Encap, KEM.Decap) of 4 algorithms with the following input/output behavior:

$$\begin{aligned}(\mathsf{params}, s) &\xleftarrow{\$} \mathsf{KEM.PG}(1^k, 1^Q)\\ (\mathsf{pk}, \mathsf{sk}) &\xleftarrow{\$} \mathsf{KEM.KG}(\mathsf{params})\\ (K, C) &\xleftarrow{\$} \mathsf{KEM.Encap}(\mathsf{params}, s, \mathsf{pk})\\ K &\leftarrow \mathsf{KEM.Decap}(\mathsf{params}, \mathsf{sk}, C).\end{aligned}$$

CORRECTNESS. Correctness of $\mathsf{DKEM} = (\mathsf{KEM.PG}, \mathsf{KEM.KG}, \mathsf{KEM.Encap}, \mathsf{KEM.Decap})$ requires that, for all $(\mathsf{params}, s) \in [\mathsf{KEM.PG}(1^k, 1^Q)]$, and all $(\mathsf{pk}, \mathsf{sk}) \in [\mathsf{KG}(\mathsf{params})]$, we have $\mathsf{KEM.Decap}(\mathsf{params}, \mathsf{sk}, C) = K$ for all $(C, K) \leftarrow \mathsf{KEM.Encap}(\mathsf{params}, s, \mathsf{pk})$ with probability one, where the probability is taken over the coins of $\mathsf{KEM.Encap}$.

DIFFICULTY. We formally define what we mean by a KEM be difficult. We observed that the difficulty in getting session keys is analogous to the difficulty of getting puzzle solutions and thus we extend the puzzle-difficulty property of Chen *et al.* to KEMs as a special security property.

The difficulty of KEM is defined by the game executed between a challenger and an adversary $\mathcal{A}$ in Figure 4. The advantage of $\mathcal{A}$ playing the difficulty game is defined as

$$\mathbf{Adv}^{k,\mathsf{Diff}}_{\mathcal{A},\mathsf{KEM}}(Q) = \Pr\left[\mathbf{Exec}^{k,\mathsf{Diff}}_{\mathcal{A},\mathsf{KEM}}(Q) = 1\right].$$

```
$\mathbf{Exec}^{k,\mathsf{Diff}}_{\mathcal{A},\mathsf{KEM}}(Q)$ :

1. $(\mathsf{params}, s) \xleftarrow{\$} \mathsf{KEM.PG}(1^k, 1^Q); KList \leftarrow \emptyset$
2. $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KEM.KG}(\mathsf{params})$
3. $state \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{Enc}}}(\mathsf{params}, \mathsf{sk})$
     • If $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{Enc}}$:
         (a) $(C, K) \xleftarrow{\$} \mathsf{KEM.Encap}(\mathsf{params}, s, \mathsf{pk})$
         (b) Append $(C, K)$ to $KList$
         (c) Answer $\mathcal{A}$ with $(C, K)$.
4. $(C^*, K^*) \xleftarrow{\$} \mathsf{KEM.Encap}(\mathsf{params}, s, \mathsf{pk})$
5. $K' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}_{\mathsf{Enc}}}(state, C^*)$
     • Answer $\mathcal{O}_{\mathsf{Enc}}$ queries as above
6. If $(C^*, K^*)$ is in the $KList$, return $\perp$
7. Return 1 if $K' = K^*$, else return 0.
```

Fig. 2. Difficulty experiment for key encapsulation mechanism

Definition 2 (KEM-difficulty). *Let $\epsilon_{k,Q}(\tau)$ be a family of functions monotonically increasing in τ, where k is a security parameter and Q is a difficulty parameter. Fix $k \geq 0$ and $Q \geq 0$. Then, a KEM is $\epsilon_{k,Q}(\cdot)$-difficult if*

$$\mathbf{Adv}^{k,\mathsf{Diff}}_{\mathcal{A},\mathsf{KEM}}(Q) \leq \epsilon_{k,Q}(\tau),$$

for all $\mathcal{A}$ running in time at most τ ($\tau \in \mathbb{N}$).

Remark 3. The difficulty definition of KEM appears quite similar to the non-invertibility under adaptive chosen ciphertext attacks of KEM but differs in the following ways: the encapsulation here takes a secret as an input which is not an input to the decapsulation algorithm, and (hence) the adversary is given access to the encapsulation oracle but not to the decapsulation oracle.

3.2 A Difficult Key Encapsulation Mechanism from Puzzles

As discussed before, cryptographic puzzles have been predominantly used for fighting against resource exhaustion attacks such as junk email (also known as spam) and DoS attacks [9, 10]. Rivest, Shamir and Wagner [14] were the first to combine puzzles with symmetric-key encryption to intentionally delay the message recovery process. In particular, a sender encrypts a message under a (random) symmetric-key. Then the sender generates a puzzle and uses its unique solution to mask the symmetric-key. The ciphertext and the puzzle is sent to the recipient who gets the message after solving the puzzle.

However this technique just delays the decryption process but anyone that is willing to solve the puzzle can acquire the message and hence confidentiality is lost. One natural way to achieve confidentiality is to encrypt the puzzle under the recipient's public key so that anyone else but the recipient cannot see the puzzle. Following this idea, we instantiate a difficult KEM by using a PKE with a difficult puzzle and we then show that the difficulty of the puzzle substantiates the difficulty of the KEM.

The other approach for this idea could be the following: first generate a KEM ciphertext using the recipient's public key, then mask the ciphertext with the solution of a puzzle and send the masked ciphertext and the puzzle to the recipient. Although this approach of adding puzzles separately to the KEM ciphertext looks interesting it may work only for the specific KEM and puzzle schemes. Thus we follow the direct approach of using the puzzle solution to derive a session key and then encrypting the puzzle as it leads to a generic construction of a difficult KEM using a PKE and a difficult puzzle. Moreover, our approach is compatible with practical PKEs such as RSA-REACT as seen in Section 4.2.

The Scheme. Let KDF be a key derivation function [15, 16]. Let ($\mathsf{PKE.KG}$, $\mathsf{PKE.Enc}$, $\mathsf{PKE.Dec}$) be a PKE scheme and let $\mathsf{CPuz} = (\mathsf{Setup}, \mathsf{GenPuz}, \mathsf{GetSoln}, \mathsf{FindSoln}, \mathsf{Vrfy})$ be a puzzle scheme. Then the proposed DKEM is a tuple ($\mathsf{KEM.PG}$, $\mathsf{KEM.KG}$, $\mathsf{KEM.Encap}$, $\mathsf{KEM.Decap}$) of algorithms as seen in Figure 3:

Security analysis. We consider two security properties for the KEM in Figure 3, namely difficulty (see definition 2) and indistinguishability under adaptive chosen-ciphertext attacks ($\mathsf{IND\text{-}CCA}$) [8, 15].

We now show that if CPuz is difficult in the sense of Chen *et al.*, then the DKEM in Figure 3 is difficult according to the definition 2.

```
KEM.PG(1^k, 1^Q) :
  • (params, s) ←$ Setup(1^Q)
  • Return (params, s)

KEM.Encap(params, s, pk) :
  • puz ←$ GenPuz(params, s, str)
  • soln ← GetSoln(params, s, puz)
  • r ←$ {0,1}^{poly(k)}; K ← KDF(soln, r)
  • C ←$ PKE.Enc(pk, puz||r)
  • Return (C, K)

KEM.KG(params) :
  • (pk, sk) ←$ PKE.KG(params)
  • Return (pk, sk)

KEM.Decap(params, sk, C) :
  • puz||r ← PKE.Dec(sk, C)
  • soln ← FindSoln(params, puz)
  • K ← KDF(soln, r)
  • Return K
```

Fig. 3. DKEM from PKE and puzzles

Theorem 1. *Assume that* KDF *is a random oracle. Let* DKEM = (KEM.PG, KEM.KG, KEM.Encap, KEM.Decap) *be the KEM scheme in Figure 3 and let* CPuz = (Setup, GenPuz, GetSoln, FindSoln, Vrfy) *be a difficult cryptographic puzzle scheme according to the definition 1. Suppose there exists an adversary $\mathcal{A}$ against the difficulty of* DKEM*, then there is an adversary $\mathcal{B}$ against the difficulty of* CPuz *such that*

$$\mathbf{Adv}^{k,\mathsf{Diff}}_{\mathcal{A},\mathsf{DKEM}}(Q) \leq \mathbf{Adv}^{\mathsf{Diff}}_{\mathcal{B},\mathsf{CPuz}}(Q) + \mathrm{negl}(k)$$

and the running time of $\mathcal{B}$ is asymptotically the same that of $\mathcal{A}$.

Proof. Let $\mathcal{A}$ be the attacker against the difficulty of DKEM that makes at most q_{Enc} queries to $\mathcal{O}_{\mathsf{Enc}}$ and at most q_{KDF} queries to the random oracle KDF. We now build an attacker $\mathcal{B}$ that breaks the difficulty of CPuz using $\mathcal{A}$ and runs in asymptotically the same time as $\mathcal{A}$.

$\mathcal{B}$ interacts individually with the challenger in puzzle-difficulty game and $\mathcal{A}$ playing the KEM-difficulty game. We describe how $\mathcal{B}$ proceeds. $\mathcal{B}$'s input are the public parameters params from the puzzle challenger. Now $\mathcal{B}$ generates $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KEM.KG}(\mathsf{params})$ and invokes $\mathcal{A}$ with $(\mathsf{params}, \mathsf{sk})$.

Encapsulation queries. Now $\mathcal{A}$ may issue a polynomial number of Enc queries for which $\mathcal{B}$ answers as follows: for each of $\mathcal{A}$'s query to $\mathcal{O}_{\mathsf{Enc}}$, $\mathcal{B}$ first selects a string $str \xleftarrow{\$} \mathsf{strSpace}$ and queries $\mathcal{O}$ (the oracle for CreatePuzSoln queries) from the puzzle challenger and in response receives a puzzle-solution pair $(puz, soln)$. Then $\mathcal{B}$ queries the random oracle KDF with $(soln, r)$ for $r \xleftarrow{\$} \{0,1\}^{\mathrm{poly}(k)}$, to get K, computes $C \xleftarrow{\$} \mathsf{PKE.Enc}(\mathsf{pk}, puz||r)$ and responds $\mathcal{A}$ with (C, K). $\mathcal{B}$ records $(soln, r, K)$ into the list it maintains for KDF queries. $\mathcal{B}$ also records $\mathcal{A}$'s queries to the KDF oracle.

Challenge. At some point of time, $\mathcal{A}$ asks for the target ciphertext and now $\mathcal{B}$ selects a random string $str^* \xleftarrow{\$} \mathsf{strSpace}$ and queries the puzzle challenger with str^* for the target puzzle. In response $\mathcal{B}$ receives the target puzzle puz^* to solve and $\mathcal{B}$ now computes the challenge ciphertext $C^* \xleftarrow{\$} \mathsf{PKE.Enc}(\mathsf{pk}, puz^*||r^*)$ and responds $\mathcal{A}$ with C^*.

At some time, when $\mathcal{A}$ returns a key K' as the output of the game, $\mathcal{B}$ checks with the list if there was an query to KDF having K' as its output. Since $\mathcal{A}$ has non-negligible advantage in breaking the KEM-difficulty it must have queried KDF with some $soln'$ and r^* to obtain K'. In this case $\mathcal{B}$ searches for the tuple matching $(soln', r^*, K')$ and returns $soln'$ to the puzzle challenger playing the difficulty game.

If $\mathcal{A}$ wins the KEM-difficulty game by guessing the session key K' it happens with probability 1/(the size of key space), which is negl(k). If $\mathcal{A}$ does not guess the session key but wins the KEM-difficulty game then $\mathcal{B}$ also wins the puzzle-difficulty game. Therefore we have,

$$\mathbf{Adv}^{k,\mathsf{Diff}}_{\mathcal{A},\mathsf{KEM}}(Q) \leq \mathbf{Adv}^{\mathsf{Diff}}_{\mathcal{B},\mathsf{CPuz}}(Q) + \mathrm{negl}(k).$$

The running time of $\mathcal{B}$ is asymptotically the same that of $\mathcal{A}$. □

The following theorem says that if PKE is $\mathsf{IND\text{-}CCA}$-secure, then DKEM in Figure 3 is $\mathsf{IND\text{-}CCA}$-secure. The proof of the theorem is similar to the proof for KEM/DEM by Cramer and Shoup[8] and is omitted due to lack of space.

Theorem 2. *Assume that* KDF *is entropy smoothing key derivation function [15, 16]. Let* $\mathsf{PKE} = (\mathsf{PKE.PG}, \mathsf{PKE.KG}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *be an* $\mathsf{IND\text{-}CCA}$*-secure public-key encryption scheme and let* DKEM *be a KEM as seen in Figure 3. Suppose there exists an* $\mathsf{IND\text{-}CCA}$ *adversary* $\mathcal{A}$ *against* DKEM*, then there exist an adversary* $\mathcal{A}_1$ *against the entropy smoothness of* KDF *and an* $\mathsf{IND\text{-}CCA}$ *adversary* $\mathcal{A}_2$ *against* PKE *such that*

$$\mathbf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{A},\mathsf{DKEM}}(k) \leq \mathbf{Adv}^{\mathsf{ES}}_{\mathcal{A}_1,KDF}(k) + \mathbf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{A}_2,\mathsf{PKE}}(k), \quad \forall k \geq 0$$

where $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *have (asymptotically) the same running time as* $\mathcal{A}$*.*

Remark 4. The proof of Theorem 1 is in the random oracle model where as the proof of Theorem 2 is in the standard model assuming KDF to be entropy smoothing(ES) [15, 16]. It is an interesting open problem to construct a DKEM from a difficult puzzle such that the proof of Theorem 1 is in the standard model.

4 Effort-Release Public Key Encryption

In this section we define difficulty for public-key encryption schemes in analogous to the difficulty for KEMs. We call a PKE scheme having this property an *Effort-Release Public Key Encryption* (ER-PKE). An ER-PKE works similar to a PKE but the recipient holding the decryption key cannot immediately and suddenly complete the decryption process but after the required number of operations. That is, the decryption process requires a certain amount of moderately-hard computation which may or may not be a parallelisable task.

An ER-PKE is a tuple ($\mathsf{PKE.PG}$,$\mathsf{PKE.KG}$, $\mathsf{PKE.Enc}$, $\mathsf{PKE.Dec}$) of 4 algorithms with the following input/output behaviour:

$$\begin{aligned}(\mathsf{params}, s) &\xleftarrow{\$} \mathsf{PKE.PG}(1^k, 1^Q)\\ (\mathsf{pk}, \mathsf{sk}) &\xleftarrow{\$} \mathsf{PKE.KG}(\mathsf{params})\\ C &\xleftarrow{\$} \mathsf{PKE.Enc}(\mathsf{params}, s, \mathsf{pk}, m)\\ m &\leftarrow \mathsf{PKE.Dec}(\mathsf{params}, \mathsf{sk}, C).\end{aligned}$$

Correctness. A ER-PKE scheme ($\mathsf{PKE.PG}$,$\mathsf{PKE.KG}$,$\mathsf{PKE.Enc}$, $\mathsf{PKE.Dec}$) is *correct* if, for all $(\mathsf{params}, s) \in [\mathsf{PKE.PG}(1^k)]$, all $(\mathsf{pk}, \mathsf{sk}) \in [\mathsf{PKE.KG}(\mathsf{params})]$, and all plaintexts $m \in \mathsf{MsgSp}$, we have $\mathsf{PKE.Dec}(\mathsf{params}, \mathsf{sk}, \mathsf{Enc}(\mathsf{params}, s, \mathsf{pk}, m)) = m$ with probability one, where the probability is taken over the coins of $\mathsf{PKE.Enc}$.

Informally we call a PKE effort-release if the decryption algorithm cannot be run trivially and the adversary should put in some computational effort for prespecified expected amount of time for the successful decryption of a ciphertext.

We now formally define what do we mean by effort-release PKE. As for the difficulty of KEM, the effort-release game is defined as the difficulty game executed between a challenger and an adversary $\mathcal{A}$ in Figure 4. The advantage of $\mathcal{A}$ playing the difficulty game is defined as

$$\mathbf{Adv}^{k,\mathsf{Diff}}_{\mathcal{A},\mathsf{PKE}}(Q) = \Pr\left[\mathbf{Exec}^{k,\mathsf{Diff}}_{\mathcal{A},\mathsf{PKE}}(Q) = 1\right].$$

```
$\mathbf{Exec}^{k,\mathsf{Diff}}_{\mathcal{A},\mathsf{PKE}}(Q)$ :

1. $(\mathsf{params}, s) \xleftarrow{\$} \mathsf{PKE.PG}(1^k, 1^Q)$; $List \leftarrow \emptyset$
2. $(\mathsf{pk},\mathsf{sk}) \xleftarrow{\$} \mathsf{PKE.KG}(\mathsf{params})$
3. $state \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{Enc}}}(\mathsf{params}, \mathsf{sk})$
     • If $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{Enc}}(m)$:
       (a) $C \xleftarrow{\$} \mathsf{PKE.Enc}(\mathsf{params}, s, \mathsf{pk}, m)$
       (b) Append $(C, m)$ to $List$
       (c) Answer $\mathcal{A}$ with $(C, m)$.
4. $m^* \xleftarrow{\$} \mathsf{MsgSp}$, the message space
5. $C^* \xleftarrow{\$} \mathsf{PKE.Enc}(\mathsf{params}, s, \mathsf{pk}, m^*)$
6. $m' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}_{\mathsf{Enc}}}(state, C^*)$
     • Answer $\mathcal{O}_{\mathsf{Enc}}$ queries as above
7. If $(C^*, m^*)$ is in the $List$, return $\perp$
8. Return 1 if $m' = m^*$, else return 0.
```

Fig. 4. Difficulty experiment for Effort-Release PKE

Definition 3 (Effort-Release PKE). *Let $\epsilon_{k,Q}(\tau)$ be a family of functions monotonically increasing in τ, where k is a security parameter and Q is a difficulty parameter. Fix $k \geq 0$ and $Q \geq 0$. Then, a PKE is $\epsilon_{k,Q}(\cdot)$-effort-release if*

$$\mathbf{Adv}^{k,\mathsf{Diff}}_{\mathcal{A},\mathsf{PKE}}(Q) \leq \epsilon_{k,Q}(\tau),$$

for all $\mathcal{A}$ running in time at most τ ($\tau \in \mathbb{N}$).

4.1 Effort-Release Hybrid PKE

Effort-release hybrid PKE can be seen as an extension of Rivest *et al.*'s timed-release (symmetric-key) encryption to the PKE setting. In this section, we give definition for an effort-release hybrid PKE, which works very similar to a hybrid PKE proposed by Cramer and Shoup [8, 15] and we then prove that (i) the hybrid PKE scheme in Figure 5 is an effort-release PKE if the DKEM is difficult according to the definition 2 and (ii) the effort-release hybrid PKE is IND-CCA-secure if both the DKEM and DEM are IND-CCA-secure.

```
$\mathsf{ER\text{-}PKE.PG}(1^k, 1^Q)$ :
    • $(\mathsf{params}, s) \overset{\$}{\leftarrow} \mathsf{KEM.PG}(1^k, 1^Q)$
    • Return $(\mathsf{params}, s)$

$\mathsf{ER\text{-}PKE.KG}(\mathsf{params})$ :
    • $(\mathsf{pk}, \mathsf{sk}) \overset{\$}{\leftarrow} \mathsf{KEM.KG}(\mathsf{params})$
    • Return $(\mathsf{pk}, \mathsf{sk})$

$\mathsf{ER\text{-}PKE.Enc}(\mathsf{params}, s, \mathsf{pk}, m)$ :
    • $(K, c_1) \leftarrow \mathsf{KEM.Encap}(\mathsf{params}, s, \mathsf{pk})$
    • $K' \leftarrow \mathsf{KDF}(K)$
    • $c_2 \leftarrow \mathsf{DEM.Enc}(K', m)$
    • $C \leftarrow (c_1, c_2)$
    • Return $C$

$\mathsf{ER\text{-}PKE.Dec}(\mathsf{params}, \mathsf{sk}, C)$ :
    • $(c_1, c_2) \leftarrow C$
    • $K \leftarrow \mathsf{KEM.Decap}(\mathsf{params}, \mathsf{sk}, c_1)$
    • $K' \leftarrow \mathsf{KDF}(K)$
    • $m \leftarrow \mathsf{DEM.Dec}(K', c_2)$
    • Return $m$
```

Fig. 5. Effort-Release hybrid PKE

Remark 5. In the Effort-Release hybrid PKE in Figure 5, the operation $K' \leftarrow \mathsf{KDF}(K)$ may look redundant and undesirable since K itself is usually the output of KDF. As shown in Theorem 3, having $K' \leftarrow \mathsf{KDF}(K)$ allows us to prove that the difficulty of DKEM implies the difficulty of the hybrid ER-PKE.

Theorem 3. *Assume that* KDF *is a random oracle. Let* $\mathsf{DKEM} = (\mathsf{KEM.PG}, \mathsf{KEM.KG}, \mathsf{KEM.Encap}, \mathsf{KEM.Decap})$ *be a difficult KEM scheme according to the definition 2 and let* $\mathsf{ER\text{-}PKE}$ *be the scheme in Figure 5. Suppose there exists an adversary* $\mathcal{A}$ *against the difficulty of* $\mathsf{ER\text{-}PKE}$, *then there is an adversary* $\mathcal{B}$ *against the difficulty of* DKEM *such that*

$$\mathbf{Adv}^{k,\mathsf{Diff}}_{\mathcal{A},\mathsf{ER\text{-}PKE}}(Q) \leq (1/q_{\mathsf{KDF}})\mathbf{Adv}^{k,\mathsf{Diff}}_{\mathcal{B},\mathsf{DKEM}}(Q),$$

where q_{KDF} *is an upper bound on the number of queries to* KDF *made by* $\mathcal{A}$ *and the running time of* $\mathcal{B}$ *is asymptotically the same that of* $\mathcal{A}$.

The proof to this theorem is similar to the proof of Theorem 1 and is omitted due to lack of space.

Now the following theorem shows that if both the Difficult-KEM and the DEM are IND-CCA-secure then so is the the Hybrid PKE scheme in Figure 5.

Theorem 4 (Difficult-KEM/DEM Composition Theorem). *Let (*$\mathsf{KEM.PG}, \mathsf{KEM.KG}, \mathsf{KEM.Encap}, \mathsf{KEM.Decap}$*) be an* $\mathsf{IND\text{-}CCA}$*-secure* DKEM*, let* ($\mathsf{DEM.Enc}, \mathsf{DEM.Dec}$) *be an* $\mathsf{IND\text{-}CCA}$*-secure* DEM*, and let (*$\mathsf{PKE.PG}$,$\mathsf{PKE.KG}$,$\mathsf{PKE.Enc}$, $\mathsf{PKE.Dec}$*) be the resulting hybrid ER-PKE scheme. Then, for any* $\mathsf{IND\text{-}CCA}$ *adversary* $\mathcal{A}$ *against* $\mathsf{ER\text{-}PKE}$*, there exists an* $\mathsf{IND\text{-}CCA}$ *adversary* $\mathcal{B}_1$ *against* DKEM *and an* $\mathsf{IND\text{-}CCA}$ *adversary* $\mathcal{B}_2$ *against* DEM *such that*

$$\mathbf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{A},\mathsf{ER\text{-}PKE}}(k) \leq \mathbf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{B}_1,\mathsf{DKEM}}(k) + \mathbf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{B}_2,\mathsf{DEM}}(k) \quad \forall k \geq 0$$

and $\mathcal{B}_1$ *and* $\mathcal{B}_2$ *have (asymptotically) the same running time as* $\mathcal{A}$*.*

The proof to this theorem is almost identical to the proof for KEM/DEM by Cramer and Shoup [8] and is omitted.

4.2 Constructions of Effort-Release Hybrid PKE

Theorem 3 states that if the underlying KEM is difficult then the resulting hybrid encryption scheme is effort-release PKE. Therefore, as shown in Section 3.2 we can easily instantiate an effort-release hybrid PKE by constructing a difficult KEM from difficult puzzles in [5, 12, 17] and combining it with a DEM.

Constructions of Timed-Release PKE. As shown by Rivest *et al.*[14] timed-release encryption can be obtained from non-parallelisable puzzles in [13, 14]. Therefore using any of these two puzzles in a generic DKEM from Section 3.2 yields a timed-release PKE.

Effort-Release RSA-REACT. To get an idea of how a practical effort-release PKE might look, we briefly describe a way of constructing ER-PKE from RSA-REACT proposed by Okamoto and Pointcheval [11]. In particular we instantiate Chen *et al.*'s generic puzzle with $\mathsf{SHA1}$ hash function and combine it with KEM part of RSA-REACT and use AES to implement DEM part of RSA-REACT. The resulting ER-RSA-REACT works as follows:

Let G and H be two hash functions with appropriate domains.

The key generation algorithm $\mathsf{KG}(1^k)$**.** On input of the security parameter k, the probabilistic algorithm generates an RSA modulus n and outputs a public-key (e, n) and a secret-key $(d, \phi(n))$ such that $d = e^{-1} \mod \phi(n)$.

The encryption algorithm $\mathsf{Enc}((e, n), m)$**.** Given a message m and a public key (e, n) the PPT algorithm first picks $r \overset{\$}{\leftarrow} \mathbb{Z}_n$ and $u \overset{\$}{\leftarrow} \{0,1\}^{\mathrm{poly}(Q)}$. Now it computes $v \leftarrow \mathsf{SHA1}(u)$ and parses u into u_1 and u_2 such that u_2 is of length Q in bits. Then the algorithm computes $K \leftarrow G(u_2, r)$ and produces a ciphertext (x, c, h) of m, where $x \leftarrow (u_1||v||r)^e \mod n$, $c \leftarrow \mathsf{AES}_K(m)$ and the checking value $h = H(r, m, x, c)$. Output is the ciphertext $C \leftarrow (x, c, h)$.

The decryption algorithm $\mathsf{Dec}((d, \phi(n)), C)$**.** The algorithm first decrypts x using the secret key to obtain $u_1||v||r = x^d \mod n$. Now it starts solving the puzzle (u_1, v) by guessing u_2 such that $v \overset{?}{=} \mathsf{SHA1}(u_1||u_2)$. Then recovers the session key $K = G(u_2, r)$ and the plaintext $m = \mathsf{AES}_K(c)$. Finally checks if $h \overset{?}{=} H(r, m, x, c)$. If the check fails, outputs $\perp$ to indicate rejection, otherwise outputs the message m.

Remark 6. In ER-RSA-REACT a recipient may securely outsource the puzzle solving process; if this is not desirable in some applications then including the key material r along with u as input to $\mathsf{SHA1}$ for a puzzle generation prevents such outsourcing by the recipient since the (stand-in) puzzle solver will then have enough information to recover the message.

5 Conclusion

Timed-release cryptography has been gaining increased popularity due to its many interesting applications. While already known schemes for this purpose are mainly based on time-servers, the only alternative way appears to be puzzle-based ones where the receiver will be able to decrypt the message after solving a puzzle. To the best of our knowledge, the puzzle-based approach has been treated in an ad hoc fashion. We have proposed the notion of *effort-release PKE* which generalises both the encapsulated key escrow techniques of Bellare and Goldwasser and the puzzle-based timed-release encryption of Rivest, Shamir, and Wagner in the PKE setting. We also gave a generic construction of effort-release PKE by adapting the KEM/DEM approach which is tailored to moderately-hard puzzles and the type of puzzle being used decides whether the obtained ER-PKE is timed-release or not.

However, our generic construction of a difficult KEM has a proof of difficulty in the random oracle model and hence it is an open problem to construct a difficult KEM (from puzzles) having the proof of difficulty in the standard model.

Acknowledgements. The authors are grateful to anonymous referees for their comments. This work is supported by Australia-India Strategic Research Fund project TA020002.

References

[1] Bellare, M., Goldwasser, S.: Encapsulated key escrow. Technical Report 688, MIT Laboratory for Computer Science (April 1996), `http://cseweb.ucsd.edu/~mihir/papers/escrow.html`

[2] Bellare, M., Goldwasser, S.: Verifiable partial key escrow. In: Graveman, R., Janson, P.A., Neumann, C., Gong, L. (eds.) ACM CCS, pp. 78–91. ACM (1997)

[3] Bellare, M., Rogaway, P.: The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 409–426. Springer, Heidelberg (2006)

[4] Chalkias, K., Hristu-Varsakelis, D., Stephanides, G.: Improved Anonymous Timed-Release Encryption. In: Biskup, J., López, J. (eds.) ESORICS 2007. LNCS, vol. 4734, pp. 311–326. Springer, Heidelberg (2007)
[5] Chen, L., Morrissey, P., Smart, N.P., Warinschi, B.: Security Notions and Generic Constructions for Client Puzzles. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 505–523. Springer, Heidelberg (2009)
[6] Cheon, J.H., Hopper, N., Kim, Y., Osipkov, I.: Provably secure timed-release public key encryption. ACM Trans. Inf. Syst. Secur. 11, 4:1–4:44 (2008)
[7] Chow, S.S.M., Yiu, S.M.: Timed-Release Encryption Revisited. In: Baek, J., Bao, F., Chen, K., Lai, X. (eds.) ProvSec 2008. LNCS, vol. 5324, pp. 38–51. Springer, Heidelberg (2008)
[8] Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM Journal on Computing 33(1), 167–226 (2003)
[9] Dwork, C., Naor, M.: Pricing via Processing or Combatting Junk Mail. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 139–147. Springer, Heidelberg (1993)
[10] Juels, A., Brainard, J.: Client puzzles: A cryptographic countermeasure against connection depletion attacks. In: Proc. Network and Distributed System Security Symposium (NDSS) 1999, pp. 151–165. Internet Society (1999)
[11] Okamoto, T., Pointcheval, D.: REACT: Rapid Enhanced-Security Asymmetric Cryptosystem Transform. In: Naccache, D. (ed.) CT-RSA 2001. LNCS, vol. 2020, pp. 159–175. Springer, Heidelberg (2001)
[12] Rangasamy, J., Stebila, D., Boyd, C., Gonzalez Nieto, J.: An integrated approach to cryptographic mitigation of denial-of-service attacks. In: Sandhu, R., Wong, D.S. (eds.) Proc. 6th ACM Symposium on Information, Computer and Communications Security (ASIACCS) 2011, pp. 114–123. ACM (2011), http://eprints.qut.edu.au/41285/
[13] Rangasamy, J., Stebila, D., Boyd, C., Gonzalez Nieto, J., Kuppusamy, L.: Efficient modular exponentiation-based puzzles for denial-of-service protection. In: Proc. International Conference on Information Security and Cryptology (ICISC 2011). LNCS, Springer, Heidelberg (2011) (to appear), http://eprints.qut.edu.au/47894/
[14] Rivest, R.L., Shamir, A., Wagner, D.A.: Time-lock puzzles and timed-release crypto. Technical Report TR-684, MIT Laboratory for Computer Science (March 1996), http://people.csail.mit.edu/rivest/RivestShamirWagner-timelock.pdf
[15] Shoup, V.: A proposal for an ISO standard for public key encryption (version 2.1). manuscript (2001), http://shoup.net/papers
[16] Shoup, V.: Sequences of games: a tool for taming complexity in security proofs. Technical report (2004), http://eprint.iacr.org/2004/332
[17] Stebila, D., Kuppusamy, L., Rangasamy, J., Boyd, C., Gonzalez Nieto, J.: Stronger Difficulty Notions for Client Puzzles and Denial-of-Service-Resistant Protocols. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 284–301. Springer, Heidelberg (2011), http://eprints.qut.edu.au/40036/

Leakage-Resilience of Stateless/Stateful Public-Key Encryption from Hash Proofs*

Manh Ha Nguyen[1], Keisuke Tanaka[1], and Kenji Yasunaga[2]

[1] Tokyo Institute of Technology, Japan
[2] Institute of Systems, Information Technologies and Nanotechnologies (ISIT), Japan

Abstract. We consider the problem of constructing public-key encryption (PKE) schemes that are resilient to a-posteriori chosen-ciphertext and key-leakage attacks. Recently, Naor and Segev (CTYPTO'09) have proven that the Naor-Yung generic construction of PKE which is secure against chosen-ciphertext attack (CCA2) is also secure against key-leakage attacks. Their construction uses simulation-sound NIZK and leakage-resilient CPA-secure PKE, and the latter is a variant of the Cramer-Shoup cryptosystem. This CCA2-secure scheme is based on the hardness of the DDH problem. In this paper, we apply the generic construction of "Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption" (EUROCRYPT'02) to generalize the above work of Naor-Segev. In comparing to the first construction of Naor-Segev, ours "removes" simulation-sound NIZK which is not efficient component. We also extend it to stateful PKE schemes. Concretely, in the construction of the stateless PKE, we use the combination of any 1-universal hash proof system that satisfies the condition of a key-leakage extractor and any 2-universal hash proof system with some condition on the length of proof. In the case of the stateful PKE, we use the combination of two hash proof systems as in the case of stateless PKE and IND-CCA-secure symmetric encryption.

1 Introduction

KEY-LEAKAGE ATTACKS. Traditionally, the security of cryptographic schemes has been analyzed in an idealized setting, where an adversary only sees the specified input/output behavior of a scheme, but has no other access to its internal secret state. Unfortunately, in the real world, an adversary may often learn some partial information about secret state via various *key-leakage attacks*. Such attacks come in a large variety and include *side-channel attacks*, where the physical realization of a cryptographic primitive can leak additional information, such as the computation-time, power-consumption, radiation/noise/heat emission etc. The *cold-boot attack* is another example of a key-leakage attack, where an adversary can learn (imperfect) information about memory contents of a machine, even after the machine is powered down. Schemes that are proven secure in an idealized setting, without key leakage, may become completely insecure if the adversary learns even a small amount of information about the secret key.

* This research was supported in part by a grant of I-System Co. Ltd., NTT Information Sharing Platform Laboratories, and Ministry of Education, Culture, Sports, Science and Technology.

The introduction of memory attacks (or "cold boot attacks") by Halderman et al. [13], gave rise to the notion of *leakage resiliency*, presented by Akavia, Goldwasser, and Vaikuntanathan [2] and further explored by Naor and Segev [16]. In their definition, security holds even if the attacker gets some information of its choosing (depending on the value of the public key) on the scheme's secret key, with the only restriction that the total amount of leakage is bounded. Akavia et al. showed that the lattice-based PKE scheme of Regev [18] and the identity-based encryption of Gentry, Peikert, and Vaikuntanathan [12] are resilient to such leakage. PKE schemes presented in [2,16] are resilent to leakage of even $1-o(1)$ fraction of secret key (we call this the "leakage rate"). In particular, the paper [2] showed how this can be achieved under the LWE assumption, while the paper [16] showed that this can be achieved under the DDH assumption.

Naor and Segev [16] extended the framework of key leakage to the setting of chosen-ciphertext attack (LR-CCA2). On the theoretical side, they proved that the Naor-Yung paradigm is applicable in this setting as well, and obtained as corollary encryption schemes that are CCA2-secure with the leakage rate of $1-o(1)$ of the secret-key length. On the practical side, they proved that variants of the Cramer-Shoup cryptosystem are CCA1-secure with the leakage rate of $1/4$, and CCA2-secure with the leakage rate of $1/6$. In particular, Dodis et al. [10] proposed an efficient encryption scheme that is CCA2-secure with the leakage rate of $1 - o(1)$ of the secret-key length. Their scheme relies only on regular non-interactive zero-knowledge (NIZK), which can be instantiated by using the powerful Groth-Sahai techniques.

STATEFUL PUBLIC-KEY ENCRYPTION. In 2006, Bellare, Kohno, and Shoup [5] proposed the first model of stateful public-key encryption (StPE). The main goal of the StPE schemes is to reduce the cost of PKE by allowing a sender to maintain *state* that is reused across different encryptions. For example, one can obtain a stateful version of the ElGamal encryption in which a message M is encrypted to $(g^r, g^{rx}M)$ for public key g^x by maintaining the random value r and its corresponding value g^r as state so that g^r does not need to be computed each time.

Reducing the computational cost of public-key encryption is of particular importance for low-power mobile devices where computational resources are constrained (such as PDA and mobile phones) or sensors communicating with the relatively powerful servers or base stations [11,17]. Due to the efficiency gained from maintaining state, StPE schemes have potential to be employed in these settings. But, even in the environments that provide reasonable amount of computational resources, it is preferable to speed up public key operation.

The model of the StPE scheme proposed by Bellare et al. is specified by six algorithms: StPE = (Setup, KG, PKCk, NwSt, Enc, Dec) (all possibly randomized except the last) whose operation is illustrated in [5, Figure 2]. The approach that they adopt to construct StPE schemes is to convert specific PKE schemes such as DHIES [1] and the Kurosawa and Desmedts hybrid encryption scheme [15] into StPE schemes.

In 2008, Baek et al. [3] presented generic constructions of StPE, built several new StPE schemes and explained existing ones using their generic constructions. Some of them are built by using "identity-based technique" whereby one can construct PKE schemes secure against chosen-ciphertext attack in the standard model from identity-based encryption schemes.

OUR CONTRIBUTIONS. In the paper [16], Naor and Segev proved that a variant of the Cramer-Shoup cryptosystem [8] is secure against LR-CCA2 attack. This CCA2-secure scheme is based on the hardness of the DDH problem. From this idea, we make the following contributions in this paper:

1. We present a generic construction of a stateless PKE that is resilient to LR-CCA2 attack. This is the construction which generalizes Naor-Segev's leakage-resilient PKE scheme [16] using hash proof system (HPS). In this construction, we use the combination of any 1-universal HPS that satisfies the condition of a key-leakage extractor and any 2-universal HPS with some condition on the length of proof. $-$See Section 4.2.
2. We present the notion of LR-CCA2 in the case of StPE. Essentially, this notion is the same as that in the case of stateless PKE. We also present a generic construction of a StPE scheme that is secure against this attack. In this construction, we use the combination of two hash proof systems as in the case of stateless PKE and IND-CCA-secure symmetric encryption. This is also a new approach to achieve CCA2-secure StPE. $-$See Section 4.3.

These constructions do not rely on additional computational assumptions, and the resulting schemes are as efficient as the underlying hash proof system. Existing constructions of hash proof systems (see, for example, [8,14]) imply that our construction can be based on a variety of number-theoretic assumptions, including the decisional Diffie-Hellman (DDH) assumption and its progressively weaker d-Linear variants, the quadratic residuosity assumption, and Paillier's composite residuosity assumption.

Our construction of stateless PKE is LR-CCA2-secure with the leakage rate depending on the parameters of the underlying HPS. The Naor-Segev scheme [16] is an efficient instantiation which is LR-CCA2-secure with the leakage rate of $1/6$ of the secret-key length. This rate is not as good as the result proposed in [10], which is LR-CCA2-secure with the leakage rate of $1-o(1)$, but it is an important result for us to construct the generic construction of StPE that is resilient to LR-CCA2 attack. To the best of our knowledge, this is the first generic construction of StPE that is secure against this attack.

ROAD-MAP. We present notations, definitions, and tools in Section 2. The definition of LR-CCA2 in the cases of both stateless and stateful PKE appears in Section 3. In Section 4, we describe our generic constructions. Finally, we conclude in Section 5.

2 Preliminaries

In this section we present notions, definitions, and tools that are used in our constructions. Let n be the security parameter of the schemes, U_t the uniform distribution of $\{0,1\}^t$ (where $t \in N$), and $\mathrm{U}(\mathbf{S})$ the uniform distribution of the set $\mathbf{S}$. We denote by $s \overset{\$}{\leftarrow} \mathbf{S}$ the assignment of a uniformly distributed random element from the set $\mathbf{S}$ to the variable s. We use $negl(n)$ to denote a negligible function in n.

The *statistical distance* between two random variables X and Y over a finite domain Ω is $\Delta(X,Y) = \frac{1}{2}\Sigma_{\omega\in\Omega}|\Pr[X=\omega] - \Pr[Y=\omega]|$. We also write $\Delta(x,y)$ instead of $\Delta(X,Y)$. We say that two variables are ϵ-*close* if their statistical distance is at most ϵ. The *min-entropy* of a random variable X is $\mathrm{H}_\infty(X) = -\log(\max_x \Pr[X=x])$.

Dodis et al. [9] formalized the notion of *average min-entropy* that captures the remaining unpredictability of a random variable X conditioned on the value of a random variable Y, formally defined as follows:

$$\widetilde{\mathrm{H}}_\infty(X|Y) = -\log(E_{y \leftarrow Y}\left[2^{-\mathrm{H}_\infty(X|Y=y)}\right]).$$

The average min-entropy corresponds exactly to the optimal probability of guessing X, given knowledge of Y. The following bound on average min-entropy was proved in [9]:

Lemma 1 ([9]). *If Y has 2^r possible values and Z is any random variable, then $\widetilde{\mathrm{H}}_\infty(X|Y,Z) \geq \mathrm{H}_\infty(X|Z) - r$.*

One of the main tools in our constructions is a strong randomness extractor. The following definition naturally generalizes the standard definition of a strong extractor to the setting of average min-entropy:

Definition 1 ([9]). *A function* **Ext** : $\{0,1\}^n \times \{0,1\}^t \to \{0,1\}^m$ *is an* average-case (k,ϵ)-strong extractor *if for all pairs of random variables (X,I) such that $X \in \{0,1\}^n$ and $\widetilde{\mathrm{H}}_\infty(X|I) \geq k$ it holds that*

$$\Delta((\mathbf{Ext}(X,U_t),U_t,I),(U_m,U_t,I)) \leq \epsilon.$$

Hash Proof Systems. We present the framework of hash proof systems, introduced by Cramer and Shoup [8]. For simplicity we frame the description by viewing hash proof systems as key-encapsulation mechanisms (using the notation of Kiltz et al. [14]), and refer the reader to [8] for a more complete description.

Let X, L, W be non-empty sets, such that L is a proper subset of X, and $R_L \subset X \times W$ be a binary relation. For $x \in X$ and $w \in W$ with $(x,w) \in R_L$, we say that w is a *witness* for x. Note that it would be quite natural to require that for all $x \in X$, we have $(x,w) \in R_L$ for some $w \in W$ if and only if $x \in L$, and that the relation R_L is efficiently computable. We can also view X as the set of all ciphertexts, L as the set of all valid ciphertexts (i.e., those generated appropriately with the corresponding witness). We denote by $(x,w) \overset{\$}{\leftarrow} \mathcal{R}_L$ the *instance sampling algorithm* of L, i.e. choose random a pair (x,w) such that $x \in X, w \in W$, and $(x,w) \in R_L$.

A hash proof system $\mathbf{HPS} = (Param, KGen, Pub, Priv)$ consists of four algorithms that run in polynomial time. The algorithm $Param(1^n)$ generates system parameter sp. We denote by $\mathcal{SK}_n$ and $\mathcal{PK}_n$ the sets of secret keys and public keys that are produced by $KGen(sp)$. That is, $KGen(sp) : \{0,1\}^* \to \mathcal{SK}_n \times \mathcal{PK}_n$ for every $n \in N$. The deterministic public evaluation algorithm Pub is used to decapsulate valid ciphertexts $x \in L$ given a witness w of the fact that x is indeed valid (i.e., $(x,w) \in R_L$). The algorithm Pub receives as input a public key $pk \in \mathcal{PK}_n$, a valid ciphertext $x \in L$, and a witness w of the fact that $x \in L$, and outputs the encapsulated key $\pi \in \mathbf{\Pi}$ (where $\mathbf{\Pi}$ denotes the set of encapsulated symmetric keys). The deterministic private evaluation algorithm $Priv$ is used to decapsulate valid ciphertexts without knowing a witness w, but by using the secret key sk. That is, the algorithm $Priv$ receives as input a secret key $sk \in \mathcal{SK}_n$ and a ciphertext $x \in X$, and outputs the encapsulated key π.

Consider the probability space defined by choosing sk randomly from the set of secret keys. We say that a HPS is *1-universal* for language L if for all $x \in X \setminus L$ and $\pi \in \mathbf{\Pi}$, it holds that:

$$\Pr[Priv(sk, x) = \pi] = \frac{1}{|\mathbf{\Pi}|}.$$

We say that a HPS is *2-universal* for language L if for all $x, x^* \in X$ and $\pi, \pi^* \in \mathbf{\Pi}$, with $x \notin L \cup \{x^*\}$, it holds that

$$\Pr[Priv(sk, x) = \pi \mid Priv(sk, x^*) = \pi^*] = \frac{1}{|\mathbf{\Pi}|}.$$

It is easy to see that, if $\mathbf{HPS} = (Param, KGen, Pub, Priv)$ is 1-universal then

$$\Delta((pk, x, Priv(sk, x)), (pk, x, \mathrm{U}(\mathbf{\Pi}))) \leq negl(n),$$

and if $\mathbf{HPS}$ is 2-universal then

$$\Delta((pk, x, x^*, \pi^*, Priv(sk, x)), (pk, x, x^*, \pi^*, \mathrm{U}(\mathbf{\Pi}))) \leq negl(n),$$

where $\pi^* = Priv(sk, x^*)$ and $\mathrm{U}(\mathbf{\Pi}) \in \mathbf{\Pi}$ is sampled uniformly at random.

We also need an extension of this notion. The definition of *extended* HPS is the same as that of ordinary HPS, except that the proof system HPS accepts an extra input from a finite set E. In this setting, the public evaluation algorithm takes as input $pk \in \mathcal{PK}_n$, $x \in L$, $e \in E$, and a witness w of the fact that $x \in L$, and the private evaluation algorithm takes as input $sk \in \mathcal{SK}_n$, $x \in X$ and $e \in E$. We shall also require that elements of E are uniquely encoded as bit strings of length bounded by a polynomial in n, and that HPS provides an algorithm that efficiently determines whether a bit string is a valid encoding of an element of E.

We can modify in the obvious way to define *extended 1(2)-universal* HPS.

Next, we define a new property for HPS that is useful in our construction.

Definition 2. *We say that a hash proof system* $\mathbf{HPS} = (Param, KGen, Pub, Priv)$ *for a language L is a 1-universal* (λ, ϵ)*-key-leakage extractor if for any function* $f : \{0,1\}^* \to \{0,1\}^\lambda$ *we have*

$$\Delta((pk, x, f(sk), Priv(sk, x)), (pk, x, f(sk), U(\mathbf{\Pi}))) \leq \epsilon,$$

where $x \in_R X \setminus L$. *If* $\epsilon = negl(n)$ *we say that* $\mathbf{HPS}$ *is a 1-universal* λ*-key-leakage extractor for L.*

Note that, we can obtain a 1-universal λ-key-leakage extractor HPS by combining any 1-universal HPS with any strong extractor (See Section 4.1).

Subset Membership Problem. As a computational problem we require that the *subset membership problem* is hard in HPS, which means that for random valid ciphertext $x_0 \in L$ and random invalid ciphertext $x_1 \in X \setminus L$, the two ciphertexts x_0 and x_1 are computationally indistinguishable. This is formally captured by defining the advantage function $\mathrm{Adv}^{\mathrm{SM}}_{\mathrm{HPS},\mathcal{A}}(n)$ of an adversary $\mathcal{A}$ as

$$\mathrm{Adv}^{\mathrm{SM}}_{\mathrm{HPS},\mathcal{A}}(n) = \left|\mathrm{Pr}_{x_0 \leftarrow L}[\mathcal{A}(X, L, x_0) = 1] - \mathrm{Pr}_{x_1 \leftarrow X \setminus L}[\mathcal{A}(X, L, x_1) = 1]\right|.$$

If for any probabilistic polynomial-time adversary $\mathcal{A}$ it holds that $\mathrm{Adv}^{\mathrm{SM}}_{\mathrm{HPS},\mathcal{A}}(n)$ is negligible in n, then we say that the subset problem problem L is hard.

3 Models

3.1 Leakage-Resilient CCA2 Stateless Public-Key Encryption

In this section we review the notion of a-posteriori chosen-ciphertext key-leakage attack, introduced by Naor and Segev [16].

For a PKE scheme $\Pi = (KGen, Enc, Dec)$ we denote by $\mathcal{SK}_n$ and $\mathcal{PK}_n$ the sets of secret keys and public keys that are produced by $KGen(1^n)$. That is, $KGen(1^n) : \{0,1\}^* \rightarrow \mathcal{SK}_n \times \mathcal{PK}_n$ for every $n \in N$. The leakage oracle, denoted $\mathrm{Leak}(sk)$, takes as input a function $f : \mathcal{SK}_n \rightarrow \{0,1\}^*$ and outputs $f(sk)$. We say that an oracle machine $\mathcal{A}$ is a λ-key-leakage adversary if the sum of output lengths of all the functions that $\mathcal{A}$ submits to the leakage oracle is at most λ.

In this game, the adversary is allowed to adaptively access a decryption oracle $Dec(sk, \cdot)$ that receives as input a ciphertext and outputs a decryption using the secret key sk. We denote by $Dec_{\neq C}(sk, \cdot)$ a decryption oracle that decrypts any ciphertext other than C.

Definition 3 (LR-CCA2 security [16]). *A public-key encryption scheme $\Pi = (KGen, Enc, Dec)$ is* semantically secure against a-posteriori chosen-ciphertext λ-key-leakage attack *if for any probabilistic polynomial-time λ-key-leakage adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ it holds that*

$$\mathbf{Adv}^{\mathrm{KL,CCA2}}_{\Pi,\mathcal{A}}(n) \overset{def}{=} \left|\mathrm{Pr}\left[\mathbf{Expt}^{\mathrm{KL,CCA2}}_{\Pi,\mathcal{A}}(0) = 1\right] - \mathrm{Pr}\left[\mathbf{Expt}^{\mathrm{KL,CCA2}}_{\Pi,\mathcal{A}}(1) = 1\right]\right|$$

is negligible in n, where $\mathbf{Expt}^{\mathrm{KL,CCA2}}_{\Pi,\mathcal{A}}(b)$ is defined as follows:

1. $(pk, sk) \leftarrow KGen(1^n)$.
2. $(M_0, M_1, st_{\mathcal{A}_1}) \leftarrow \mathcal{A}_1^{\mathrm{Leak}(sk), Dec(sk,\cdot)}(pk)$ *such that* $|M_0| = |M_1|$.
3. $C \leftarrow Enc_{pk}(M_b)$.
4. $b' \leftarrow \mathcal{A}_2^{Dec_{\neq C}(sk,\cdot)}(C, st_{\mathcal{A}_1})$.
5. *Output* b'.

3.2 Leakage-Resilient CCA2 Stateful Public-Key Encryption

In this section we review the definition of StPE and its security as given in [3], then present the notion of a-posteriori chosen-ciphertext key-leakage attacks in the case of StPE.

Definition 4 (StPE [3]). *A StPE scheme consists of the following algorithms:*

- $\mathbf{StPE.Setup}(1^n) \to sp$: *Taking* 1^n *for a security parameter* $n \in \mathbb{Z}_{\geq 0}$ *as input, this algorithm generates a system parameter* sp *which includes* n.
- $\mathbf{StPE.KGen}(sp) \to (sk, pk)$: *Taking* sp *as input, this algorithm generates a private/public key pair* (sk, pk).
- $\mathbf{StPE.PKCk}(sp, pk) \to \delta$: *Taking* sp *and* pk *as input, this algorithm returns 1 if the public key* pk *is valid or returns 0 otherwise (i.e.,* $\delta \in \{0, 1\}$*).*
- $\mathbf{StPE.NwSt}(sp) \to st$: *Taking* sp *as input, this algorithm generates a new state.*
- $\mathbf{StPE.Enc}(sp, pk, st, M) \to (C, st)$: *Taking* sp, pk, st *and a plaintext* M *as input, this algorithm outputs a ciphertext* C *and state* st *which may be different from the state provided as input to this algorithm.*
- $\mathbf{StPE.Dec}(sp, sk, C) \to (M)$: *Taking* sp, sk, *and* C *as input, this deterministic algorithm outputs* M *which is either a plaintext or* $\perp$ *(meaning reject) message.*

We impose a usual consistency condition on $\mathbf{StPE}$: For any sp output by $\mathbf{StPE.Setup}$, (sk, pk) generated by $\mathbf{StPE.KG}$ and st output by either $\mathbf{StPE.NwSt}$ or $\mathbf{StPE.Enc}$, if (C, st) is an output of $\mathbf{StPE.Enc}(sp, pk, st, M)$, then $\mathbf{StPE.Dec}\ (sp, sk, C) = M$.

We now define the notion of a-posteriori chosen-ciphertext and key-leakage attacks in the case of StPE, which naturally extends that of chosen ciphertext security for StPE of [3] by giving an adversary the leakage oracle and only allowing him to query this oracle before challenge query.

Note that in the framework of this type of StPE, we can consider the secret key of the receiver and the state issued by the sender as possibly leaking information. Therefore, it seems natural to discuss also the security with state-leakage, in addition to the secret key. However, in this paper, we only focus on the security of the security with (secret) key-leakage.

Definition 5 (LR-CCA2 security of StPE). *Let* $\mathbf{StPE}$ *be a StPE scheme. Consider a game played with an attacker* $\mathcal{A}$*:*

Phase 1: *The game computes* $sp \leftarrow \mathbf{StPE.Setup}(1^n)$, $(pk_1, sk_1) \leftarrow \mathbf{StPE.KGen}(sp)$ *and* $st \leftarrow \mathbf{StPE.NwSt}(sp)$*. Note that* (sk_1, pk_1) *is the private/public key pair of the honest receiver* R_1*. The game sends* (sp, pk_1) *to* $\mathcal{A}$*.*

Phase 2: $\mathcal{A}$ *outputs public keys* $pk_2, \ldots, pk_t$ *of receivers* $R_2, \ldots, R_t$ *respectively, all of which are in the range of the second element of* $\mathbf{StPE.KGen}(sp)$*. Note that* $\mathcal{A}$ *may or may not know the private keys corresponding to the public keys* $pk_2, \ldots, pk_t$*.*

Phase 3: $\mathcal{A}$ *issues a number of (but polynomially many) queries, each of which is responded by the game. The type of each query and the action taken by the game are described as follows:*

- *Leakage queries, each of which is denoted by* f_j*: The game computes* $f_j(sk_1)$ *and sends this result to* $\mathcal{A}$*. Note that, the sum of output lengths of all leakage functions is at most* λ*, and these queries are requested before the challenge query.*
- *A challenge query* (m_0, m_1) *such that* $|m_0| = |m_1|$*: The game picks* $b \xleftarrow{\$} \{0, 1\}$*, computes* $(C^*, st) \leftarrow \mathbf{StPE.Enc}(sp, pk_1, st, m_b)$*, where* st *denotes the current state, and sends* C^* *to* $\mathcal{A}$*.*

- *Encryption queries, each of which is denoted by* (i, M) *where* $i \in \{1, ..., t\}$*: The game computes* $(C, st) \leftarrow \mathbf{StPE}.\mathbf{Enc}(sp, pk_i, st, M)$*, where* st *denotes the current state, and sends* C *to* $\mathcal{A}$*.*
- *Decryption queries, each of which is denoted by* $C \neq C^*$*: The game computes* $\mathbf{StPE}.\mathbf{Dec}(sp, sk_1, C)$ *and sends the resulting decryption message or* $\perp$ *(Reject) to* $\mathcal{A}$*.*

Phase 4: $\mathcal{A}$ *outputs its guess* $b' \in \{0, 1\}$*.*

We define $\mathcal{A}$*'s advantage by* $\mathbf{Adv}^{\mathrm{KL,CCA2}}_{\mathrm{StPE},\mathcal{A}}(n) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

The above StPE is semantically secure against a-posteriori chosen-ciphertext λ-key-leakage attack *if for any probabilistic polynomial-time* λ*-key-leakage adversary* $\mathcal{A}$*, the advantage of* $\mathcal{A}$ *is negligible in* n*.*

The LR-CCA2 security of $\mathbf{StPE}$ defined above can be considered in the KSK (Known Secret Key) or the USK (Unknown Secret Key) models [5]. In the KSK model, we assume that the attacker $\mathcal{A}$ possesses the corresponding private (secret) keys $sk_2, \ldots, sk_n$ of the public keys output in Phase 2 of the attack game.

3.3 Symmetric Encryption

First, we review the formal definition of symmetric encryption as follows.

Definition 6 (SYM [3]). *Let* $\mathcal{K}_D$ *be the key space. A symmetric encryption scheme, denoted by* **SYM***, consists of the following algorithms:*

- $\mathbf{SYM}.\mathbf{Enc}(K, M) \rightarrow e$ *: Taking a key* $K \in \mathcal{K}_D$ *and a plaintext* M *as input, this algorithm encrypts* M *into a ciphertext* e*.*
- $\mathbf{SYM}.\mathbf{Dec}(K, e) \rightarrow M$ *: Taking* $K \in \mathcal{K}_D$ *and* e *as input, this algorithm returns decrypts* e *into* M*. Note that* M *can be* $\perp$*.*

To construct IND-CCA secure StPE schemes, we need a symmetric encryption scheme secure against CCA attack in which the attacker does issue encryption queries. Now, a formal definition follows.

Definition 7 (IND-CCA of SYM [3]). *Let* **SYM** *be a symmetric encryption scheme as defined in Definition 6. Consider a game played with an attacker* $\mathcal{A}$*:*

Phase 1: *The game chooses* $K \overset{\$}{\leftarrow} \mathcal{K}_D$*.*

Phase 2: $\mathcal{A}$ *issues encryption queries, each of which is denoted by* M*. On receiving this, the game computes* $e \overset{\$}{\leftarrow} \mathbf{Enc}(K, M)$ *and gives* e *to* $\mathcal{A}$*.* $\mathcal{A}$ *also issues decryption queries, each of which is denoted by* e*. On receiving this, the game computes* $M \leftarrow \mathbf{Dec}(K, e)$ *and gives* M *to* $\mathcal{A}$*.*

Phase 3: $\mathcal{A}$ *issues a challenge query (a pair of plaintexts)* (m_0, m_1) *such that* $|m_0| = |m_1|$*. On receiving this, the game picks* $b \overset{\$}{\leftarrow} \{0, 1\}$*, computes* $e^* \overset{\$}{\leftarrow} \mathbf{SYM}.\mathbf{Enc}(K, m_b)$ *and gives* e^* *to* $\mathcal{A}$*.*

Phase 4: $\mathcal{A}$ *continues to issue encryption and decryption queries as in Phase 2. However, a restriction here is that* $\mathcal{A}$ *is not allowed to issue* e^* *as decryption query. The game responds to* $\mathcal{A}$*'s queries in the same way as it did in Phase 2.*

Phase 5: *$\mathcal{A}$ outputs its guess $b' \in \{0,1\}$.*

We define $\mathcal{A}$'s advantage by $\mathbf{Adv}^{\text{IND-CCA}}_{\text{SYM},\mathcal{A}}(n) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

We remark that as mentioned in [5], the symmetric encryption schemes meeting the IND-CCA definition can in fact be easily constructed, eg. using the encrypt-then-mac composition [6] with an AES mode of operation (such as CBC) and a MAC (such as CBC-MAC or HMAC [4]).

4 Generic Constructions from Hash Proof Systems

In this section, we give a generic construction of a 1-universal λ-key-leakage extractor that will be used in our constructions. We then present generic constructions of both stateless and stateful PKE scheme that are resilient to LR-CCA2 attack. In the case of stateless PKE, we apply the generic construction of Cramer-Shoup [7] to generalize the work of Naor-Segev. The stateful scheme is extension from the above stateless scheme.

4.1 The Construction of a 1-Universal λ-key-Leakage Extractor HPS

Assume that L is a membership indistinguishable language, $\mathbf{HPS} = (Param, KGen, Pub, Priv)$ be a 1-universal HPS for L. Let **Ext**: $\mathbf{\Pi} \times \{0,1\}^t \to \{0,1\}^m$ be an average-case $(\log|\mathbf{\Pi}| - \lambda, \epsilon)$-strong extractor. The following describes the hash proof system $\mathbf{HPS^{ext}} = (Param^{ext}, KGen^{ext}, Pub^{ext}, Priv^{ext})$ for language $L' = \{(x,s)|x \in L, s \in \{0,1\}^t\}$ (it is easy to see that if L is a membership indistinguishable language, so is L'):

$Param^{ext}$**:** the same as the original algorithm $Param$ of **HPS**.

$KGen^{ext}$**:** On input sp generated by $Param^{ext}$, choose $(pk, sk) \leftarrow KGen(sp)$, and return (pk, sk).

Pub^{ext}**:** On input a public key pk, a pair of variables $((x,s),w)$ with $(x,w) \overset{\$}{\leftarrow} \mathcal{R}_L$, and $s \overset{\$}{\leftarrow} \{0,1\}^t$, compute $\pi = Pub(pk,x,w)$, $\pi' = \mathbf{Ext}(\pi, s)$. Output π'.

$Priv^{ext}$**:** On input a private key sk, and (x,s), compute $\pi = Priv(sk,x)$, $\pi' = \mathbf{Ext}(\pi, s)$. Output π'.

The correctness of the scheme follows from the property that $Priv(sk,x) = Pub(pk, x, w)$ for any valid ciphertext $x \in L$ with witness w. Thus, the output of $Priv^{ext}$ is always the key encapsulated by Pub^{ext}.

The following theorem shows that this HPS is a 1-universal λ-key-leakage extractor.

Theorem 1. *Assuming that* **HPS** *is a 1-universal HPS for the language L, and* **Ext** *is an average-case* $(\log|\mathbf{\Pi}| - \lambda(n), \epsilon)$*-strong extractor, the hash proof system* $\mathbf{HPS^{ext}}$ *is a 1-universal* $\lambda(n)$*-key-leakage extractor for the language* L' *for any* $\lambda(n) \le \log|\mathbf{\Pi}| - \omega(\log n) - m$*, where n is the security parameter and m is the proof size of* $\mathbf{HPS^{ext}}$.

The details of the proof appear in the full version of this paper.

4.2 The Construction of Stateless Public-Key Encryption

Assume that L is a membership indistinguishable language, $\mathbf{HPS}_1 = (Param_1, KGen_1, Pub_1, Priv_1)$ be a 1-universal HPS for a language L, and $\mathbf{HPS}_2 = (Param_2, KGen_2, Pub_2, Priv_2)$ be an extended 2-universal HPS for the same language L. We define an encryption scheme $\Pi = (KGen, Enc, Dec)$ as follows:

Key Generation: On input 1^n for $n \in \mathbb{Z}_{\geq 0}$, generate system parameter $sp = (sp_1, sp_2)$, where$sp_1 \leftarrow Param_1(1^n)$, $sp_2 \leftarrow Param_2(1^n)$. Choose $(pk_1, sk_1) \leftarrow KGen_1(sp_1)$, $(pk_2, sk_2) \leftarrow KGen_2(sp_2)$, and return $pk = (pk_1, pk_2)$, $sk = (sk_1, sk_2)$.

Encryption: Given 1^n for $n \in \mathbb{Z}_{\geq 0}$, a public key $pk = (pk_1, pk_2)$, along with a message $M \in \mathbf{\Pi}_1$ (where $\mathbf{\Pi}_1$ is the domain of Pub_1; it may be, for example, $\{0,1\}^m$), do as follows.

E0: choose a pair $(x, w) \overset{\$}{\leftarrow} \mathcal{R}_L$;
E1: $\pi_1 = Pub_1(pk_1, x, w)$;
E2: $e = M \oplus \pi_1$;
E3: $\pi_2 = Pub_2(pk_2, x, w, e)$;
E4: Output $c = (x, e, \pi_2)$.

Decryption: Given 1^n for $n \in \mathbb{Z}_{\geq 0}$, a secret key $sk = (sk_1, sk_2)$, along with a ciphertext c, do the following.

D0: Parse c as a 3-tuple (x, e, π_2); output $\perp$ if c is not of this form.
D1: Compute $\pi_2' = Priv_2(sk_2, x, e)$.
D2: Test if $\pi_2' = \pi_2$; output $\perp$ and halt if this is not the case.
D3: Compute $\pi_1 = Priv_1(sk_1, x)$.
D4: Output $M = e \oplus \pi_1$.

Next, we show that this encryption scheme is LR-CCA2 secure.

Theorem 2. *Assume that L is a membership indistinguishable language, $\mathbf{HPS}_1$ is a 1-universal λ-key-leakage extractor for L, and $\mathbf{HPS}_2$ is an extended 2-universal HPS for L, with proofs π_2 of size $|\pi_2| = p \geq \lambda + \omega(\log n)$. Then the encryption scheme constructed from $\mathbf{HPS}_1, \mathbf{HPS}_2$ is semantically secure against a-posteriori chosen-ciphertext λ-key-leakage attacks, where n denotes the security parameter.*

The details of the proof appear in the full version of this paper.

Proof (Sketch). Before starting to prove, we state the following simple but useful lemma, which explicitly appears in [8, Lemma 4].

Lemma 2 ([8, Lemma 4]). *Let X_1, X_2, and F be events defined on some probability space. Suppose that the event $X_1 \wedge \neg F$ occurs if and only if $X_2 \wedge \neg F$ occurs. Then $|\Pr[X_1] - \Pr[X_2]| \leq \Pr[F]$.*

Let $f : \{0,1\}^* \to \{0,1\}^\lambda$ be the function that adversary used to learn λ bits from the secret key.

We will prove the theorem by using a sequences of game transitions.

Game 0: This is the original LR-CCA2 security game where the challenge ciphertext and the decryption queries are generated/answered correctly. In other words:

Challenge ciphertext $c^* = (x^*, e^*, \pi_2^*)$ of message M_b is computed as : $(x^*, w^*) \overset{\$}{\leftarrow} \mathcal{R}_L$; $\pi_1^* = Pub_1(pk_1, x^*, w^*)$; $e^* = M_b \oplus \pi_1^*$; $\pi_2^* = Pub_2(pk_2, x^*, w^*, e^*)$.

Let $b' \in \{0,1\}$ denote the output of $\mathcal{A}$, and let T_0 be the event that $b' = b$ in Game 0, so that

$$\mathbf{Adv}_{\Pi,\mathcal{A}}^{\mathrm{KL,CCA2}}(n) = |\Pr[T_0] - 1/2|. \tag{1}$$

Game 1: We now modify Game 0 to obtain a new Game 1. These two games are identical, except for a small modification to the encryption oracle. Instead of using the encryption algorithm as given to compute the target ciphertext c^*, we use a modified encryption algorithm, in which steps **E1** and **E3** are replaced by **E1**$'$: $\pi_1^* = Priv_1(sk_1, x^*)$ and **E3**$'$: $\pi_2^* = Priv_2(sk_2, x^*, e)$.

The change we have made is purely conceptual: the values of π_1^* and π_2^* are exactly the same in Game 1 as they were in Game 0. Therefore,

$$\Pr[T_1] = \Pr[T_0]. \tag{2}$$

Note that the encryption oracle now makes use of some components of the secret key, which is something the original encryption oracle does not do.

Game 2: We now modify Game 1 to obtain a new Game 2. We modify the challenge ciphertext, replacing step **E0** of the encryption algorithm by **E0**$'$: $x^* \overset{\$}{\leftarrow} X \setminus L$.

By the membership indistinguishability property of the language L, Game 1 and Game 2 are indistinguishable. Therefore, we have

$$|\Pr[T_2] - \Pr[T_1]| \leq negl(n). \tag{3}$$

Game 3: In this game, we modify the decryption oracle in Game 2 to obtain a new Game 3. We replace step $\mathbf{D2}$ in the decryption algorithm by $\mathbf{D2}'$: Test if $x \in L$; if it is the case, the oracle runs as previously; else outputs $\perp$ and halts.

Let R_3 be the event that in Game 3, some ciphertext C is submitted to the decryption oracle that is rejected in step $\mathbf{D2}'$ but that would have passed the test in step $\mathbf{D2}$. Note that if a ciphertext passes the test in $\mathbf{D2}'$, it would also have passed the test in $\mathbf{D2}$.

It is clear that Game 2 and Game 3 proceed identically until the event R_3 occurs. In particular, the event $T_2 \wedge \neg R_3$ and $T_3 \wedge \neg R_3$ are identical. So by Lemma 2, we have

$$|\Pr[T_3] - \Pr[T_2]| \leq \Pr[R_3]. \tag{4}$$

On the other hand, since $\mathbf{HPS}_2$ is a 2-universal HPS for L, with proofs π_2 of size $|\pi_2| = p \geq \lambda + \omega(\log n)$, using Lemma 1, we proved that the event R_3 occurs with only a negligible probability.

Game 4: This game is identical to Game 3, except for a small modification to the encryption oracle. In the challenge phase, replacing step **E1**$'$ by **E1**$''$: $\pi_1^* \overset{\$}{\leftarrow} \mathbf{\Pi}_1$.

It is clear by construction that

$$\Pr[T_4] = 1/2, \tag{5}$$

since in Game 4, the variable b is never used at all, and so the adversary's output is independent of b.

Claim 1.

$$| \Pr[T_4] - \Pr[T_3] | \leq negl(n). \tag{6}$$

Proof of Claim 1. Now, let us condition on a fixed value of pk_2, b, and the adversary's coins. In this conditional probability space, since the simulator rejects all ciphertexts (x, e, π_2) with $x \notin L$, it follows that the output of the simulator in Game 3 is completely determined as a function of $pk_1, x^*, f(sk_1)$ and $Priv_1(sk_1, x^*)$, while the output in Game 4 is determined as the same function of $pk_1, x^*, f(sk_1)$, and $\pi_1^* \in_R \mathbf{\Pi}_1$. Moreover, by independence, the joint distribution of $(pk_1, x^*, f(sk_1), \pi_1^*)$ does not change in passing from the original probability space to the conditional probability space. It now follows directly from the assumption $\mathbf{HPS}_1$ is a 1-universal λ-key-leakage extractor for L that

$$\varDelta((pk_1, x^*, f(sk_1), Priv_1(x^*, sk_1)), (pk_1, x^*, f(sk_1), \mathrm{U}(\mathbf{\Pi}_1))) \leq negl(n).$$

The claim follows. □

The theorem now follows immediately from (1) - (6). □

4.3 The Construction of Stateful Public-Key Encryption

Assume that $L, \mathbf{HPS}_1, \mathbf{HPS}_2$ are components as in Section 4.2, and **SYM** be a IND-CCA symmetric encryption. We assume that the HPS scheme $\mathbf{HPS}_1$ and the symmetric encryption scheme **SYM** are "compatible" meaning that the key space $\mathcal{K}_K$ of $\mathbf{HPS}_1$ is the same as the key space $\mathcal{K}_D$ of **SYM**. We define a StPE scheme $\mathbf{StPE}$ as follows:

StPE.Setup: On input 1^n for $n \in \mathbb{Z}_{\geq 0}$, return (system parameter) $sp = (sp_1, sp_2)$, where $sp_1 \leftarrow Param_1(1^n), sp_2 \leftarrow Param_2(1^n)$.

StPE.KGen: On input sp, choose $(pk_1, sk_1) \leftarrow KGen_1(sp_1)$ and $(pk_2, sk_2) \leftarrow KGen_2(sp_2)$. Then return $PK = (pk_1, pk_2)$, $SK = (sk_1, sk_2)$.

StPE.PKCk: On input sp and $PK = (pk_1, pk_2)$, output 1.

StPE.NwSt: On input sp, execute the instance sampling algorithm of L by **E0:**$(x, w) \overset{\$}{\leftarrow} \mathcal{R}_L$, and return $st = (x, w)$.

StPE.Enc: On input sp, a public key $PK = (pk_1, pk_2)$, a state st, along with a message M, do the following.

If st is of the form (x, w) then compute **E1:** $\pi_1 = Pub_1(pk_1, x, w)$; else, parse st as (x, w, PK, π_1). Next, do as follows.

E2: $e = \mathbf{SYM.Enc}(\pi_1, M)$;

E3: $\pi_2 = Pub_2(pk_2, x, w, e)$;

E4: Output $c = (x, e, \pi_2)$ and the new state $st = (x, w, PK, \pi_1)$.

StPE.Dec: On input sp, a secret key $SK = (sk_1, sk_2)$, along with a ciphertext c, do the following.

D0: Parse c as a 3-tuple (x, e, π_2); output $\perp$ if c is not of this form.

D1: Compute $\pi_2' = Priv_2(sk_2, x, e)$.

D2: Test if $\pi_2' = \pi_2$; output $\perp$ and halt if this is not the case.
D3: Compute $\pi_1 = Priv_1(sk_1, x)$.
D4: Output $M = \mathbf{SYM}.\mathbf{Dec}(\pi_1, e)$.

Note that, **StPE.PKCk** returns 1 (and does nothing else) as the KSK model implies that any public keys in this system are generated correctly following the algorithm **StPE.KGen**. (Namely the entity that has generated a public key must know the corresponding private key.)

In our construction of stateful encryption, there are two types of state: 1) (x, w) which is output by **StPE.NwSt**; 2)(x, w, PK, π_1) which is produced by the algorithm **StPE.Enc**. Note also that for state $st = (x, w)$ generated by the algorithm **StPE. NwSt**, $[\mathbf{StPE}.\mathbf{Enc}(PK, st, M)]_C = [\mathbf{StPE}.\mathbf{Enc}(PK, st', M)]_C$ for any st' output by **StPE.Enc** before **StPE.NwSt** is invoked to generate new state (different from st). Here "$[\mathbf{StPE}.\mathbf{Enc}(\cdots)]_C$" denotes the ciphertext part of an output of **StPE.Enc**.

We remark that the **StPE.Enc** algorithm becomes *highly efficient* when a sender sends encryptions to a single receiver: If the sender wants to send encryptions of M_1, ..., M_n to the same receiver whose public key is PK, he does not have to run Pub_1 for each plaintext M_i for $i = 1, \ldots, n$ but just runs them once at the beginning and then only does steps **E2, E3, E4** (in **StPE.Enc**).

Next, we show that in the KSK model, the above scheme is LR-CCA2 secure.

Theorem 3. *Assume that L is a membership indistinguishable language, $\mathbf{HPS}_1$ is a 1-universal λ-key-leakage extractor for L, $\mathbf{HPS}_2$ is an extended 2-universal HPS for L, with proofs π_2 of size $|\pi_2| = p \geq \lambda + \omega(\log n)$, and the underlying symmetric encryption* **SYM** *is IND-CCA secure. Then in the KSK model, the proposed generic StPE scheme* **StPE** *is semantically secure against a-posteriori chosen-ciphertext λ-key-leakage attacks. More precisely, we have*

$$\mathbf{Adv}^{\mathrm{KL,CCA2}}_{\mathrm{StPE},\mathcal{A}}(n) \leq \mathbf{Adv}^{\mathrm{IND\text{-}CCA}}_{\mathrm{SYM},\mathcal{B}}(n),$$

where n denotes the security parameter.

The details of the proof appear in the full version of this paper.

Remark. It follows from Theorems 1 and 2 (3) that the above constructions are LR-CCA2 secure with the leakage rate at most $\frac{\max\{|\pi_1|,|\pi_2|\}}{|sk_1|+|sk_2|}$ of the secret-key length, where $|a|$ denotes the size of a. An efficient instantiation of the proposed construction of stateless PKE is the encryption scheme of Naor-Segev [16], which is LR-CCA2 secure with the leakage rate of $1/6$ (the details description appear in the full version of this paper). Since the stateful encryption scheme can also be constructed from the same HPSs, we obtained an efficient instantiation of the proposed StPE with the same leakage rate.

5 Conclusion

We have introduced the generic constructions of both stateless and stateful PKE and proved that they are LR-CCA2 secure. In these constructions, we have used the combination of any 1-universal HPS that satisfies the condition of a key-leakage extractor and any 2-universal HPS with some condition on the length of proof. In the case of StPE, we have also used IND-CCA-secure symmetric encryption.

We leave it as an open problem to identify other generic cryptography primitives (other than hash proof systems) that are sufficient for constructing PKE schemes that are resilient to key-leakage. It is also an interesting open problem of constructing a StPE secure against both state-leakage and key-leakage. We could discuss the security with state-leakage in a similar way as that with randomness-leakage in our framework.

Acknowledgements. We thank the anonymous reviewers for their helpful comments.

References

1. Abdalla, M., Bellare, M., Rogaway, P.: The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES. In: Naccache, D. (ed.) CT-RSA 2001. LNCS, vol. 2020, pp. 143–158. Springer, Heidelberg (2001)
2. Akavia, A., Goldwasser, S., Vaikuntanathan, V.: Simultaneous Hardcore Bits and Cryptography against Memory Attacks. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 474–495. Springer, Heidelberg (2009)
3. Baek, J., Zhou, J., Bao, F.: Generic Constructions of Stateful Public Key Encryption and Their Applications. In: Bellovin, S.M., Gennaro, R., Keromytis, A.D., Yung, M. (eds.) ACNS 2008. LNCS, vol. 5037, pp. 75–93. Springer, Heidelberg (2008)
4. Bellare, M., Canetti, R., Krawczyk, H.: Keying Hash Functions for Message Authentication. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 1–15. Springer, Heidelberg (1996)
5. Bellare, M., Kohno, T., Shoup, V.: Stateful Public-Key Cryptosystems: How to Encryption with One 160-bit Exponentiaton. In: ACM CCS 2006, pp. 380–389. ACM Press (2006)
6. Bellare, M., Namprempre, C.: Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 531–545. Springer, Heidelberg (2000)
7. Cramer, R., Shoup, V.: Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (2002)
8. Cramer, R., Shoup, V.: Design and Analysis of Practical Public-Key Encryption Schemes Secure Against Adaptive Chosen Ciphertext Attack. SIAM J. Comput. 33(1), 167–226 (2003)
9. Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy Extractors: How to Generate Strong Keys from Biometrics and other Noisy Data. SIAM J. Comput. 38(1), 97–139 (2008)
10. Dodis, Y., Haralambiev, K., López-Alt, A., Wichs, D.: Efficient Public-Key Cryptography in the Presence of Key Leakage. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 613–631. Springer, Heidelberg (2010)
11. Gaubatz, G., Kaps, J.-P., Sunar, B.: Public Key Cryptography in Sensor Networks—Revisited. In: Castelluccia, C., Hartenstein, H., Paar, C., Westhoff, D. (eds.) ESAS 2004. LNCS, vol. 3313, pp. 2–18. Springer, Heidelberg (2005)
12. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for Hard Lattices and New Cryptographic Constructions. In: Proc. of the 40th Annual ACM Symposium on Theory of Computing, pp. 197–206 (2008)
13. Halderman, A., Schoen, D., Heninger, N., Clarkson, W., Paul, W., Calandrino, A., Feldman, J., Appelbaum, J., Felten, W.: Lest we remember: Cold Boot Attack on Encryption Keys. In: Oorschot, P. (ed.) USENIX Security Symposium, pp. 45–60. USENIX Association (2008)
14. Kiltz, E., Pietrzak, K., Stam, M., Yung, M.: A New Randomness Extraction Paradigm for Hybrid Encryption. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 590–609. Springer, Heidelberg (2009)

15. Kurosawa, K., Desmedt, Y.: A New Paradigm of Hybrid Encryption Scheme. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 426–442. Springer, Heidelberg (2004)
16. Naor, M., Segev, G.: Public-Key Cryptosystems Resilient to Key Leakage. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 18–35. Springer, Heidelberg (2009)
17. Phan, T., Huang, L., Dulan, C.: Challenge: Integrating Mobile Wireless Devices Into the Computational Grid. In: MobiCom 2002, pp. 271–278. ACM Press (2002)
18. Regev, O.: On Lattices, Learning With Errors, Random Linear Codes, and Cryptography. In: Proc. of the 37th Annual ACM Symposium on Theory of Computing, pp. 84–93 (2005)

How to Fix Two RSA-Based PVSS Schemes —Exploration and Solution

Kun Peng and Matt Henricksen

Institute for Infocomm Research, Singapore
dr.kun.peng@gmail.com

Abstract. At ACISP 2011, Peng shows that efficiency of two RSA-based PVSS schemes deteriorates to an intolerable level when practical parameter setting is adopted. In this paper, we show that Peng's newest PVSS scheme cannot solve the problem. A new PVSS scheme is designed in this paper to fix the problem in the two RSA-based PVSS schemes. It demonstrates that secure and practical PVSS can be designed on the base of RSA encryption.

1 Introduction

The first threshold secret sharing technique is Shamir's t-out-of-n secret sharing [18]. A dealer has a secret s and wants to share it among n share holders. The dealer builds a polynomial $f(x) = \sum_{j=0}^{t-1} \alpha_j x^j$ and sends $f(i)$ to the i^{th} share holder for $i = 1, 2, \ldots, n$ through a secure communication channel where $\alpha_0 = s$. Any t shares can be used to reconstruct the secret, while no information about the secret is obtained if the number of available shares is less than t. Secret sharing is widely employed in various secure protocols like e-auction, e-voting and multiparty computation. As most of the applications require public verifiability, very often secret sharing must be publicly verifiable. Namely, it must be publicly verified that all the n shares are consistently generated from a unique secret and any t of them reconstruct the unique secret, while the shares are kept secret. As the verification is public, any wrong-doing can be publicly detected by anyone and thus is undeniable.

Publicly verifiable secret sharing is usually called PVSS. The idea was firstly proposed by Feldman [7] and them developed into concrete schemes in [19,9,17,3,14]. As an important cryptographic tool, they are widely employed in various complex secure systems and applications like mix network [1], threshold access control [16], e-voting [12,11], encryption algorithm [4], zero knowledge proof [6], anonymous tokens [10] and fair exchange [13]. There are two important security properties in PVSS: completeness and soundness. Completeness requires that if the dealer is honest and does not deviate from the PVSS protocol he can always successfully prove validity of the shares. Soundness requires that if the dealer's proof of validity of the shares are passed with a non-negligible probability, it is guaranteed that any t shares reconstruct the same secret.

Peng explains in [15] that among the existing PVSS schemes [19,9,17,3], only two of them [9,3] are general and efficient and the other PVSS schemes have their drawbacks. Both of them employ RSA encryption in share distribution. However, Peng notices that high efficiency in [9] and [3] depends on a special condition: the share holders employ RSA encryption and use very small public keys like 3. As explained in [15], such small RSA public keys are often impractical in real life applications, especially in those applications unable to use appropriate padding of information like PVSS. Peng points out that when larger practical RSA keys are employed efficiency of the PVSS schemes in [9] and [3] decline to an intolerable level in real life as extremely large integers must be stored and used in computation. The PVSS scheme suggested by Peng for practical application is his own work, the PVSS scheme by Peng and Bao [14].

We show that the newest PVSS scheme by Peng and Bao [14] cannot be a solution to the problem in the two PVSS schemes in [9] and [3]. In this paper, to solve the problem, the proof techniques in [9] and [3] are replaced with a new proof method. Firstly, we show that extremely large integer can be avoided in the proof even full-length RSA public keys are employed. Then, a more efficient proof protocol is proposed, whose cost is independent of the size of the employed RSA public keys. The new proof protocol can flexibly satisfy the security requirements of the real life PVSS applications at a reasonable cost. It is formally illustrated that the concern found in [9] and [3] is solved and our efficiency improvement does not compromise or weaken security of PVSS in any way.

2 Background

Technical background in this paper is provided in this section.

2.1 Publicly Verifiable Secret Sharing

PVSS is a combination of secret sharing and publicly verifiable encryption and usually works as follows.

1. The dealer encrypts the shares for the share holders using their public keys and publishes the ciphertexts. Suppose $E_i()$ is P_i's public encryption algorithm, then $c_i = E_i(s_i)$ for $i = 1, 2, \ldots, n$ are published. Each share holder can decrypt the ciphertext for him and obtain his share.
2. The dealer commits to the share-generating polynomial (including all its coefficients) and publishes the commitment.
3. The dealer publicly proves that each encrypted share is generated by the committed share-generating polynomial without revealing them.

The following three important security properties are desired in PVSS.

– Completeness: if the dealer is honest and does not deviate from the PVSS protocol, he can always successfully prove validity of the shares.
– Soundness: if the dealer's proof of validity of the shares passes the corresponding verification with a non-negligible probability, it is guaranteed that they are generated by the committed share-generating polynomial such that any t of them reconstruct the same secret.

– Privacy: no information about the secret or any of its shares is revealed in the proof of validity of the shares. More precisely, a private PVSS scheme should employ zero knowledge proof techniques, which do not reveal any secret information as their proof transcripts can be simulated without any difference by a party without any knowledge of the secret or any of its shares.

The key technique in PVSS is the proof that each encrypted share is generated by the committed share-generating polynomial. Firstly, commitment to every share needs to be calculated from the commitment to the coefficients of the share-generating polynomial. Then it is only needed to prove that the share committed for each share holder is encrypted in the ciphertext for him using zero knowledge proof techniques.

2.2 The Two RSA-Based PVSS Schemes in [3] and [9]

The PVSS protocols in [3] and [9] are recalled in the following strictly according to their original presentations without any change or omission.

In [3], there are four PVSS protocols, two with fast recovery and two with delayed recovery. Note that PVSS with delayed recovery does not support real time secret reconstruction and has a quite limited range of applications. In Section 4.1 of [3], its first PVSS protocol with fast recovery is as follows where the secret integer is s and $S = g^s \bmod p$ is a public number and detailed definitions of functions like $Share()$, $Proof3()$ and $Proof6()$ can be found in [3].

1. Alice runs $Share(s) = ((s_1, \ldots, s_l); (a_1, \ldots, a_{k-1}))$, computes $S_i = g^{s_i} \bmod p$ for $i = 1, \ldots, l$ and $A_j = g^{a_j} \bmod p$ for $j = 1, \ldots, k-1$. Then she executes $Proof3(Share(\log_g(S)) = ((\log_g(S1), \ldots, \log_g(Sl)), (\log_g(A1), \ldots, \log_g(A_{k-1})))$.
2. For $i = 1, \ldots, l$, Alice computes $E_i = s_i^3 \bmod n_i$ and executes $Proof6(s_i, S_i = g^{s_i} \bmod p \ \wedge \ E_i = s_i^3 \bmod n_i \ \wedge \ abs(s_i) < (n_i - 1)/2)$.

In this PVSS protocol g is a generator of a large cyclic group in Z_p^*, n_i is the i^{th} share holder's RSA modulus and $Proof6()$ is a proof technique detailed in [3]. In Section 4.2 of [3], the PVSS protocol with fast recovery is extended to share a secret factorization. The other PVSS protocols with delayed recovery in [3] employ similar techniques.

In Pages 8-9 in [9], a dealer D has a secret s in Z_v and shares it using a polynomial $f(X) = s + \sum_{j=1}^{k-1} a_j X^j \bmod v$ where each $a_j \in_R Z_v$ and $a_{k-1} \neq 0$. Then D sets, for $i = 1, \ldots, l$, $s_i = (f(i) \bmod v) + (2^m - \delta_i)v$ where $\delta_i \in \{0, 1\}$. D sets and broadcasts $C_i = s_i^{e_i} \bmod n_i$ for $i = 1, 2, \ldots, l$. The PVSS protocol in [9] is described in its Page 9 and recalled as follows where V is a verifier.

1. D sets and sends to V $c_0 = BC_{(b,v)}(s, r_0)$ where $r_0 \in_R Z_N^*$.
2. D sets, for $j = 1, \ldots, k-1$, $c_j = BC_{(b,v)}(a_j, r_j)$, where $r_1, \ldots, r_{k-1} \in_R Z_N^*$. Then, for $i = 1, \ldots, l$, D sets $A_i = BC_{(b,v)}(s_i, t_i)$ and $B_i = BC_{(b,n_i)}(s_i, r'_i)$ where $t_i = \prod_{j=1}^{k-1} r_j^{i^j} \bmod N$ and $r'_i \in_R Z_N^*$. D broadcast $(c_1, \ldots, c_{k-1})$, $(A_1, \ldots, A_l)$ and $(B_1, \ldots, B_l)$.

3. V checks $A_i = \prod_{j=1}^{k-1} c_j^{i^j} \bmod N$.
4. For $i = 1, \ldots, l$, D executes with V $TRAN_{(b,v,n_i)}(A_i, B_i)$ and $PROOF[B_i = BC_{(b,n_i)}(s_i) \wedge D_i(s_i) = 0 \bmod n_i]$ where $D_i(X) = X^{e_i} - C_i$.

The symbols and denotations used in the PVSS protocol are dispersedly defined in many different places in Pages 5, 6, 7, 8 and 9 of [9]. Some of them are needed in our analysis and thus put together as follows.

– $N = PQ$, $P = 2p + 1$ and $Q = 2q + 1$ where p and q primes. Although not explicitly stated in [9], P and Q should be large secret primes (or at least product of large secret primes), otherwise polynomial factorization of N enables calculation of the v^{th}-root in polynomial time and breaks bindingness[1] of the commitment function $BC_{(b,v)}(s, r) = b^s r^v \bmod N$.
– b is a generator of the cyclic subgroup with order pq in Z_N^*. Although not explicitly stated in [9], p and q should be secret to guarantee hardness in factorizing N as implied by citation of [8] in [9].
– s_i is the secret share for the i^{th} share holder.
– $BC()$ is a commitment function $BC_{(b,v)}(s, r) = b^s r^v \bmod N$ and B_i is a commitment value.
– n_i is an RSA modulus for the i^{th} share holder's RSA encryption algorithm and e_i is the i^{th} share holder's RSA public key.
– $TRAN_{(b,v_1,v_2)}(c_1, c_2)$ is a proof primitive defined in Page 7 of [9]. It proves knowledge of integers x_1, x_2, r_1, r_2 such that $c_1 = BC_{(b,v_1)}(x_1, r_1)$ and $c_1 = BC_{(b,v_2)}(x_2, r_2)$ where $x_1 = x_2 \bmod v$.
– $PROOF[\,]$ is a key proof technique to be discussed.

2.3 The Concern Raised in [15]

A concern about the two existing RSA-Based PVSS schemes [3,9] is pointed out in [15].

The most important operation in PVSS is to prove and verify that the same share is encrypted in a ciphertext and committed to in a commitment. In Step 4 of the PVSS protocol (in Page 9 of [9]), the dealer needs to prove

$$PROOF[B_i = BC_{(b,n_i)}(s_i) \wedge D_i(s_i) = 0 \bmod n_i] \tag{1}$$

where the parameter setting is as follows.

– $N = PQ$, $P = 2p + 1$ and $Q = 2q + 1$ where p and q primes. Although not explicitly stated in [9], P and Q should be large secret primes (or at least product of large secret primes), otherwise polynomial factorization of N makes calculation of the v^{th}-root polynomial and breaks bindingness of the commitment function $BC_{(b,v)}(s, r) = b^s r^v \bmod N$.
– b is a generator of the cyclic subgroup with order pq in Z_N^*. Although not explicitly stated in [9], p and q should be secret to guarantee hardness in factorizing N as implied by citation of [8] in [9].

1 Bindingness here refers to the security property "......opening the commitment with different representations is equivalent to breaking RSA" in [9].

– s_i is a secret share.
– n_i is an RSA modulus.

According to the definition of $BC()$ and $D_i()$ in [9], this proof in the form $PROOF[\ ]$ is actually a proof of knowledge of secret integers s_i and r to satisfy

$$B_i = b^{s_i} r^{n_i} \bmod N \tag{2}$$

$$s_i^{e_i} = C_i \bmod n_i \tag{3}$$

where C_i is an RSA ciphertext. The proof primitive $PROOF[\ ]$ is defined in Page 6 of [9] and implementation of the proof is given in the so-called Example 1 in the same page. Applying the proof method in Example 1 to proof of (2) and (3) in [9] is as follows.

1. e_i is set to be 3.
2. The dealer publishes $c' = BC(s_i^2)$ and $c'' = BC(s_i^3)$.
3. The dealer runs two proof primitives $SQR_{(b,n_i)}(B_i : c')$ and $MUL_{(b,n_i)}$ $(B_i, c' : c'')$ defined earlier in Page 6 of [9].
4. The dealer opens $(c'' b^{-C_i})^{1/n_i}$.

In Section 4.1 of [3], its first PVSS protocol with fast recovery includes an operation, Step 2: "For $i = 1, ..., l$, Alice computes $E_i = s_i^3 \bmod n_i$ and executes $Proof6(s_i, S_i = g^{s_i} \bmod p \wedge E_i = s_i^3 \bmod n_i \wedge abs(s_i) < (n_i - 1)/2)$" to show that the same share s_i is committed to in S_i and encrypted in E_i where $Proof6(x, G = g^x \bmod p \wedge E = x^3 \bmod n_1 \wedge abs(x) < (n_1 - 1)/2)$ is realized in its Section 3.6 as follows.

1. Alice computes $\alpha = \frac{E - x^3}{n_1}$, $G_1 = g^{-x} \bmod N$, $G_2 = g^{-x^2} \bmod N$, $G_3 = g^{-x^3} \bmod N$ and $Z = g^{-\alpha n_1} \bmod N$.
2. Alice proves knowledge of x such that $G = g^x \bmod p$, $G_1 = g^{-x} \bmod N$, $G_2 = G_1^x \bmod N$, $G_3 = G_2^x \bmod N$ and $abs(x) < \lambda(N)/2)$ and knowledge of α such that $Z = (g^{-n_1})^\alpha \bmod N$.
3. The verifier checks the proofs, computes $T = g^{-E} \bmod N$ and checks that $G_3 = T/Z \bmod N$.

In Section 4.2 of [3], its second PVSS protocol with fast recovery includes an operation, Step 3: "For $i = 1, ..., l$, Alice computes $E_i = s_i^3 \bmod n_i$ and executes $Proof6(s_i, S_i = g^{s_i} \bmod p \wedge E_i = s_i^3 \bmod n_i \wedge abs(s_i) < (n_i - 1)/2)$" to show that the same share s_i is committed to in S_i and encrypted in E_i.

So the PVSS schemes in [9] and [3] are only specified in the case where the share holders use 3 as their RSA public keys. Their high efficiency is also achieved in this case.

The specification of proof of validity of shares for general RSA keys seems to work in theory. However, in practice, it includes a very costly operation, especially when e_i is large. The operation is calculation of $s_i^2, s_i^4, \ldots, s_i^{2^{L-1}}$ where L is the bit length of e_i. In theory, they can be calculated using $L - 1$ square operations $S_1 = s_i^2$, $S_2 = S_1^2, \ldots, S_{L-1} = S_{L-2}^2$ and then used in $SQR()$ and

$MUL()$ as the secret logarithms to be proved knowledge of where $S_k = s_i^{2^k}$. In practice, the size of S_k increases rapidly and becomes intolerably large very quickly. In [9], $s_i = (f(i) \bmod v) + (2^m - \delta_i)v$ where $f()$ is the share-generating polynomial proposed by Shamir [18] and employed by most threshold PVSS schemes, $m = O(|N|)$, v is an integer decided by the dealer or a verifier and $\delta_i \in \{0,1\}$. So 2^m and thus s_i should be hundreds of bits long in a practical secure setting. When e_i is large (e.g. no smaller than 65537), $s_i^{2^{L-1}}$ is millions or even billions of bits long. So large integers are difficult to store or process (e.g. use them to calculate S_k and c''), not to mention $s_i^2, s_i^4, \ldots, s_i^{2^{L-1}}$ are used in $SQR()$ and $MUL()$ as secret logarithms to be proved knowledge of.

It is explained in [15] why we cannot calculate $S_k = S_{k-1}^2$ with a multiplicative modulus instead in Z. The key point is what modulus to use. As $s_i^2, s_i^4, \ldots, s_i^{2^{L-1}}$ are in the form of exponents to the base b, the multiplicative modulus must be a multiple of the order of b. However, the order of b and its multiples are secret since factorization of N must be hard. If the dealer knows the order of b, he can open his commitment in many different ways and commitment of the secret fails. So, the dealer cannot know the order of b or any of its multiples. Therefore, $s_i^2, s_i^4, \ldots, s_i^{2^{L-1}}$ must be calculated in Z and become extremely large when a large enough secure public key is employed in RSA cipher. In summary, intolerably high cost is inevitable in [9] if secure RSA public keys are employed. The same problem exists in [3] as well. The concerns in the two PVSS schemes are still not solved in the most recent PVSS scheme by Peng and Bao in [14]. The range proofs in the PVSS scheme in [14] has its own security concern. In Section 5 of [14], to guarantee that s_i is in the range $\{0, 1, \ldots, N_i\}$ where decryption modulus n_i is denoted in [14] as N_i, it is proved that $s_i + s'_i = N_i$ where s'_i is another integer with unlimited distribution. Obviously, $s_i + s'_i = N_i$ cannot guarantee that s_i is in the range $\{0, 1, \ldots, N_i\}$. When s_i is out of the range, it is changed when its encryption is decrypted.

3 Secure and Efficient PVSS Based on RSA Encryption

In this section, it is firstly shown that the concern in the existing RSA-based PVSS schemes can be eased. In $CHCK2$ in [9], X should be expected to fall in a larger range to achieve completeness. To maintain soundness with this change, e can be smaller and m can be larger. In $Proof6()$ in [3], the range proof should prove that s_i is in a smaller range Z_{n_i}. More importantly, an efficient proof protocol is needed to specify the proof of validity of shares encrypted in RSA ciphertexts with large RSA in [9] and [3]. Firstly, it is shown in Section 3.1 that extremely large integer can be avoided in the proof. Then in Section 3.2, a more advanced proof protocol is proposed to further improve efficiency.

3.1 Avoiding Too Large Integers

The method to specify $PROOF[\ ]$ is modified to avoid extremely large integers as follows.

1. The dealer publishes $c_{k+1} = c_k^{s_i} r_k^{n_i} \bmod N$ for $k = 0, 1, \ldots, e_i - 2$ where $c_0 = B_i$ and r_k is randomly chosen from Z_N^* and their validity is proved as follows.
   (a) The dealer proves that the committed integer in c_1 is the square of the the committed integer in B_i through $SQR_{(b,n_i)}(B_i : c_1)$ where he uses s_i as his secret witness.
   (b) The dealer proves that the committed integer in c_2 is the product of the the committed integers in B_i and c_1 through $MUL(B_i, c_1 : c_2)$ where he uses s_i as his secret witness.
   (c) The dealer proves that the committed integer in c_3 is the product of the the committed integers in B_i and c_2 through $MUL(B_i, c_2 : c_3)$ where he uses s_i as his secret witness.
   ......
   (d) The dealer proves that the committed integer in c_{e_i-1} is the product of the the committed integers in B_i and c_{e_i-2} through $MUL(B_i, c_{e_i-2} : c_{e_i-1})$ where he uses s_i as his secret witness.
2. The dealer opens $(c_{e_i-1} b^{-C_i})^{1/n_i}$.

As s_i is used as the secret exponent in calculating $c_1, c_2, \ldots, c_{e_i-1}$ and the secret witness in all the $SQR()$ and $MUL()$ proof primitives, no extremely large integer is needed in this modified specification of $PROOF[\ ]$. However, cost of the modified specification is linear in $O(e_i)$, so is still too high.

3.2 More Efficient Specification of *PROOF*[ ]

A new proof method whose cost is independent of e_i is proposed to specify $PROOF[\ ]$ without using any extremely large integer. Firstly, a proof primitive is designed in Fig 1 to prove satisfaction of (2) and (3).

This new proof primitive in Fig 1 is called basic proof in this paper. As illustrated in Theorem 1 and Theorem 2, an honest dealer's proof can always be successfully verified and passing the verification in basic proof with a probability larger than 0.5 guarantees satisfaction of (2) and (3).

Theorem 1. *If the dealer is honest and (2) and (3) are satisfied, he can always pass the verification in basic proof.*

Proof: As the dealer is honest, he knows integers S and R such that

$$B = b^S R^{n_i} \bmod N \tag{4}$$
$$C = S^{e_i} \bmod n_i \tag{5}$$

– If $u = 0$, the dealer publishes $S' = S$ and $R' = R$. (4) and (5) imply

$$B = b^{S'} R'^{n_i} \bmod N$$
$$C = S'^{e_i} \bmod n_i.$$

and the verification is passed.

– If $u = 1$, as the dealer is honest he commits to s_i in B_i and publish $a = b^{s_i S} R''^{n_i} \bmod N$ such that $s_i S$ is committed to in a. (4) implies that S is committed to in B. So the dealer can successfully give a proof $MUL(B_i, B : a)$ and pass its verification. The dealer publishes

$$S' = s_i S \bmod n_i$$
$$R' = R'' b^{(s_i S - S')/n_i} \bmod N.$$

As he is honest, he has encrypted s_i in

$$C_i = s_i^{e_i} \bmod n_i$$

So

$$\begin{aligned} b^{S'} R'^{n_i} &= b^{s_i S \bmod n_i} (R'' b^{(s_i S - S')/n_i})^{n_i} \\ &= b^{s_i S} R''^{n_i} = a \bmod N \end{aligned}$$

and

$$C_i C = s_i^{e_i} S^{e_i} = (s_i S)^{e_i} = S'^{e_i} \bmod n_i$$

and the verification is passed. □

1. The dealer randomly chooses S in Z_{n_i} and R in Z_N^* and publishes

$$B = b^S R^{n_i} \bmod N$$
$$C = S^{e_i} \bmod n_i$$

2. A verifier or multiple cooperating verifiers or a (pseudo)random algorithm generates a random bit u as a challenge.
3. If $u = 0$, the dealer publishes $S' = S$ and $R' = R$. If $u = 1$, the dealer
   (a) randomly chooses R'' in Z_N^* and publishes $a = b^{s_i S} R''^{n_i} \bmod N$;
   (b) gives a proof $MUL(B_i, B : a)$ to demonstrate that the integer committed in a is the product of the two integers committed in B_i and B respectively;
   (c) publishes

$$S' = s_i S \bmod n_i$$
$$R' = R'' b^{(s_i S - S')/n_i} \bmod N.$$

Public verification of the dealer's proof is as follows.

– If $u = 0$, anyone can verify

$$B = b^{S'} R'^{n_i} \bmod N$$
$$C = S'^{e_i} \bmod n_i.$$

– If $u = 1$, anyone can verify the dealer's $MUL()$ proof and

$$a = b^{S'} R'^{n_i} \bmod N$$
$$C_i C = S'^{e_i} \bmod n_i.$$

Fig. 1. Basic proof of satisfaction of (2) and (3)

Theorem 2. *If the dealer passes the verification in basic proof with a probability larger than 0.5, (2) and (3) are satisfied.*

Proof: As the dealer passes the verification in basic proof with a probability larger than 0.5, he can provide integers S' and R' when $u = 0$ such that

$$B = b^{S'} R'^{n_i} \bmod N, \tag{6}$$

$$C = S'^{e_i} \bmod n_i, \tag{7}$$

otherwise, his proof fails when $u = 0$ and the probability that he can pass the verification is no larger than 0.5.

As the dealer passes the verification in basic proof with a probability larger than 0.5, he can prove $MUL(B_i, B : a)$ and provide integers S' and R' when $u = 1$ such that

$$a = b^{S'} R'^{n_i} \bmod N, \tag{8}$$

$$C_i C = S'^{e_i} \bmod n_i, \tag{9}$$

otherwise, his proof fails when $u = 1$ and the probability that he can pass the verification is no larger than 0.5.

Denote S' and R' in (6) and (7) as S'_1 and R'_1 and denote S' and R' in (8) and (9) as S'_2 and R'_2, and we have

$$B = b^{S'_1} {R'_1}^{n_i} \bmod N, \tag{10}$$

$$C = {S'_1}^{e_i} \bmod n_i, \tag{11}$$

$$a = b^{S'_2} {R'_2}^{n_i} \bmod N, \tag{12}$$

$$C_i C = {S'_2}^{e_i} \bmod n_i. \tag{13}$$

As $MUL(B_i, B : a)$ implies that the message committed in a is the product of the messages committed in B_i and B, (12) and (10) imply that the dealer knows S'_2/S'_1 and R'' such that

$$B_i = b^{S'_2/S'_1} R''^{n_i} \bmod N$$

where R'' is an integer in Z_N^*. As (13) divided by (11) yields

$$C_i = (S'_2/S'_1)^{e_i} \bmod n_i,$$

the dealer knows $s_i = S'_2/S'_1$ and $r = R''$ such that

$$B_i = b^{s_i} r^{n_i} \bmod N$$

$$s_i^{e_i} = C_i \bmod n_i$$

and thus (2) and (3) are satisfied. □

Although the basic proof only guarantees soundness with a probability 0.5, it can be repeated T (a security parameter) times such that soundness is guaranteed with an overwhelmingly large probability $1 - 2^{-T}$ when all the T instances of proof pass their verifications. For example, when T is 20, the probability of failure of soundness is reduced to 2^{-20} (which is smaller than one out of one million); and when T is 30, the probability of failure of soundness is reduced to 2^{-30} (which is smaller than one out of one billion). Obviously, running basic proof 20 or 30 times can guarantee soundness in a practical sense, but is still more efficient than $O(\log_2 e_i)$ proofs involving extremely large integers (e.g. millions or billions of bits long). Especially, cost of the new solution does not depends on e_i and is only determined by the security parameter T. So our new solution achieves better flexibility in real life applications. In different applications T can be flexibly adjusted to guarantee soundness at only the necessary cost. Replacing the implementation of proof $PROOF[\,]$ in [9] and the same proof in [3] with our new solution does not compromise or weaken privacy of the PVSS scheme as illustrated in Theorem 3.

Theorem 3. *Basic proof is private. More precisely, its proof transcript can be simulated by a party without any knowledge of any secret such that the simulating transcript has the same distribution with the real proof transcript.*

Proof: The transcript of basic proof contains B, C, u, S', R' and sometimes a and the variables published by an $MUL()$ proof. As $MUL()$ has been proved to be private in [9], we only need to consider how to simulate B, C, u, S', R', a. A party without any knowledge of any secret can simulate them as follows.

1. He randomly chooses a bit u.
2. If $u = 0$,
   (a) he randomly chooses S' from Z_{n_i} and R' from Z_N^*;
   (b) he calculates

$$B = b^{S'} R'^{n_i} \bmod N$$
$$C = S'^{e_i} \bmod n_i.$$

3. If $u = 1$,
   (a) he randomly chooses S' from Z_{n_i} and R' from Z_N^*;
   (b) he calculates

$$a = b^{S'} R'^{n_i} \bmod N$$
$$C = S'^{e_i} / C_i \bmod n_i.$$

In both the real proof transcript of basic proof and this simulating transcript,

– B is uniformly distributed in Z_N^*;
– C is uniformly distributed in Z_{n_i};
– u is uniformly distributed in $\{0, 1\}$;
– S' is uniformly distributed in Z_{n_i};
– R' is uniformly distributed in Z_N^*;
– a is uniformly distributed in Z_N^*;
– if $u = 0$,

$$B = b^{S'} R'^{n_i} \bmod N$$
$$C = S'^{e_i} \bmod n_i;$$

If $u = 1$,

$$a = b^{S'} R'^{n_i} \bmod N$$
$$C_i C = S'^{e_i} \bmod n_i.$$

As the two transcripts have the same distribution, basic proof is private. □

4 Conclusion

In this paper, we have revisited the two PVSS schemes in [9,3], which are shown to be not so efficient in practice as widely believed in [15]. Impractical proof techniques in them are fixed and optimised. The resulting PVSS scheme overcomes their problem and avoids any highly costly operation. So really general and efficient PVSS is implemented in this paper.

References

1. Adida, B., Wikström, D.: How to Shuffle in Public. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 555–574. Springer, Heidelberg (2007)
2. Boneh, D., Shacham, H.: Fast variants of RSA. CryptoBytes 5(1), 1–9 (2002)
3. Boudot, F., Traoré, J.: Efficient Publicly Verifiable Secret Sharing Schemes with Fast or Delayed Recovery. In: Varadharajan, V., Mu, Y. (eds.) ICICS 1999. LNCS, vol. 1726, pp. 87–102. Springer, Heidelberg (1999)
4. Chandran, N., Ostrovsky, R., Skeith III, W.E.: Public-Key Encryption with Efficient Amortized Updates. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 17–35. Springer, Heidelberg (2010)
5. Damgård, I., Cramer, R.: On Σ-protocols. In: Cryptologic Protocol Theory (2002), http://www.daimi.au.dk/~ivan/Sigma.ps
6. Damgård, I., Thorbek, R.: Non-interactive Proofs for Integer Multiplication. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 412–429. Springer, Heidelberg (2007)
7. Feldman, P.: A practical scheme for non-interactive verifiable secret sharing. In: FOCS 1987, pp. 427–437 (1987)
8. Fujisaki, E., Okamoto, T.: Statistical Zero Knowledge Protocols to Prove Modular Polynomial Relations. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 16–30. Springer, Heidelberg (1997)
9. Fujisaki, E., Okamoto, T.: A Practical and Provably Secure Scheme for Publicly Verifiable Secret Sharing and Its Applications. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 32–46. Springer, Heidelberg (1998)
10. Ge, H., Tate, S.R.: A Direct Anonymous Attestation Scheme for Embedded Devices. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 16–30. Springer, Heidelberg (2007)
11. Juels, A., Catalano, D., Jakobsson, M.: Coercion-Resistant Electronic Elections. In: Chaum, D., Jakobsson, M., Rivest, R.L., Ryan, P.Y.A., Benaloh, J., Kutylowski, M., Adida, B. (eds.) Towards Trustworthy Elections. LNCS, vol. 6000, pp. 37–63. Springer, Heidelberg (2010)

12. Kiayias, A., Yung, M.: Tree-Homomorphic Encryption and Scalable Hierarchical Secret-Ballot Elections. In: Sion, R. (ed.) FC 2010. LNCS, vol. 6052, pp. 257–271. Springer, Heidelberg (2010)
13. Küpçü, A., Lysyanskaya, A.: Optimistic Fair Exchange with Multiple Arbiters. In: Gritzalis, D., Preneel, B., Theoharidou, M. (eds.) ESORICS 2010. LNCS, vol. 6345, pp. 488–507. Springer, Heidelberg (2010)
14. Peng, K., Bao, F.: Efficient Publicly Verifiable Secret Sharing with Correctness, Soundness and ZK Privacy. In: Youm, H.Y., Yung, M. (eds.) WISA 2009. LNCS, vol. 5932, pp. 118–132. Springer, Heidelberg (2009)
15. Peng, K.: Impracticality of Efficient PVSS in Real Life Security Standard (Poster). In: Parampalli, U., Hawkes, P. (eds.) ACISP 2011. LNCS, vol. 6812, pp. 451–455. Springer, Heidelberg (2011)
16. Saxena, N., Tsudik, G., Yi, J.: Threshold cryptography in p2p and manets: The case of access control. Computer Networks 51(12), 3632–3649 (2007)
17. Schoenmakers, B.: A Simple Publicly Verifiable Secret Sharing Scheme and Its Application to Electronic Voting. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 148–164. Springer, Heidelberg (1999)
18. Shamir, A.: How to share a secret. Communication of the ACM 22(11), 612–613 (1979)
19. Stadler, M.A.: Publicly Verifiable Secret Sharing. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 190–199. Springer, Heidelberg (1996)

Relation between Verifiable Random Functions and Convertible Undeniable Signatures, and New Constructions

Kaoru Kurosawa[1], Ryo Nojima[2], and Le Trieu Phong[2]

[1] Ibaraki University
kurosawa@mx.ibaraki.ac.jp
[2] NICT, Japan
{ryo-no,phong}@nict.go.jp

Abstract. Verifiable random functions (VRF) and selectively convertible undeniable signature (SCUS) schemes were proposed independently in the literature. In this paper, we observe that they are tightly related. This directly yields several deterministic SCUS schemes based on existing VRF constructions. In addition, we create a new probabilistic SCUS scheme, which is very compact. The confirmation and disavowal protocols of these SCUS are efficient, and can be run either sequentially, concurrently, or arbitrarily. These protocols are based on what we call *zero-knowledge protocols for generalized DDH and non-DDH*, which are of independent interest.

Keywords: Selectively convertible undeniable signatures, verifiable random function, standard model.

1 Introduction

1.1 Background

Selectively convertible undeniable signatures (**SCUS**) were introduced by Boyar, Chaum, Damgård, and Pedersen [2] in 1990, extending the concept of undeniable signatures of Chaum and Antwerpen [5]. Recall that the verification of undeniable signatures is restricted, namely controlled by the signer. Any verifier needs to run an interactive protocol with the signer to check whether the purported signature is valid. However, in SCUS schemes, the verification of signatures can be additionally done non-interactively; namely the signer now can release a piece of information, often called a converter, to make his signature publicly checked.

SCUS schemes can be used directly, when one would like to allow some verifiers to freely check his signature, while the other cannot do so. For example, IACR members who receive the converter, can directly verify the signature, while non-members cannot. SCUS schemes can also be used as a the main building block in protocols such as fair payment [3]. There is a long list of work on SCUS schemes in the literature, including [8,12,14,22,24] in the standard model.

Table 1. Some recent SCUS schemes in the standard model, operating on a pairing group $\mathbb{PG} = (G, G_T, g, q, e)$, where g is a generator of G, and the order $|G| = |G_T| = q$ for prime q, and $e : G \times G \to G_T$. The notation 3_G means 3 group elements in G; 1_{G_T} means 1 group element in G_T. Typically, $s_G = 160_G$

SCUS schemes	Signature	Converter	Public keys
	(all sizes are in group elements)		
PKO [22]	$\geq 3_G$	2_G	$\geq 12_G$
SM [24]	4_G	2_G	$\approx s_G$
Our $\mathsf{SCUS}_{\mathsf{DY}}$ (Sect.4.2)	1_{G_T}	1_G	2_G
Our $\mathsf{SCUS}_{\mathsf{HW}}$ (Sect.4.2)	1_{G_T}	$\approx s_G$	$\approx s_G$
Our $\mathsf{SCUS}_{\mathsf{BB}}$ (Sect.5)	$1_{\mathbb{Z}_q} + 1_{G_T}$	1_G	4_G

On the other hand, *verifiable random functions* (**VRF**s), later introduced by Micali, Rabin, and Vadhan [17] in 1999, are like pseudo-random functions in the sense that their outputs are indistinguishable from random. However, unlike standard pseudo-random functions, the output of VRFs can be proved coming from a given input if one (owning the secret key) releases an additional piece of information, often called the proof of correctness.

VRFs have been used in various contexts, including resettable zero-knowledge proofs [18], micro-payment schemes [19], updatable zeroknowledge databases [16], and verifiable transaction escrow schemes [11].

1.2 Our Contributions

We first show VRF implies deterministic SCUS. The reverse side holds if SCUS has an additional property called uniqueness. Furthermore, probabilistic SCUS can be seen as "randomized" VRF (taking randomness as a function input). Perhaps surprisingly, these facts were not notified before in the literature of both SCUS and VRF.

Then, also based on the relation, we construct new deterministic SCUS in the standard model. Going further, we build a probabilistic SCUS scheme with *very neat* parameters. A comparison with the best known schemes in the standard model is given in Table 1. More details are as follows.

Deterministic SCUS from VRF. We show that VRFs directly imply deterministic SCUS schemes. This result, coupling with our zero-knowledge protocols for generalized DDH and non-DDH, then yields several efficient SCUS schemes. We give in Sect.4.2 two concrete deterministic SCUS schemes denoted as $\mathsf{SCUS}_{\mathsf{DY}}$ and $\mathsf{SCUS}_{\mathsf{HW}}$, respectively based on the VRFs of Dodis, Yampolskiy [7] and Hohenberger, Waters [10]. We note that $\mathsf{SCUS}_{\mathsf{DY}}$ requires small signing space, while $\mathsf{SCUS}_{\mathsf{HW}}$ does not.

Interlude: A new probabilistic SCUS. The converters and/or public keys in the above deterministic SCUS schemes are a bit long (e.g., of at least

160 group elements in $\mathsf{SCUS}_{\mathsf{HW}}$). Inspired from $\mathsf{SCUS}_{\mathsf{DY}}$ and the technique of Boneh and Boyen [1], we newly construct a more compact probabilistic SCUS scheme, called $\mathsf{SCUS}_{\mathsf{BB}}$, with arbitrary signing space. The scheme is given in Sect.5.

VRF from SCUS. We investigate the other side as well, namely VRFs from SCUS schemes with a unique property. The property requires, for *every*[1] public key, there is one valid signature for each message in SCUS. However, since there is no known unique SCUS scheme currently, this is just of theoretical interest. See Sect.4.3 for the construction.

Note that the confirmation and disavowal protocols for our SCUS constructions can be efficiently and elegantly realized from what we call *zero-knowledge protocols for generalized DDH and non-DDH* in Sect.3. These protocols are of independent interest, and may be useful in other context as well.

Due to the restriction of space, we leave all the proofs of theorems on the full version [13].

2 Definitions

By $a \overset{\$}{\leftarrow} U$ we mean a is chosen randomly from a set U, and by $\mathcal{A}^{\mathcal{O}(\cdot)}$ we mean the adversary $\mathcal{A}$ gets access to the oracle $\mathcal{O}(\cdot)$. The value $|a|$ is the length in bits of the element a, while $|\mathcal{H}|$ is the order of a group $\mathcal{H}$.

VERIFIABLE RANDOM FUNCTION. The function, described as $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Func}, \mathsf{V})$, is as follows.

- $\mathsf{Gen}(1^\lambda)$: return the public key pk and the secret key sk.
- $\mathsf{Func}_{sk}(x)$: return the (pseudo-random) output $y = F_{sk}(x)$ for $F_{sk}(\cdot) : \mathsf{Domain}(\lambda, pk) \to \mathsf{Range}(\lambda, pk)$[2], and its proof of correctness $\pi = \pi_{sk}(x)$.
- $\mathsf{V}_{pk}(x, y, \pi)$: return 1 (meaning $y = F_{sk}(x)$) or 0.

We require the following properties on VRF.

Provability: if $(y, \pi) = \mathsf{Func}_{sk}(x)$ then $\mathsf{V}_{pk}(x, y, \pi) = 1$ (except with negligible probability), for all $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$, and $x \in \mathsf{Domain}(\lambda)$.

Uniqueness: There is no tuple $(x, y_1, y_2, \pi_1, \pi_2)$, except with negligible probability, satisfying

$$y_1 \neq y_2 \text{ and } \mathsf{V}_{pk}(x, y_1, \pi_1) = \mathsf{V}_{pk}(x, y_2, \pi_2) = 1,$$

for all pk, and $x \in \mathsf{Domain}(\lambda)$.

[1] This unique property is stronger than "deterministic", since it deals also with maliciously generated keys.

[2] We will be mainly interested in the case $\mathsf{Range}(\lambda, pk)$ is big enough, e.g., $\mathsf{Range}(\lambda, pk) = \{0,1\}^\lambda$ for $\lambda = 170$.

Pseudo-randomness: For all poly-time distinguisher $\mathcal{D} = (\mathcal{D}_1, \mathcal{D}_2)$, the following advantage

$$\mathbf{Adv}^{\mathrm{rand}}_{\mathsf{VRF}}(\mathcal{D}) = \left| \Pr\left[b' = b : \begin{array}{c} (pk, sk) \xleftarrow{\$} \mathsf{Gen}(1^\lambda), (x^*, \mathtt{st}) \leftarrow \mathcal{D}_0^{\mathsf{Func}_{sk}(\cdot)}(1^\lambda, pk) \\ y_0^* \leftarrow F_{sk}(x^*), y_1^* \xleftarrow{\$} \mathsf{Range}(\lambda), \\ b \xleftarrow{\$} \{0,1\}, b' \leftarrow \mathcal{D}_1^{\mathsf{Func}_{sk}(\cdot)}(y_b^*, \mathtt{st}) \end{array} \right] - \frac{1}{2} \right|,$$

where x^* is not queried to the oracle $\mathsf{Func}_{sk}(\cdot)$, is negligible in λ. Above, $\mathtt{st}$ stands for state.

Selectively convertible undeniable signatures. The scheme, denoted as SCUS, consists of the algorithms (KGen, USign, $\mathsf{Convert}$, Verify) and the protocols ($\mathsf{Confirm}$, $\mathsf{Disavowal}$) described as follows.

- $\mathsf{KGen}(1^\lambda)$: return the public key pk and the secret key (signing key) sk.
- $\mathsf{USign}_{sk}(m)$: return a signature σ on a message m. We sometimes require that it is deterministic so that a signature σ is valid on m if and only if $\sigma = \mathsf{USign}_{sk}(m)$. If the algorithm is probabilistic with randomness r, we sometimes write $\sigma = \mathsf{USign}_{sk}(m; r)$.
- $\mathsf{Convert}_{sk}(m, \sigma)$: release a converter cvt if (m, σ) is valid, and $\perp$ otherwise.
- $\mathsf{Verify}_{pk}(m, \sigma, cvt)$: return 1 (meaning (m, σ) is valid) or 0.
- $\mathsf{Confirm}$: This is a protocol between the signer and a verifier, on common input (pk, m, σ), the signer with sk proves that (m, σ) is a valid message-signature pair in zero-knowledge.
- $\mathsf{Disavowal}$: This is a protocol between the signer and a verifier, on common input (pk, m, σ), the signer with sk proves that (m, σ) is an invalid message-signature pair in zero-knowledge.

Definition 1 (Unforgeability of SCUS). *The scheme* SCUS *is unforgeable if for all poly-time adversary* $\mathcal{A}$*, the advantage*

$$\mathbf{Adv}^{\mathrm{forge}}_{\mathsf{SCUS}}(\mathcal{A}) = \Pr\left[(m^*, \sigma^*) \textit{ is valid } : \begin{array}{c} (pk, sk) \leftarrow \mathsf{KGen}(1^\lambda), \\ (m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{USign}_{sk}(\cdot), \mathsf{Convert}_{sk}(\cdot,\cdot), \mathcal{P}}(pk), \\ m^* \textit{ was not queried to } \mathsf{USign}_{sk}(\cdot) \end{array} \right]$$

is negligible in λ*. Above and below,* $\mathcal{P}$ *stands for the confirmation/disavowal oracle working as follows:* $\mathcal{A}$ *submits a message-signature pair of the form* (m, σ) *to* $\mathcal{P}$*, which first checks the validity of* (m, σ)*. If it is a valid pair, the oracle returns* 1 *and executes the confirmation protocol with* $\mathcal{A}$ *(acting as a cheating verifier). Otherwise, the oracle returns* 0 *and executes the disavowal protocol with* $\mathcal{A}$*.*

For strong unforgeability, the forged pair (m^*, σ^*) *is different from the pairs appeared at the oracle* $\mathsf{USign}_{sk}(\cdot)$*. (Yet* m^* *can be queried to* $\mathsf{USign}_{sk}(\cdot)$*, yielding for example* σ_**. In that case,* $\sigma^* \neq \sigma_*$*.).*

We note that all SCUS schemes in this paper meet the notion of strong unforgeability.

Definition 2 (Invisibility of SCUS [9]). *The scheme* SCUS *has invisibility if for all poly-time distinguisher $\mathcal{D} = (\mathcal{D}_0, \mathcal{D}_1)$, the advantage*

$$\mathbf{Adv}^{\text{inv}}_{\mathsf{SCUS}}(\mathcal{D}) =$$

$$\left| \Pr \left[b' = b : \begin{array}{l} (pk, sk) \leftarrow \mathsf{KGen}(1^\lambda), \\ (m^*, \mathtt{st}) \leftarrow \mathcal{D}_0^{\mathsf{USign}_{sk}(\cdot), \mathsf{Convert}_{sk}(\cdot,\cdot), \mathcal{P}}(pk), \\ \sigma_0^* \leftarrow \mathsf{USign}_{sk}(m^*), \sigma_1^* \xleftarrow{\$} \mathsf{SigSpace}(\lambda), \\ b' \leftarrow \mathcal{D}_1^{\mathsf{USign}_{sk}(\cdot), \mathsf{Convert}_{sk}(\cdot,\cdot), \mathcal{P}}(\sigma^* \overset{\text{def}}{=} \sigma_b^*, \mathtt{st}) \end{array} \right] - 1/2 \right|$$

is negligible in λ, where $\mathsf{SigSpace}(\lambda)$ *is the signature space of the scheme. Also, there are some natural restrictions on $\mathcal{D}_1$'s queries: no confirmation/disavowal query and conversion query (m^*, σ^*), and no signing query m^* (needed only if* USign *is deterministic), are allowed.*

3 Zero-Knowledge Protocols for Generalized DDH and Non-DDH

Protocols from q-oneway Homomorphism [6]. Let Dom and Rng be finite Abelian groups, and $f : \mathsf{Dom} \to \mathsf{Rng}$ be a group homomorphism. Following Cramer, Damgård, and MacKenzie, we say a one-way function f is q-oneway for a fixed prime q, if given f and b in f's range, one can efficiently find $a \in \mathsf{Dom}$ such that $f(a) = b^q$. From such f, it is shown in [6] how to build a 4-move zero-knowledge (ZK) protocol proving the knowledge of f's pre-image. Namely, for public x, the prover shows the knowledge of ω such that $f(\omega) = x$. The zero-knowledge protocol is based on the following Σ-protocol.

Σ-protocol proving $f(\omega) = x$

1. The prover chooses $\omega' \xleftarrow{\$} \mathsf{Dom}$, and sends $x' = f(\omega')$ to the verifier.
2. The verifier sends back a random challenge $c \in \{0, \ldots, q-1\}$.
3. The prover returns $\omega'' = \omega' + c\omega$ to the verifier who checks $f(\omega'') = x'x^c$.

The well-known Schnorr protocol [23] becomes a special case of the above, when considering $f : \mathbb{Z}_q \to \mathcal{H}$, and $f(\omega) = h^\omega$, where the order $|\mathcal{H}| = q$ and h is a generator of $\mathcal{H}$. It is easy to see that f is an oneway homomorphism. For $y \in \mathcal{H}$, we have $y^q = 1$, thus $f(0) = y^q$, and hence f is q-oneway.

ZK Protocol for Generalized DDH. Consider two generators $g, h \in \mathcal{H}$. The elements $X_i = h^{x_i}$ for $1 \le i \le n$, and Y are given in public. The prover with secrets x_i wants to show in ZK that $Y = g^{\prod_{i=1}^n x_i}$. When $g = h$ and $n = 1$, this is exactly the Schnorr protocol.

Let $y_j = \prod_{i=1}^j x_i$ for $1 \le j \le n$. For the following q-oneway homomorphism

$$f(y_1, \ldots, y_n) = \left(h^{y_1}, h^{y_2} X_2^{-y_1}, \ldots, h^{y_n} X_n^{-y_{n-1}}, g^{y_n}\right),$$

our prover proves that he knows $(y_1, \ldots, y_n)$ such that $f(y_1, \ldots, y_n) = (X_1, 1, \ldots, 1, Y)$ by using the above Σ-protocol. The protocol is a 3-move ZK protocol against honest verifier. It can be transformed to a 4-move ZK protocol against any verifier.

(Completeness) It holds that $h^{y_j} X_j^{-y_{j-1}} = 1$ because $h^{y_j} = X_j^{y_{j-1}}$ for all $2 \le j \le n$. (Soundness) Note that $h^{y_1} = X_1$ implies $y_1 = x_1$, and $h^{y_2} X_2^{-y_1} = 1$ implies $y_2 = y_1 x_2 = x_1 x_2$, and so on, so that $y_n = \prod_{i=1}^{n} x_i$ as required.

Neff [21], with other techniques, also gave a protocol called iterated logarithmic multiplication protocol realizing the above task when $g = h$. Ours, described above, is more compact and general.

ZK Protocol for Generalized Non-DDH. Given $h, X_i = h^{x_i}, Y \in \mathcal{H}$, and now the prover wants to show that $Y \neq h^{\prod_{i=1}^{n} x_i}$. We call this *generalized non-DDH protocol*, which will be used for disavowal in later SCUS schemes. Employing a trick of Camenisch and Shoup [4], the prover takes $x_0 \xleftarrow{\$} \mathbb{Z}_q$ and sends $T = \left(Y^{-1} h^{\prod_{i=1}^{n} x_i}\right)^{x_0}$ to the verifier who checks $T \neq 1$. Let $y_j = \prod_{i=0}^{j} x_i$ so that $T = Y^{-x_0} h^{y_n}$. Note that $h^{y_1} X_1^{-x_0} = 1$ and $h^{y_j} X_j^{-y_{j-1}} = 1$ for $2 \le i \le n$. We then consider the following q-oneway homomorphism

$$f(x_0, y_1, \ldots, y_n) = \left(h^{y_1} X_1^{-x_0}, h^{y_2} X_2^{-y_1}, \ldots, h^{y_n} X_n^{-y_{n-1}}, Y^{-x_0} h^{y_n}\right),$$

so that the prover just needs to prove the knowledge of the pre-image satisfying $f(x_0, y_1, \ldots, y_n) = (1, \ldots, 1, T)$. The protocol is 3-move against honest verifier, and 4-move against any verifier.

Full zero-knowledgeness from the Σ-protocols is described in the full version [13].

4 Relation between SCUS and VRF

4.1 SCUS from VRF: A Generic Construction

Let $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Func}, \mathsf{V})$ be a VRF for $\mathsf{Func}_{sk}(\cdot) = \big(F_{sk}(\cdot), \pi_{sk}(\cdot)\big)$ where the range of $F_{sk}(\cdot)$ is considered as $\{0,1\}^{\kappa}$ ($\kappa \approx 170$ in later sections). We now construct a scheme $\mathsf{SCUS} = (\mathsf{KGen}, \mathsf{USign}, \mathsf{Convert}, \mathsf{Verify}, \mathsf{Confirm}, \mathsf{Disavowal})$ described as follows.

The Construction of SCUS**:**

- $\mathsf{KGen}(1^{\lambda})$: Return $(pk, sk) \leftarrow \mathsf{Gen}(1^{\lambda})$.
- $\mathsf{USign}_{sk}(m)$: Return $\sigma = F_{sk}(m)$ as the undeniable signature.
- $\mathsf{Convert}_{sk}(m, \sigma)$: Return $\pi = \pi_{sk}(m)$ as the converter if $\mathsf{V}_{pk}(m, \sigma, \pi) = 1$, else return $\perp$.
- $\mathsf{Verify}_{pk}(m, \sigma, \pi)$: Return $\mathsf{V}_{pk}(m, \sigma, \pi)$.
- $\mathsf{Confirm}$: The common input is (pk, m, σ). With private input sk, the signer proves in zero-knowledge that $\sigma = F_{sk}(m)$.
- $\mathsf{Disavowal}$: The common input is (pk, m, σ). With private input sk, the signer proves in zero-knowledge that $\sigma \neq F_{sk}(m)$.

In general, the $\mathsf{Confirm}$ and $\mathsf{Disavowal}$ protocols can be realized by general techniques for zero-knowledge, but not very efficiently. Later, we will present efficient protocols for specific cases. The proofs for below results in [13] assume that the confirmation and disavowal protocols are run sequentially. (However, it is not hard to extend them to handle concurrent or arbitrary executions of the protocols.)

Theorem 3. *The* SCUS *construction above satisfies unforgeability.*

Theorem 4. *The* SCUS *construction above has invisibility. In particular, if there is a poly-time $\mathcal{D}_{\mathrm{scus}}$ against* SCUS*, then there are poly-time $\mathcal{D}_{\mathrm{vrf}}$ and $\mathcal{A}$ satisfying*

$$\mathbf{Adv}^{\mathrm{inv}}_{\mathsf{SCUS}}(\mathcal{D}_{\mathrm{scus}}) \leq \mathbf{Adv}^{\mathrm{rand}}_{\mathsf{VRF}}(\mathcal{D}_{\mathrm{vrf}}) + \mathbf{Adv}^{\mathrm{forge}}_{\mathsf{SCUS}}(\mathcal{A})$$

4.2 Concrete Instantiations

In this section, we will use a pairing group $\mathbb{PG} = (G, G_T, g, q, e)$, where g is a generator of G, and the order $|G| = |G_T| = q$ for prime q, and $e : G \times G \to G_T$.

SCUS from the VRF of Dodis and Yampolskiy [7]. The VRF works on a pairing group $\mathbb{PG} = (G, G_T, g, q, e)$, and is pseudo-random under the $\mathfrak{q}$-DBDHI assumption [20]. The secret key is $sk = s \in \mathbb{Z}_q$ and the public key is $pk = g^s$. On input $m \in \{0,1\}^{\ell_m} \subset \mathbb{Z}_q$ (in which 2^{ℓ_m} must be polynomial), define $F_s(m) = e(g,g)^{1/(m+s)}$, and $\pi = \pi_s(m) = g^{1/(m+s)}$. The function $F_s(\cdot)$ serves as a random function, and $\pi_s(\cdot)$ as the proof of its correctness. To ensure that $y(= F_s(m))$ was computed correctly, one checks

$$e(\pi_m, g^m \cdot pk) \stackrel{?}{=} e(g,g) \text{ and } e(\pi_m, g) \stackrel{?}{=} y.$$

We now show how to turn the above VRF into a SCUS scheme. The public and secret keys are the same as above. The signature on m is $\sigma = F_s(m)$, which is pseudo-random, so invisible, under the $\mathfrak{q}$-DBDHI assumption. For selective conversion on (m, σ), release $\pi_s(m)$.

Confirm: To show that (m, σ) is valid in the confirmation protocol, the signer with secret s proves that

$$\sigma = e(g,g)^{1/(m+s)} \iff \sigma^{m+s} = e(g,g) \iff \sigma^s = \sigma^{-m} e(g,g) \in G_T.$$

Thus the signer just proves $\big(e(g,g), e(pk,g), \sigma, \sigma^{-m}e(g,g)\big)$ is a DDH tuple in G_T, which can be realized in 4 moves as follows. Using ideas in Sect. 3, the q-oneway homomorphism is $f(s) = (e(g,g)^s, \sigma^s)$, and the signer needs to prove for secret $s \in \mathbb{Z}_q$ that $f(s) = (e(pk,g), \sigma^{-m}e(g,g))$.

Disavowal: The signer needs to prove that the value $\sigma^{m+s} \cdot e(g,g) \neq 1$. The signer sets $U = (\sigma^{m+s} \cdot e(g,g))^u \in G_T$ for random $u \in \mathbb{Z}_q$ and sends U to the verifier who checks $U \neq 1$. With secrets u, s, the signer shows in zero-knowledge that

$$U = \big(\sigma^{m+s} \cdot e(g,g)\big)^u.$$

Imagine $v = us$, then what must be proven becomes, for secrets (u, v),

$$U = (\sigma^m)^u \cdot \sigma^v \cdot e(g,g)^u \quad \wedge \quad e(g,g)^v \cdot e(g,pk)^{-u} = 1,$$

which can be again realized in 4 moves by the following q-oneway homomorphism (mentioned in Sect.3)

$$\begin{array}{ll} f\colon \mathbb{Z}_q \times \mathbb{Z}_q \longrightarrow G_T \times G_T \\ \quad (u,v) \quad \longmapsto \big((\sigma^m)^u \cdot \sigma^v \cdot e(g,g)^u, e(g,g)^v \cdot e(g,pk)^{-u}\big) \end{array}$$

Then, the Disavowal protocol just becomes proving $f(u,v) = (U,1)$ for $U \neq 1$.

It is worth noting that security results given in [7] hold only if the message length ℓ_m is logarithmic in the security parameter. However, other SCUS schemes below will not suffer from the shortcoming.

SCUS from the VRF of Hohenberger and Waters [10]. The VRF also works on a pairing group $\mathbb{PG} = (G, G_T, g, q, e)$, and is pseudo-random under the q-DDHE assumption. The secret key is $sk = (\tilde{u}, u_0, \ldots, u_n)$ for $n \approx 160$ typically (or more, depending on the output of a hash function), and the public key is $pk = (\mathbb{PG}, h, \tilde{U}, U_0, \ldots, U_n)$ where $\tilde{U} = g^{\tilde{u}}, U_0 = g^{u_0}, \ldots, U_n = g^{u_n}$. The function $F_{sk}(m \in \{0,1\}^n)$ is

$$\sigma = F_{sk}(m) = e(g,h)^{\tilde{u}u_0 \prod_{i=1}^{n} u_i^{m[i]}},$$

for $m = m[1] \ldots m[n]$, which serves as the undeniable signature on m. (For arbitrary $m \in \{0,1\}^*$, one can apply a hash function first to get an n-bit string, so that $n = 160$ typically.)

The proof of correctness $\pi_{sk}(m)$, which serves as the converter in the SCUS, consists of $\pi = (\pi_1, \ldots, \pi_n, \pi_0)$, in which

$$\pi_k = g^{\tilde{u} \prod_{i=1}^{k} u_i^{m[i]}} (1 \leq k \leq n), \text{ and } \pi_0 = g^{\tilde{u}u_0 \prod_{i=1}^{n} u_i^{m[i]}}.$$

To verify, $\mathsf{Verify}_{pk}(m, \sigma, \pi)$ of the SCUS works step-by-step, checking

$$e(\pi_1, g) \stackrel{?}{=} \begin{cases} e(\tilde{U}, g) & \text{if } m[1] = 0 \\ e(\tilde{U}, U_1) & \text{if } m[1] = 1 \end{cases}$$

and for $2 \leq i \leq n$,

$$e(\pi_i, g) \stackrel{?}{=} \begin{cases} e(\pi_{i-1}, g) & \text{if } m[i] = 0 \\ e(\pi_{i-1}, U_i) & \text{if } m[i] = 1 \end{cases}$$

and finally

$$e(\pi_0, g) \stackrel{?}{=} e(\pi_n, U_0), \text{ and } e(\pi_0, h) \stackrel{?}{=} \sigma,$$

and return 1 if and only if all checks pass.

Confirm: On common input (pk, m, σ), the signer with secret $sk = (\tilde{u}, u_0, \ldots, u_n)$ proves in zero-knowledge

$$\sigma = e(g,h)^{\tilde{u}u_0 \prod_{i=1}^{n} u_i^{m[i]}},$$

which can be implemented by the generalized DDH protocol (see Sect.3) on G_T, with $e(\tilde{U}, h)$, $e(U_0, h)$, …, $e(U_n, h)$ and σ as public elements.

Disavowal: On common input (pk, m, σ), the signer with secret $sk = (\tilde{u}, u_0, \ldots, u_n)$ proves in zero-knowledge

$$\sigma \neq e(g,h)^{\tilde{u}u_0 \prod_{i=1}^{n} u_i^{m[i]}},$$

which can be implemented using the generalized non-DDH protocol in Sect. 3.

4.3 VRF from Unique SCUS

Consider a deterministic SCUS consisting of the algorithms (KGen, USign, Convert, Verify) and two protocols for confirmation and disavowal of signatures. (These protocols are not used in the below construction of VRF.) The signing space is $\{0,1\}^*$, and the signature space is $\mathsf{SigSpace}(\lambda)$.

The Construction of VRF from Deterministic SCUS:

- $\mathsf{Gen}(1^\lambda)$: run $\mathsf{KGen}(1^\lambda)$ of SCUS to obtain (pk, sk).
- $\mathsf{Func}_{sk}(x)$: return $y = \mathsf{USign}_{sk}(x)$, and the proof $\pi = \mathsf{Convert}_{sk}\big(x, y\big)$. By construction, $F_{sk}(\cdot) : \{0,1\}^* \to \mathsf{SigSpace}(\lambda)$.
- $\mathsf{V}_{pk}(x, y, \pi)$: return $\mathsf{Verify}_{pk}(x, y, \pi)$.

We now check the properties of the VRF. Provability is easy: if $(y, \pi) = \mathsf{Func}_{sk}(x)$, then y is a valid signature on x, and π is the converter, so $\mathsf{V}_{pk}(x, y, \pi) = 1$.

Uniqueness holds if we require USign produces only one valid signature on each message. Then, if $\mathsf{V}_{pk}(x, y_1, \pi_1) = \mathsf{V}_{pk}(x, y_2, \pi_2) = 1$ then y_1, y_2 are valid signatures on the same message x. Since there is only one valid signature on each message, we have $y_1 = y_2$.

Pseudo-randomness is ensured by the following theorem.

Theorem 5. *If* SCUS *has the property of invisibility, then* VRF *has the property of pseudo-randomness. Moreover, if $\mathcal{D}_{\text{vrf}}$ is a distinguisher of* VRF*, then there exists $\mathcal{D}_{\text{scus}}$ against* SCUS *such that*

$$\mathbf{Adv}^{\text{rand}}_{\mathsf{VRF}}(\mathcal{D}_{\text{vrf}}) \leq \mathbf{Adv}^{\text{inv}}_{\mathsf{SCUS}}(\mathcal{D}_{\text{scus}}),$$
$$\mathbf{T}(\mathcal{D}_{\text{vrf}}) \approx \mathbf{T}(\mathcal{D}_{\text{scus}}),$$

where $\mathbf{T}(\cdot)$ expresses the running time.

5 A New Probabilistic SCUS with Neat Converters and Signatures

In Sect. 4.2, we have seen the deterministic SCUS resulting from the VRF of Dodis and Yampolskiy [7] with small signing space. In this section, we aim at increasing the signing space to arbitrary one, while keeping the converters and signatures as short as possible. We will use the result of Boneh and Boyen [1] to build a probabilistic SCUS scheme called $\mathsf{SCUS_{BB}}$ in this section.

Let us provide some intuition first. Recall that a Boneh-Boyen signature is of two elements $\left(r, \pi = g_1^{1/(x+m+ry)}\right)$ for random $r \in \mathbb{Z}_q$, secrets $x, y \in \mathbb{Z}_q$ and message $m \in \mathbb{Z}_q$ (for $m \in \{0,1\}^*$, just applying a collision-resistant hash to $\mathbb{Z}_q$). The element π will serve as the converter[3]. To achieve invisibility, we hide π in G_T, namely apply the pairing to compute $e(\pi, g_2)$, which will, together with r, be the undeniable signature. The confirmation and disavowal protocols can be efficiently designed thanks to the algebraic structure of the construction.

We now proceed with the concrete description of the scheme, denoted as $\mathsf{SCUS_{BB}}$. To be compatible with [1], we will be more general than previous schemes by considering the pairing $e : G_1 \times G_2 \to G_T$ in which $|G_1| = |G_2| = |G_T| = q$, yet G_1 may be different from G_2.

The scheme $\mathsf{SCUS_{BB}}$:

KGen: Generate the generators g_1 for G_1, and g_2 for G_2. Pick the secret key $sk = (x, y) \xleftarrow{\$} \mathbb{Z}_q^2$. The public key $pk = (u, v, g_1, g_2)$ for $u = g_2^x, v = g_2^y$.

$\mathsf{USign}_{sk}(m)$: For a message $m \in \mathbb{Z}_q$, pick $r \xleftarrow{\$} \mathbb{Z}_q$, and return the undeniable signature $\sigma = \left(r, e(g_1, g_2)^{\frac{1}{x+m+ry}}\right)$.

$\mathsf{Convert}_{sk}(m, \sigma)$: Parse $\sigma = (r, \rho) \in \mathbb{Z}_q \times G_T$. If $\rho = e(g_1, g_2)^{\frac{1}{x+m+ry}}$ then return $\pi = g_1^{\frac{1}{x+m+ry}}$ as the converter.

$\mathsf{Verify}_{pk}(m, \sigma, \pi)$: Let $\sigma = (r, \rho)$. Return 1 iff $\rho = e(\pi, g_2)$ and $e(\pi, u g_2^m v^r) = e(g_1, g_2)$.

$\mathsf{Confirm}$: On common input pk, m, and $\sigma = (r, \rho)$, the signer shows in ZK that $\rho = e(g_1, g_2)^{\frac{1}{x+m+ry}}$, namely $\rho^{x+m+ry} = e(g_1, g_2)$, or equivalently $\rho^x (\rho^r)^y = e(g_1, g_2)\rho^{-m}$.
Consider the q-oneway homomorphism $f(x, y) = (\rho^x \cdot (\rho^r)^y, g_2^x, g_2^y)$. The protocol is equivalent to proving $f(x, y) = (e(g_1, g_2)\rho^{-m}, u, v)$ for secret (x, y), which achieves 4 moves with full zero-knowledge.

$\mathsf{Disavowal}$: The notation is as above, and the signer proves in ZK that $\rho^x (\rho^r)^y \neq e(g_1, g_2)\rho^{-m}$, namely $\rho^x (\rho^r)^y \cdot \rho^m e(g_1, g_2)^{-1} \neq 1$. The prover takes $t \xleftarrow{\$} \mathbb{Z}_q$ and sends

$$T = \left(\rho^x (\rho^r)^y \cdot \rho^m e(g_1, g_2)^{-1}\right)^t,$$

to the verifier who checks $T \neq 1$. Let $x' = xt$ and $y' = yt$, then $T = \rho^{x'} (\rho^r)^{y'} \left(\rho^m e(g_1, g_2)^{-1}\right)^t$, and $g_2^{x'} u^{-t} = 1$, $g_2^{y'} v^{-t} = 1$. The q-oneway homomorphism is as follows

$$f(x', y', t) = \left(\rho^{x'} (\rho^r)^{y'} \left(\rho^m e(g_1, g_2)^{-1}\right)^t, g_2^{x'} u^{-t}, g_2^{y'} v^{-t}\right),$$

so that the protocol becomes proving $f(x', y', t) = (T, 1, 1)$ for published $T \neq 1$, which is done in 4 moves.

[3] Considering groups *without* pairings so the DDH assumption holds, Laguillaumie and Vergnaud [15] observed that π is pseudo-random, on which they built an undeniable scheme. The scheme, however, is not convertible. More schemes based on the Boneh-Boyen signature scheme are in Vergnaud's PhD thesis [26].

To prove security of $\mathsf{SCUS}_{\mathsf{BB}}$, we need the following assumptions, which are variants of the strong Diffie-Hellman assumption [1].

Computational Bilinear Strong Diffie-Hellman (CBSDH). Given $g_1 \in G_1$, $g_2 \in G_2$, and $g_1^x, \ldots, g_1^{x^\ell}$, g_2^x, it's hard to compute $(c, e(g_1, g_2)^{\frac{1}{x+c}})$ for some $c \in \mathbb{Z}_q \setminus \{-x\}$.

Decisional Bilinear Strong Diffie-Hellman (DBSDH). Given $g_1 \in G_1$, $g_2 \in G_2$, and $g_1^x, \ldots, g_1^{x^\ell}$, g_2^x, and random $c \in \mathbb{Z}_q \setminus \{-x\}$, it's hard to distinguish $e(g_1, g_2)^{\frac{1}{x+c}}$ from a random element in G_T.

The DBSDH assumption can be shown equivalent to the decisional bilinear Diffie-Hellman inversion assumption [20], since c is fixed (see [1, Sect.3.3]). These assumptions can be evaluated in the generic group model [25].

Theorem 6. *The* $\mathsf{SCUS}_{\mathsf{BB}}$ *scheme is strongly unforgeable under the CBSDH assumption.*

Theorem 7. *The* $\mathsf{SCUS}_{\mathsf{BB}}$ *scheme is invisible under the CBSDH assumption and the DBSDH assumption.*

Acknowledgment. We thank Michel Abdalla for being the shepherd, and the anonymous reviewers for fruitful comments.

References

1. Boneh, D., Boyen, X.: Short signatures without random oracles and the SDH assumption in bilinear groups. J. Cryptology 21(2), 149–177 (2008)
2. Boyar, J., Chaum, D., Damgård, I.B., Pedersen, T.P.: Convertible Undeniable Signatures. In: Menezes, A., Vanstone, S.A. (eds.) CRYPTO 1990. LNCS, vol. 537, pp. 189–205. Springer, Heidelberg (1991)
3. Boyd, C., Foo, E.: Off-Line Fair Payment Protocols Using Convertible Signatures. In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 271–285. Springer, Heidelberg (1998)
4. Camenisch, J.L., Shoup, V.: Practical Verifiable Encryption and Decryption of Discrete Logarithms. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 126–144. Springer, Heidelberg (2003)
5. Chaum, D., van Antwerpen, H.: Undeniable Signatures. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 212–216. Springer, Heidelberg (1990)
6. Cramer, R., Damgård, I., MacKenzie, P.D.: Efficient Zero-Knowledge Proofs of Knowledge without Intractability Assumptions. In: Imai, H., Zheng, Y. (eds.) PKC 2000. LNCS, vol. 1751, pp. 354–373. Springer, Heidelberg (2000)
7. Dodis, Y., Yampolskiy, A.: A Verifiable Random Function with Short Proofs and Keys. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 416–431. Springer, Heidelberg (2005)
8. El Aimani, L.: Anonymity from Public Key Encryption to Undeniable Signatures. In: Preneel, B. (ed.) AFRICACRYPT 2009. LNCS, vol. 5580, pp. 217–234. Springer, Heidelberg (2009)

9. Galbraith, S.D., Mao, W.: Invisibility and Anonymity of Undeniable and Confirmer Signatures. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 80–97. Springer, Heidelberg (2003)
10. Hohenberger, S., Waters, B.: Constructing Verifiable Random Functions with Large Input Spaces. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 656–672. Springer, Heidelberg (2010)
11. Jarecki, S.: Handcuffing Big Brother: an Abuse-Resilient Transaction Escrow Scheme (Extended Abstract). In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 590–608. Springer, Heidelberg (2004)
12. Kikuchi, R., Phong, L.T., Ogata, W.: A Framework for Constructing Convertible Undeniable Signatures. In: Heng, S.-H., Kurosawa, K. (eds.) ProvSec 2010. LNCS, vol. 6402, pp. 70–86. Springer, Heidelberg (2010)
13. Kurosawa, K., Nojima, R., Phong, L.T.: Relation between verifiable random functions and convertible undeniable signatures, and new constructions. Cryptology ePrint Archive, Report 2012/259, Full version of this paper (2012), `http://eprint.iacr.org/`
14. Kurosawa, K., Takagi, T.: New Approach for Selectively Convertible Undeniable Signature Schemes. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 428–443. Springer, Heidelberg (2006)
15. Laguillaumie, F., Vergnaud, D.: Short Undeniable Signatures Without Random Oracles: The Missing Link. In: Maitra, S., Veni Madhavan, C.E., Venkatesan, R. (eds.) INDOCRYPT 2005. LNCS, vol. 3797, pp. 283–296. Springer, Heidelberg (2005)
16. Liskov, M.: Updatable Zero-Knowledge Databases. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 174–198. Springer, Heidelberg (2005)
17. Micali, S., Rabin, M.O., Vadhan, S.P.: Verifiable random functions. In: FOCS, pp. 120–130 (1999)
18. Micali, S., Reyzin, L.: Soundness in the Public-Key Model. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 542–565. Springer, Heidelberg (2001)
19. Micali, S., Rivest, R.L.: Micropayments Revisited. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 149–163. Springer, Heidelberg (2002)
20. Mitsunari, S., Sakai, R., Kasahara, M.: A new traitor tracing. IEICE Trans. Fundamentals E85-A(2), 481–484 (2002)
21. Neff, C.A.: A verifiable secret shuffle and its application to e-voting. In: ACM Conference on Computer and Communications Security, pp. 116–125 (2001)
22. Phong, L.T., Kurosawa, K., Ogata, W.: Provably Secure Convertible Undeniable Signatures with Unambiguity. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 291–308. Springer, Heidelberg (2010)
23. Schnorr, C.-P.: Efficient signature generation by smart cards. J. Cryptology 4(3), 161–174 (1991)
24. Schuldt, J.C.N., Matsuura, K.: An Efficient Convertible Undeniable Signature Scheme with Delegatable Verification. In: Kwak, J., Deng, R.H., Won, Y., Wang, G. (eds.) ISPEC 2010. LNCS, vol. 6047, pp. 276–293. Springer, Heidelberg (2010)
25. Shoup, V.: Lower Bounds for Discrete Logarithms and Related Problems. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 256–266. Springer, Heidelberg (1997)
26. Vergnaud, D.: Approximation diophantienne et courbes elliptiques. Protocoles asymétriques d'authentification non-transférable. PhD Thesis, `http://www.di.ens.fr/~vergnaud/TheseVergnaud.ps.gz`

Generalized First Pre-image Tractable Random Oracle Model and Signature Schemes

Xiao Tan and Duncan S. Wong

Department of Computer Science
City University of Hong Kong
xiaotan4@gapps.cityu.edu.hk, duncan@cityu.edu.hk

Abstract. Weakened Random Oracle Models (WROMs) are variants of the Random Oracle Model (ROM) under some weakened collision resistance assumptions. Cryptographic schemes proven secure in WROMs can ensure security even when the underlying random oracles are susceptible to certain extent of collision attacks, second pre-image attacks, or first pre-image attacks. In this paper, we show that a WROM variant called FPT-ROM (First Pre-Image Tractable ROM) can further be weakened to a Generalized FPT-ROM which can capture more practical attacks, for example, the chosen prefix collision attack by Stevens *et al.* (CRYPTO 2009). This type of attacks has never been captured by any existing WROMs. Achieving security against FPT-ROM has been known as one of the most challenging problems in constructing cryptographic schemes in WROMs. In the second part of this paper, we propose a generic transformation which converts a large class of signature schemes secure in ROM to a class of variants, which can be proven secure in all the WROMs, including our newly proposed Generalized FPT-ROM. The transformation does not increase the signature size, and it can apply to many practical and highly efficient signature schemes such as the Full-Domain Hash signature, Schnorr signature, and many others.

Keywords: Random Oracle Model (ROM), Weakened ROM, First Pre-Image Tractable ROM.

1 Introduction

The Random Oracle Model [2] (ROM) enables us to construct elegant and efficient cryptographic schemes with provable security. The security of most schemes under ROM relies on some well-studied, number-theoretic complexity assumptions with tight security proofs. In ROM, all the parties are supposed to have oracle access to some truly random functions, called "random oracles". When a random oracle comes to practice, it is usually instantiated by a cryptographic hash function. However, recent attacks on hash functions [35,34] revealed that it might not be as hard as we originally thought on breaking a hash function, for example, finding a collision. This raises a question that instead of giving up ROM altogether for a stronger model (e.g. the standard model), whether we can build highly efficient cryptographic schemes using weak hash functions, that is, when the random oracle assumption is weakened in some extent. To answer this question, a useful observation is that the absence of collision resistance property does not always result in breaking the security proofs of some schemes in ROM. Instead, the programmability, uniformity and unpredictability of the random oracles are the key factors of detraining whether a security proof still holds or not if weak hash functions are used to instantiate random oracles. From the observation above, Liskov introduced a set of ideal weak compression functions [23], and some of which were later formalized by Numayama *et al.* [27] as *Weakened Random Oracle Models* (WROMs).

In WROMs, participants not only have access to random oracles, but also some additional oracles that can extract collisions or pre-images of the random oracles. More precisely, currently there are three kinds of WROMs: (1) Collision Tractable Random Oracle Model CT-ROM, (2) Second Pre-image Tractable Random Oracle Model SPT-ROM, and (3) First Pre-image Tractable Random Oracle Model FPT-ROM, where each random oracle, say, H is associated with an additional oracle CO_H, SPO_H, or FPO_H, respectively:

- $\mathsf{CO}_H()$: it uniformly returns a collision pair (x, x') of H, such that $x \neq x'$ and $H(x) = H(x')$.
- $\mathsf{SPO}_H(x, y)$: given an image y and a pre-image x of H such that $H(x) = y$, it uniformly returns a second pre-image $x' \neq x$, such that $H(x') = y$.
- $\mathsf{FPO}_H(y)$: given an image y of H, it uniformly returns a pre-image x such that $H(x) = y$.

In the literature, many of the hash-related attacks are to find random collisions, which can be captured in CT-ROM. There are also some research results regarding second pre-image attacks and first pre-image attacks, e.g. attacks on reduced-round MD5 or SHA-1 [7], or on historical hash functions like MD4 [22]. These two types of attacks are covered in SPT-ROM and FPT-ROM, respectively.

Nevertheless, a few other related attacks are yet to be captured and studied in WROMs. For instance, the *chosen prefix collision attack*, described by Stevens *et al.* in [32,33]. The attack is to find non-random collisions of MD5, and is yet to be captured in WROMs. In this attack, an adversary can arbitrarily choose a pair of prefixes (P, P') and produce a target collision ($P \| S$, $P' \| S'$) by making roughly 2^{49} calls to the compression function. The computation complexity for finding a collision can be reduced to 2^{16} if the prefixes are identical, i.e. $P = P'$. This type of attacks could cause great impact on the security of various applications, e.g. a rogue Certificate Authority could be able to generate two MD5-based X.509 certificates, such that the certificates have identical signatures, but the name fields and public keys may be different.

A reducible separation of WROMs for signature schemes is given in [27], and it can be shown that $S/\mathsf{ROM} \Leftarrow S/\mathsf{CT\text{-}ROM} \Leftarrow S/\mathsf{SPT\text{-}ROM} \Leftarrow S/\mathsf{FPT\text{-}ROM}$, where S denotes the security notion of existential unforgeability of signatures against chosen message attack (EUF-CMA). Later, Kawachi *et al.* [21] gave a reducible separation of WROMs for the IND-CCA2 security of public key encryption schemes as well. These results show that FPT-ROM is the most weakened variant of ROM in existing WROMs.

Several encryption and signature schemes have been proposed in specific WROMs [21,27]. One of the most challenging problems is to construct FPT-ROM-secure schemes. Since many cryptographic schemes are proven secure in ROM, it is of great significance to generically transform ROM-secure schemes to WROMs-secure ones without losing efficiency. In the literature, there are only two works [28,25] on generic transformation to FPT-ROM secure schemes.

Our Results. As mentioned above, the chosen prefix collision attack is yet to be captured in WROMs. In this paper, we first generalize FPT-ROM, which is currently the most weakened variant of ROM, such that our new *Generalized* model, namely Generalized FPT-ROM or GFPT-ROM, in short, allows an adversary to extract polynomially many pre-images with *adaptively chosen prefixes*. We then show that GFPT-ROM is strictly weaker (in terms of the random oracle assumption level) than existing WROMs. The goal of our model is to further reduce the gap between ROM security and the real-world security. In the second part of this paper, we focus on building a generic transformation for a large class of signature schemes to a version secure in GFPT-ROM, and also in existing WROMs. Another merit of our generic transformation is that it does not incur any expansion of signature size.

The rest of the paper is organized as follows. After the review of related work, in Sec. 2, we propose our new Generalized FPT-ROM, and show that it captures more attacks including the chosen prefix collision attack. In Sec. 3 and 4, we propose a generic transformation for signature schemes, and show how to apply it onto an RSA Full Domain Hash (FDH) signature for making it secure in the Generalized FPT-ROM. We then compare our results with the existing ones and conclude the paper in Sec. 5.

1.1 Related Work

Signature schemes with provable security roughly fall into two classes: Hash-and-Sign Signatures [29,31,2,13,10,6,5,14] and Tree-based Signatures [15,1,26,30,11]. Tree-based signatures have the following restrictions [11]: (1) there is an upper bound on the number of messages that could be signed; and (2) the signature size depends on the depth of the underlying tree. On the other side, Hash-and-Sign Signatures are more efficient with shorter signatures and public keys. Besides, a hash function can work as a domain extender which allows signing on arbitrarily long messages. Hash-and-Sign Signatures can also be classified into two categories: signatures in ROM [29,31,2,6,14] and signatures that require strong complexity assumptions [13,10,5]. One exception is the signatures [18] applying programmable hash functions [19], e.g. Waters' signature [36]. However, in this type of schemes, either public keys or signatures are of linear size to the message length. For the signatures based on strong complexity assumptions, they may not provide high confidence in security, since that a given problem instance generally has an exponential number of solutions. By comparison, ROM-secure signatures usually have smaller signature and key sizes with high computational efficiency, and relying on standard complexity assumptions.

A widely applied approach to weaken the random oracle assumption is to put a *salt*, or say randomness, in hashing. By following this approach, Bellare and Rogaway [4] proposed using Target Collision Resistant (TCR) hash functions, i.e. some keyed compression function $H : (K, m) \mapsto H_K(m)$, to replace conventional hash functions in Hash-and-Sign signature schemes. Later, Halevi and Krawczyk [16] proposed an enhanced primitive eTCR which helps avoid signing the compression key K. The drawback of using TCR or eTCR is that K has to be included in a signature, resulting in the increase of signature size by $|K|$. Other signatures applying the salted hashing, such as PFDH [9] and its variants [27], also have the similar problem in size expansion. As a partial solution, Mironv [24] suggested to recycle the randomness in a signature scheme with a randomized pre-computation phase (SRP). Combining the techniques in [16] and [24], Pasini and Vaudenay [28] proposed a generic method to construct strongly unforgeable Hash-and-Sign signature schemes in $\mathsf{FPT\text{-}ROM}$ from weakly secure signatures. More precisely, a signature is generated as $\mathsf{sign}(G(H_K(m)))$, where H is an eTCR hash function, G is modeled as a first pre-image tractable oracle, and sign is the signing algorithm of an SRP which is only secure against known message attacks. However, the randomness generated by SRP's probabilistic pre-computation algorithm needs to have a large enough entropy. This restriction implies that the transformation only works for probabilistic signature schemes, but not for deterministic signatures such as FDH signature. Naito *et al.* [25] proposed an approach to construct $\mathsf{FPT\text{-}ROM}$-secure cryptographic schemes by padding the random oracles with some constant c. In Sec. 2.2, we show that this cannot provide security in the $\mathsf{Generalized\ FPT\text{-}ROM}$.

One of the compression function variants proposed by Liskov [23], namely (two-way) partially-specified pre-image tractable compression function, has a similar idea to ours. It requires that for a compression function $H : (K, m) \mapsto H_K(m)$, there is an additional oracle which on input (K, w) (resp. (m, w)), it returns m (resp. K) such that $H(K, m) = w$. In the rest of the paper, we will see that we have a similar idea when defining the new $\mathsf{Generalized\ FPT\text{-}ROM}$.

2 The Generalization of FPT-ROM

Notations. If S is a finite set, $|S|$ denotes the number of elements in S. If s is a string, $|s|$ denotes the length of s. For a table $T = \{(x, y)\}$, we define $T(y, r) = \{(\tilde{x}, \tilde{y}) \in T | (\tilde{x} = r \| \tilde{z}) \wedge (\tilde{y} = y)\}$. $x \leftarrow \mathcal{D}$ denotes that x is sampled from distribution $\mathcal{D}$. $B(N, p)$ denotes the binomial distribution with N trials and success probability p. Readers can refer to [21] for details of efficient sampling algorithm of binomial distribution.

We now propose a Generalized $\mathsf{FPT\text{-}ROM}$, denoted by $\mathsf{GFPT\text{-}ROM}$, which captures more practical attacks including the chosen prefix collision attack. Informally, for each random oracle $h : X \to Y$, the adversary can access a *generalized* first pre-image oracle GFPO_h parameterized by a constant k' where $|X| > 2^{k'}$. Given a query $y \in Y$ with an auxiliary input r where $|r| = k'$, GFPO_h uniformly returns a pre-image $x \in X$ such that $h(x) = y$ and the k'-bit prefix of x is r.

Denote $T_h = \{(x, y)\}$ as the hash table of $h : X \to Y$ that defines the correspondence of the elements in X with the elements in Y. Similar to existing WROMs, we require that the message space X to be finite, which is for preserving the weak uniformity of pre-images [21]. GFPT-ROM works as follows:

– Oracle $h(x)$: when an input $x \in X$ is queried, it returns y such that $(x, y) \in T_h$.
– Oracle $\mathsf{GFPO}_h(y, r)$: when $y \in Y$ is queried with an auxiliary input r where $|r| = k'$, it answers as follows: if there is no $(\tilde{x}, y) \in T_h$ such that $\tilde{x} = r\|\tilde{z}$ for some $\tilde{z}$, returns $\perp$, otherwise returns such an $\tilde{x}$ uniformly.

Due to page limitation, we skip the details of the simulation algorithms for oracles h and GFPO_h as they are implicitly shown in the proof of Theorem 3.

Lemma 1. *GFPT-ROM captures the chosen prefix collision attack*[1].

Proof. Given oracle h in GFPT-ROM, a PPT (probabilistic polynomial time) adversary can arbitrarily choose a pair of prefixes (P, P') such that $k' = |P| = |P'|$, and perform the following:

1. Randomly choose S such that $P\|S \in X$, and query h for $C = h(P\|S)$.
2. Query GFPO_h with (C, P'), and get $P'\|S'$.
3. Output a chosen prefix collision $(P\|S,\ P'\|S')$. □

Notice that GFPT-ROM degenerates to FPT-ROM when $k' = 0$.

2.1 Separation for the Security of Signature Schemes

Theorem 1. *GFPT-ROM parameterized by $k' > 0$ relies on a strictly weaker collision resistance assumption than FPT-ROM with respect to the security notion of EUF-CMA for signature schemes.*

We show that $\mathsf{PFDH}^{\oplus}$ signature scheme, though secure in FPT-ROM [27], is insecure in GFPT-ROM. Below is a brief review of $\mathsf{PFDH}^{\oplus}$. Let $h : \{0,1\}^{k'+l} \to \{0,1\}^k$ be a hash function where l is the message length and k is a security parameter. Define $\mathsf{param} := \langle k, l, h \rangle$.

– $(pk, sk) \leftarrow \mathsf{keyGen}(\mathsf{param})$. It generates an RSA tuple $(N, e, d) \overset{R}{\leftarrow} \mathsf{RSA}(1^k)$, and sets $pk = (N, e)$, $sk = (N, d)$.
– $\sigma \leftarrow \mathsf{sign}(sk, m)$. It randomly picks $\tau \in \{0,1\}^{k'}$, computes $\xi = h(\tau\|m)$, $y = \xi \oplus \tau$, $x = y^d \bmod N$ and outputs $\sigma = (\tau, x)$.
– $b \leftarrow \mathsf{verify}(pk, m, \sigma)$. It computes $\xi = h(\tau\|m)$, $y = x^e \bmod N$, and outputs 1 if $y = \xi \oplus \tau$, otherwise outputs 0.

[1] To capture the chosen prefix collision attack, it is enough to generalize CT-ROM in a similar way to what we do on FPT-ROM as shown in this paper. Our generalization on FPT-ROM relies on the weaker collision resistance assumption, and may cover more known or unknown attacks on hash functions.

Lemma 2. *In GFPT-ROM, there exists a PPT adversary $\mathcal{A}$ that breaks PFDH$^{\oplus}$ with probability at least $1 - e^{\frac{1-2^l}{2^k}}$, by making queries to the signing oracle and GFPO$_h$ parameterized by k', where $h : \{0,1\}^{k'+l} \to \{0,1\}^k$ is modeled as a random oracle.*

Proof. We construct an adversary $\mathcal{A}$ as follows:

(1) Query a message m to the signing oracle where $|m| = l$, and obtain a signature $\sigma = (\tau, x)$.
(2) Query h with $\tau \| m$, and get ξ.
(3) Query GFPO_h with (ξ, τ), and get $\tau \| m'$ such that $h(\tau \| m') = h(\tau \| m) = \xi$.
(4) If $m' = m$, **abort**. Otherwise output (m', σ) as a valid forgery.

The probability that the event **abort** happens is:

$$\begin{aligned} Pr[\mathbf{abort}] &= Pr[|T_h(\xi,\tau)| \leq 1 \text{ for } (\tau\|m,\xi) \in T_h] \\ &= Pr[n' = 0 | n' \leftarrow B(2^l - 1, \frac{1}{2^k})] \\ &= (1 - \frac{1}{2^k})^{2^l - 1} \leq e^{\frac{1-2^l}{2^k}} \end{aligned}$$

For example, if $k \geq 1$ and $l = k$, we have $Pr[\mathbf{abort}] \leq e^{-(1-2^{-k})} \leq e^{-\frac{1}{2}}$. So $\mathcal{A}$ can forge a PFDH$^{\oplus}$ signature with probability at least $1 - e^{-1/2}$. □

Theorem 1 follows this lemma directly. GFPT-ROM is therefore separated from FPT-ROM under the security notion of EUF-CMA for signatures, and we have EUF-CMA/FPT-ROM $\Leftarrow$ EUF-CMA/GFPT-ROM.

2.2 Separation for the Security of Encryption Schemes

Theorem 2. *GFPT-ROM parameterized by $k' > 0$ relies on a strictly weaker collision resistance assumption than FPT-ROM with respect to the security notion of IND-CCA2 for encryption schemes.*

Kawachi *et al.* [21] showed that a variant of Fujisaki-Okamato conversion, denote by FO* that applies Naito *et al.*'s approach [25], can be used for constructing IND-CCA2 secure encryption schemes in FPT-ROM. Let $h : \{0,1\}^{k'+l_1+l_2} \to \{0,1\}^{l_1}$ and $g : \{0,1\}^{k'+l_2} \to \{0,1\}^{l_1}$ be two cryptographic hash functions where l_1 and l_2 are security parameters. Let $\mathcal{PKE}$ = (Gen, Enc, Dec) be a OW-CPA secure encryption scheme. By FO* conversion, a message $m \in \{0,1\}^{l_1}$ under the public key (pk, c) is encrypted as $(c_1, c_2) = (\mathsf{Enc}_{pk}(r; h(c\|m\|r)), g(c\|r) \oplus m)$, where pk is a public key produced by Gen, c is a constant string in $\{0,1\}^{k'}$, and r is the randomness picked from $\{0,1\}^{l_2}$. For the decryption of (c_1, c_2), it computes $m = c_2 \oplus g(c\|r)$ where $r = \mathsf{Dec}_{sk}(c_1)$, and outputs m as the plaintext if $c_1 = \mathsf{Enc}_{pk}(r; h(c\|m\|r))$.

Lemma 3. *The encryption schemes converted by FO* are not IND-CPA secure in GFPT-ROM where GFPO$_g$ is parameterized by k', if $k' + l_2 = l_1 + O(1)$.*

The proof is similar to that of Theorem 8 in [21], so is omitted here for lack of space. In [21], it proves that FO conversion is insecure in $\mathsf{FPT\text{-}ROM}$. More precisely, the adversary uses $(0^k, 1^k)$ as the pair of challenge messages, and queries c_2^* to FPO_g where (c_1^*, c_2^*) is the target ciphertext responded by the challenger. For the proof of Lemma 3, the adversary queries (c_2^*, c) to GFPO_g instead of querying c_2^* to FPO_g.

Theorem 2 follows the lemma above, so $\mathsf{GFPT\text{-}ROM}$ is separated from $\mathsf{FPT\text{-}ROM}$ under the security notion of $\mathsf{IND\text{-}CCA2}$ for encryptions, and we have $\mathsf{IND\text{-}CCA2/FPT\text{-}ROM} \Leftarrow \mathsf{IND\text{-}CCA2/GFPT\text{-}ROM}$. By the security reducibility from $\mathsf{FPT\text{-}ROM}$ to $\mathsf{SPT\text{-}ROM}$ and $\mathsf{CT\text{-}ROM}$ [27,21], we deduce that $\mathsf{GFPT\text{-}ROM}$ captures all the attacks covered by existing WROMs.

3 Generic Transformation for Signatures in GFPT-ROM

We give a generic transformation to convert a large class of Hash-and-Sign Signature schemes in ROM to secure ones in $\mathsf{GFPT\text{-}ROM}$, with no expansion of signature size. First we formally define Hash-and-Sign Signatures, then explain how our transformation works.

3.1 Hash-and-Sign Signatures

A large class of signature schemes, including FDH signature, Schnorr signature, and so on, are constructed by the Hash-and-Sign paradigm. The paradigm is usually referred to as $\mathsf{sign}(H(m))$ where sign is a signing algorithm and H is a cryptographic hash function. In this section, we present a more generic formalization of Hash-and-Sign Signatures (HaSS) where the hash functions can either be salted or non-salted.

Let us first review the definition of digital signature. It consists of the following three algorithms: keyGen, sign, verify.

– $(pk, sk) \leftarrow \mathsf{keyGen}(\mathsf{param})$. It outputs a key pair $(pk,\ sk)$ on the system parameters param.
– $\sigma \leftarrow \mathsf{sign}(sk, m)$. It outputs a signature σ on a message m.
– $b \leftarrow \mathsf{verify}(pk, m, \sigma)$. It outputs a bit b to indicate if σ is a valid signature of m $(b = 1)$ or not $(b = 0)$.

Definition 1 (Hash-and-Sign Signatures). *Hash-and-sign signature HaSS is a class of signature schemes that* sign (*resp.* verify) *consists of three subprocedures* $\{\mathsf{preSign},\ \mathsf{hash},\ \mathsf{postSign}\}$ (*resp.* $\{\mathsf{preVer}, \mathsf{hash}, \mathsf{postVer}\}$)*, where only* hash *invokes hash functions.*

More precisely, the signing and verification algorithm work as follows:
$\mathsf{sign}(sk, m)$
(1) Run $(Str, \tau) = \mathsf{preSign}(sk, m)$.
(2) Compute $\xi = \mathsf{hash}(Str)$.
(3) Run $\sigma = \mathsf{postSign}(sk, \xi, \tau)$, output σ as the signature.

$\mathsf{verify}(pk, m, \sigma)$
(1) Run $Str = \mathsf{preVer}(pk, m, \sigma)$.
(2) Compute $\xi = \mathsf{hash}(Str)$.
(3) Run $b = \mathsf{postVer}(pk, \xi, \sigma)$ and output b.
where the sub-procedures are listed by:

- $(Str, \tau) \leftarrow \mathsf{preSign}(sk, m)$. It outputs a string Str and some auxiliary information τ on the message m.
- $\xi \leftarrow \mathsf{hash}(Str)$. It outputs a hashed value ξ of Str.
- $\sigma \leftarrow \mathsf{postSign}(sk, \xi, \tau)$. It outputs a signature σ on a hashed message ξ using sk and some auxiliary input τ.
- $Str \leftarrow \mathsf{preVer}(pk, m, \sigma)$. It outputs a string Str on the signature σ of a message m. This is a deterministic algorithm.
- $b \leftarrow \mathsf{postVer}(pk, \xi, \sigma)$. It outputs a bit b to indicate if σ is a valid signature ($b = 1$) or not ($b = 0$).

Notice that HaSS covers both probabilistic and deterministic signatures, depending on whether the randomness τ is void or not. The EUF-CMA security for HaSS is defined similarly to that of conventional signatures [15].

3.2 Transformation

Given any HaSS scheme that uses a group hash function modeled as a random oracle, we propose a generic transformation to construct a signature scheme with provable security in GFPT-ROM. A group hash function is an efficient function that maps binary strings into a group $\mathbb{G}$. The transformed signature is also a HaSS scheme, and it does not have size expansion compared to the original signature.

Definition 2 (Transformation). *Given any HaSS scheme $S_h = (\mathsf{keyGen}, \mathsf{preSign}, \mathsf{hash}, \mathsf{postSign}, \mathsf{preVer}, \mathsf{postVer})$ such that $S_h.\mathsf{hash} := \{\xi \leftarrow h(Str)\}$ where $h : \mathcal{M} \rightarrow \mathbb{G}$ is a hash function and $\mathbb{G}$ is a cyclic group, we can choose two independent hash functions $u, v : \mathcal{M} \rightarrow \mathbb{G}$ and construct a new HaSS scheme $\tilde{S}_h$ as below:*

$$\tilde{S}_h.\mathsf{keyGen} := S_h.\mathsf{keyGen}$$
$$\tilde{S}_h.\mathsf{preSign} := S_h.\mathsf{preSign}$$
$$\tilde{S}_h.\mathsf{postSign} := S_h.\mathsf{postSign}$$
$$\tilde{S}_h.\mathsf{preVer} := S_h.\mathsf{preVer}$$
$$\tilde{S}_h.\mathsf{postVer} := S_h.\mathsf{postVer}$$
$$\tilde{S}_h.\mathsf{hash} := \{\xi \leftarrow u(Str) \cdot v(Str)\}$$

where "$\cdot$" is the group operation defined on $\mathbb{G}$.

Since that the hash functions u, v, h map to the same group $\mathbb{G}$, $\tilde{S}_h$ has exactly the same signature size as that of S_h, and there is almost no increase of computational cost in the transformation.

We prove that the transformed signature $\tilde{S}_h$ is $\mathsf{EUF\text{-}CMA}$ secure in $\mathsf{GFPT\text{-}ROM}$, if the original scheme S_h is $\mathsf{EUF\text{-}CMA}$ secure in ROM.

Theorem 3. *Let $\tilde{S}_h$ be any signature scheme constructed by the generic transformation (Def. 2), and S_h be the original scheme proven secure in ROM. Then in GFPT-ROM where both GFPO_u and GFPO_v are parameterized by $k' \geq 0$ and $|\mathcal{M}| > 2^{k'}$, for all PPT machines that (t_{euf}, ϵ_{euf})-break $\tilde{S}_h$ by making q_{sig} queries to the signing oracle, q_u (resp. q_v) queries to u (resp. v), and q_u^{fp} (resp. q_v^{fp}) queries to GFPO_u (resp. GFPO_v), there exists a PPT machine that (t_{euf}, ϵ_{euf})-break S_h in ROM by making q_{sig} queries to the signing oracle and q_h queries to h, where $q_h = q_u + q_v + q_u^{fp} + q_v^{fp}$.*

Proof. Assume there exists an adversary $\mathcal{F}$ that forges a signature of $\tilde{S}_h$ with probability ϵ_{euf} within time t_{euf}, we build an adversary $\mathcal{C}$ that forges a signature of S_h with the same probability and time bound. Let us denote by T_u, L_u (resp. T_v, L_v) the tables used for simulating the oracles referring to u (resp. v). All the tables are empty at the beginning.

$\mathcal{C}$ simulates the signing oracle of $\tilde{S}_h$, the random oracles u, v and the generalized first pre-image tractable oracles for u, v as follows:

i. When $\mathcal{F}$ queries $Str \in \mathcal{M}$ to u, $\mathcal{C}$ performs the following steps:
   (1) If there is an entry $(Str, \tilde{\xi}_u) \in T_u$, then set $\xi_u = \tilde{\xi}_u$ and return ξ_u.
   (2) Parse $Str = r_u \| z_u$ such that $|r_u| = k'$.
   (3) Flip a biased coin with probability $Pr[\alpha = 0] = \frac{\sum_{((\tilde{\xi}_u, r_u), \tilde{n}) \in L_u} (\tilde{n} - |T_u(\tilde{\xi}_u, r_u)|)}{|\mathcal{M}| - |T_u|}$ that the oracle returns an old $\tilde{\xi}_u$ with the same prefix r_u, i.e. there already exists $(\tilde{Str}, \tilde{\xi}_u) \in T_u$ for some $\tilde{Str} \neq Str$ and $\tilde{Str} = r_u \| \tilde{z}_u$.
   (4) If $\alpha = 0$, pick $\xi_u \leftarrow \mathcal{D}$ and go to Step (7). The probability mass function of distribution $\mathcal{D}$ is defined as $f_{\mathcal{D}}(\xi_u) = \frac{n - |T_u(\xi_u, r_u)|}{\sum_{((\tilde{\xi}_u, \tilde{r}_u), \tilde{n}) \in L_u} (\tilde{n} - |T_u(\tilde{\xi}_u, \tilde{r}_u)|)}$, for any $((\xi_u, r_u), n) \in L_u$.
   (5) If $\alpha \neq 0$, pick $\xi_u \leftarrow \mathbb{G} \setminus \cup_{((\tilde{\xi}_u, \tilde{r}_u), \tilde{n}) \in L_u} \tilde{\xi}_u$ uniformly.
   (6) Pick $n' \leftarrow B(\frac{|\mathcal{M}| - \sum_{((\tilde{\xi}_u, \tilde{r}_u), \tilde{n}) \in L_u} \tilde{n}}{2^{k'}} - 1, \frac{1}{|\mathbb{G}| - |L_u|})$, set $n = n' + 1$ and insert $((\xi_u, r_u), n)$ in L_u.
   (7) Insert (Str, ξ_u) in T_u, call $\mathsf{SyncTables}(Str, \xi_u, T_v, L_v)$, and return ξ_u.

   The function $\mathsf{SyncTables}(Str, \xi_u, T_v, L_v)$ is defined as following, where $Str \in \mathcal{M}$, $\xi_u \in \mathbb{G}$, and T_v, L_v are the tables for simulating v and GFPO_v.

   (1) Query Str to h, and get ξ_h.
   (2) Compute $\xi_v = \xi_h \cdot \xi_u^{-1}$, and insert (Str, ξ_v) in T_v.
   (3) Parse $Str = r_v \| z_v$ such that $|r_v| = k'$.
   (4) If there is no entry $((\xi_v, r_v), \tilde{n}) \in L_v$, pick $n' \leftarrow B(\frac{|\mathcal{M}| - \sum_{((\tilde{\xi}_v, \tilde{r}_v), \tilde{n}) \in L_v} \tilde{n}}{2^{k'}} - 1, \frac{1}{|\mathbb{G}| - |L_v|})$, set $n = n' + 1$ and insert $((\xi_v, r_v), n)$ in L_v.

   This sub-procedure synchronizes the entries in T_v and L_v whenever a new entry (Str, ξ_u) is inserted into T_u.

ii. When $\mathcal{F}$ queries (ξ_u, r_u) where $\xi_u \in \mathbb{G}$ and $|r_u| = k'$ to GFPO_u, $\mathcal{C}$ performs the following steps:

(1) If there is no entry $((\xi_u, r_u), \tilde{n}) \in L_u$, pick $n \leftarrow B(\frac{|\mathcal{M}| - \sum_{((\tilde{\xi}_u, \tilde{r}_u), \tilde{n}) \in L_u} \tilde{n}}{2^{k'}}, \frac{1}{|\mathbb{G}| - |L_u|})$, and insert $((\xi_u, r_u), n)$ in L_u.
(2) If $n = 0$ for $((\xi_u, r_u), n) \in L_u$, return $\perp$.
(3) If $n \neq 0$ for $((\xi_u, r_u), n) \in L_u$, flip a biased coin with probability $Pr[\beta = 0] = \frac{|T_u(\xi_u, r_u)|}{n}$ that the oracle returns an old $\tilde{Str} = r_u \| \tilde{z}_u$, i.e. there already exists an entry $(r_u \| \tilde{z}_u, \xi_u) \in T_h$ for some $\tilde{z}_u$.
(4) If $\beta = 0$, pick an entry $(\tilde{Str}, \xi_u) \in T_u$ such that $\tilde{Str} = r_u \| \tilde{z}_u$ uniformly, set $Str = \tilde{Str}$ and return Str.
(5) If $\beta \neq 0$, pick $Str \leftarrow \mathcal{M}$ uniformly such that $Str = r_u \| z_u$ and there is no entry $(Str, \tilde{\xi}_u) \in T_u$.
(6) Insert (Str, ξ_u) in T_u, call $\mathsf{SyncTables}(Str, \xi_u, T_v, L_v)$, and return Str.

iii. When $\mathcal{F}$ queries $Str \in \mathcal{M}$ (resp. (ξ_v, r_v) where $\xi_v \in \mathbb{G}$ and $|r_v| = k'$) to v (resp. GFPO_v), $\mathcal{C}$ works similarly as querying u (resp. GFPO_u), except that all the variables referring to u are exchanged with the corresponding variables referring to v. For instance, $\mathsf{SyncTables}(Str, \xi_v, T_u, L_u)$ is called instead.

iv. When $\mathcal{F}$ queries a message m to the signing oracle of $\tilde{S}_h$, $\mathcal{C}$ performs the following steps:

(1) Query m to the signing oracle of S_h and get σ.
(2) Run $Str = \mathsf{preVer}(pk, m, \sigma)$.
(3) If there is no entry $(Str, \tilde{\xi}_u) \in T_u$, query Str to u.
(4) Output σ as the signature.

The simulation for signing oracle of $\tilde{S}_h$ is perfect since that:[2]

$$\begin{aligned}
1 &= S_h.\mathsf{verify}(pk, m, \sigma) \\
&= \mathsf{postVer}(pk, S_h.\mathsf{hash}(\mathsf{preVer}(pk, m, \sigma)), \sigma) \\
&= \mathsf{postVer}(pk, h(Str), \sigma) \\
&= \mathsf{postVer}(pk, u(Str) \cdot v(Str), \sigma) \\
&= \mathsf{postVer}(pk, \tilde{S}_h.\mathsf{hash}(\mathsf{preVer}(pk, m, \sigma)), \sigma) \\
&= \tilde{S}_h.\mathsf{verify}(pk, m, \sigma)
\end{aligned}$$

Eventually $\mathcal{F}$ outputs a forgery $(m^\star, \sigma^\star)$ of the signature $\tilde{S}_h$. Then $\mathcal{C}$ performs the following computation: (1) Run $Str^\star = \mathsf{preVer}(pk, m^\star, \sigma^\star)$; (2) If there is no entry $(Str^\star, \tilde{\xi}_u^\star) \in T_u$, query $Str^\star$ to u. Due to $\mathsf{SyncTables}$, there must exist an entry $(Str^\star, \xi_u^\star)$ in T_u and an entry $(Str^\star, \xi_v^\star)$ in T_v, such that $u(Str^\star) \cdot v(Str^\star) = \xi_u^\star \cdot \xi_v^\star = \xi_h^\star = h(Str^\star)$. $\mathcal{C}$ outputs $(m^\star, \sigma^\star)$ as a forgery of the signature S_h.

The running time of $\mathcal{C}$ is approximately the running time of $\mathcal{F}$, since that whenever $\mathcal{F}$ queries to the random oracles u, v or GFPO_u, GFPO_u, $\mathcal{C}$ makes a random oracle query to h; whenever $\mathcal{F}$ queries to the signing oracle of $\tilde{S}_h$, $\mathcal{C}$ makes a signing query to S_h on the same message. The winning advantage of $\mathcal{C}$ is also approximately the same as $\mathcal{F}$ since that $\mathcal{C}$ outputs a forgery of S_h whenever $\mathcal{F}$ outputs a forgery of $\tilde{S}_h$. $\square$

[2] As shown by the transformation, S_h and $\tilde{S}_h$ share the same verification algorithms $\mathsf{postVer}$ and preVer.

4 Application and Comparison

Applying the transformation, we construct an efficient and short signature in $\mathsf{GFPT\text{-}ROM}$ from FDH signature [3].

First, we show that FDH is a HaSS scheme. Define $\mathsf{param} := \langle k, l \rangle$ where l is the message length and k is the security parameter.

- $(pk, sk) \leftarrow \mathsf{keyGen}(\mathsf{param})$. It generates an RSA tuple $(N, e, d) \overset{R}{\leftarrow} \mathsf{RSA}(1^k)$, selects a cryptographic hash function $h : \{0,1\}^l \rightarrow \mathbb{Z}_N^*$, and sets $pk = (N, e, h)$, $sk = (N, d, h)$.
- $(Str, \tau) \leftarrow \mathsf{preSign}(sk, m)$. It sets $Str = m$ and $\tau = \perp$.
- $\xi \leftarrow \mathsf{hash}(Str)$. It outputs $\xi = h(Str)$.
- $\sigma \leftarrow \mathsf{postSign}(sk, \xi, \tau)$. It outputs $\sigma = \xi^d \bmod N$.
- $Str \leftarrow \mathsf{preVer}(pk, m, \sigma)$. It outputs $Str = m$.
- $b \leftarrow \mathsf{postVer}(pk, \xi, \sigma)$. It outputs 1 if $\xi = \sigma^e \bmod N$, otherwise outputs 0.

The security of FDH is based on RSA assumption.

Definition 3 (RSA Problem). *Let N be a randomly generated RSA modulus on the security parameter k, (e, d) be a pair of RSA exponents such that $e \in \mathbb{Z}_{\phi(N)}^*$ and $e * d = 1 \bmod \phi(N)$. The RSA problem is that given (N, e) and a random $\eta \in \mathbb{Z}_N^*$, find $\eta^d \bmod N$. The RSA assumption holds if no PPT machine can solve this problem with non-negligible probability in k.*

Numayama *et al.* showed that FDH is not secure in $\mathsf{CT\text{-}ROM}$ [27]. A FDH variant denoted by FDH$^+$ can be constructed following our transformation, by replacing the hash function h with two independent ones u, v. Coron proved the security of FDH in ROM [8]. Applying Theorem 3, FDH$^+$ is proven secure in $\mathsf{GFPT\text{-}ROM}$ by reduction to the hardness of RSA problem, as given by the theorem below. We skip the proof here.

Theorem 4. *In $\mathsf{GFPT\text{-}ROM}$, for all PPT machines that $(t_{euf}, \epsilon_{euf})$-break FDH$^+$ by making q_{sig} queries to the signing oracle, q_u (resp. q_v) queries to u (resp. v), q_u^{fp} (resp. q_v^{fp}) queries to GFPO_u (resp. GFPO_v), there exists a PPT machine that $(t_{rsa}, \epsilon_{rsa})$-break the RSA assumption such that:*

$$t_{euf} \approx t_{rsa} - (q_{sig} + q_u + q_v + q_u^{fp} + q_v^{fp} + 1) * O(k^3)$$

$$\epsilon_{euf} \approx \frac{1}{(1 - \frac{1}{q_{sig}+1})^{q_{sig}+1}} * q_{sig} * \epsilon_{rsa} \approx e * q_{sig} * \epsilon_{rsa}$$

In Table 1, we compare FDH$^+$ with the best existing signatures based on RSA assumption (Def. 3).[3] HW scheme is the first short and stateless RSA signature

[3] The signature size and public key size of the schemes are represented by bit. The RSA modulus N is 1024 bits long. All the schemes provide existential unforgeability with $\kappa = 80$ bits of security after adaptively revealing maximally $q_{sig} = 2^{30}$ signatures, i.e. $k' = 30$. The message length is $l = 2\kappa = 160$ bits to provide 80-bit security. s and r denote the randomness used in signing. P_{1024} denotes the time needed to generate and test a 1024-bit prime. EXP denotes 1 full modular exponentiation over $\mathbb{Z}_N^*$.

Table 1. Comparison of RSA-based Hash-and-Sign Signatures

Schemes	**Signature Size**	**Public Key Size**	**Signing Cost**	**Model**
HW [20]	$2 \times \lvert\mathbb{Z}_N\rvert = 2048$	$\lvert\mathbb{Z}_N\rvert + \lvert pk_{\mathrm{CH}}\rvert = 3\mathrm{k}$	$160 \times \mathrm{P}_{1024}$	–
$\mathrm{H}_{\mathrm{cfs}}$ $(m = 4)$ [18]	$\lvert\mathbb{Z}_N\rvert + \lvert s\rvert = 1074$	$16m^2 l \times \lvert\mathbb{Z}_N\rvert = 40\mathrm{m}$	$50 \times \mathrm{P}_{1024}$	–
$\mathrm{H}_{\mathrm{rand}}$ $(m = 4)$ [18]	$\lvert\mathbb{Z}_N\rvert + \lvert s\rvert + \lvert r\rvert = 1214$	$2m^2 l \times \lvert\mathbb{Z}_N\rvert = 32\mathrm{k}$	$50 \times \mathrm{P}_{1024}$	–
$\mathrm{H}_{\mathrm{weak}}$ $(m = 11)$ [18]	$2 \times \lvert\mathbb{Z}_N\rvert = 2048$	$12\lvert\mathbb{Z}_N\rvert + \lvert pk_{\mathrm{CH}}\rvert = 14\mathrm{k}$	$1 \times \mathrm{P}_{1024}$	–
FDH [3]	$\lvert\mathbb{Z}_N\rvert = 1024$	$\lvert\mathbb{Z}_N\rvert = 1\mathrm{k}$	$1 \times \mathrm{EXP}$	ROM
PFDH [9]	$\lvert\mathbb{Z}_N\rvert + \lvert r\rvert = 1054$	$\lvert\mathbb{Z}_N\rvert = 1\mathrm{k}$	$1 \times \mathrm{EXP}$	CT-ROM
PFDH^{+} [27]	$\lvert\mathbb{Z}_N\rvert + \lvert r\rvert = 1054$	$\lvert\mathbb{Z}_N\rvert = 1\mathrm{k}$	$1 \times \mathrm{EXP}$	SPT-ROM
$\mathrm{PFDH}^{\oplus}$ [27]	$\lvert\mathbb{Z}_N\rvert + \lvert r\rvert = 1054$	$\lvert\mathbb{Z}_N\rvert = 1\mathrm{k}$	$1 \times \mathrm{EXP}$	FPT-ROM
FDH^{+}	$\lvert\mathbb{Z}_N\rvert = 1024$	$\lvert\mathbb{Z}_N\rvert = 1\mathrm{k}$	$1 \times \mathrm{EXP}$	GFPT-ROM

in the standerd model, which needs a chameleon hash function CH to achieve EUF-CMA security. $\mathrm{H}_{\mathrm{cfs}}$, $\mathrm{H}_{\mathrm{rand}}$ and $\mathrm{H}_{\mathrm{weak}}$ are implemented by several $(m, 1)$-programmable hash functions, providing various tradeoffs between signature size, public key size and signing cost. Though the schemes are proven secure without the random oracle methodology, all of them have to generate l random primes of 1024 bits for signing l-bit messages. On the other hand, FDH^{+} outperforms these signatures by cutting down $5\% \sim 50\%$ of signature size and at least 66% of public key size. Besides, modular exponentiation is the dominant operation in the signing algorithm of FDH^{+}, which requires for much less computations than generating long primes. Therefore, FDH^{+} is more preferable for resource-limited applications. Even compared to FDH, FDH^{+} needs only one more hash operation and one multiplication on $\mathbb{Z}_N^*$ (the computational cost is negligible to modular exponentiation, so is omitted in the table). Furthermore, the security in GFPT-ROM ensures that FDH^{+} is robust against more practical attacks, including the chosen prefix collision attack which can easily break FDH, PFDH, PFDH^{+} and $\mathrm{PFDH}^{\oplus}$. These results illustrate that our transformation offers a flexible tradeoff between efficiency and security of Hash-and-Sign Signatures.

Besides FDH signature, our transformation is applicable to many other well-known signature schemes, e.g. Boneh *et al.*'s short signature BLS [6]. It also works for all the signature schemes applying Fiat-Shamir heuristics [12], e.g. Schnorr signature [31] and Hess's identity based signature on pairing [17]. We leave the details to the full version of this paper.

5 Conclusion

In this paper, we weakened FPT-ROM to a more generalized model, namely GFPT-ROM, which ensures security against many practical attacks on cryptographic hash functions, including the chosen prefix collision attack and all the attacks captured by the existing WROMs. Besides, a generic transformation is proposed to construct a large class of Hash-and-Sign Signatures in the new model. The transformed signatures are both short and efficient as the best existing ones, in the sense that there is no expansion of signature size, and almost no increase of public key size or signing cost in the transformation. We leave the study of encryption schemes in GFPT-ROM as one of our future work.

References

1. Bellare, M., Micali, S.: How to sign given any trapdoor permutation. Journal of the ACM 39(1), 214–233 (1992)
2. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: Denning, D.E., Pyle, R., Ganesan, R., Sandhu, R.S., Ashby, V. (eds.) CCS 1993, pp. 62–73. ACM Press, New York (1993)
3. Bellare, M., Rogaway, P.: The Exact Security of Digital Signatures - How to Sign with RSA and Rabin. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 399–416. Springer, Heidelberg (1996)
4. Bellare, M., Rogaway, P.: Collision-Resistant Hashing: Towards Making UOWHFs Practical. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 470–484. Springer, Heidelberg (1997)
5. MacKenzie, P.D., Yang, K.: On Simulation-Sound Trapdoor Commitments. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 382–400. Springer, Heidelberg (2004)
6. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the weil pairing. Journal of Cryptology 17(4), 297–319 (2004)
7. De Cannière, C., Rechberger, C.: Preimages for Reduced SHA-0 and SHA-1. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 179–202. Springer, Heidelberg (2008)
8. Coron, J.-S.: On the Exact Security of Full Domain Hash. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 229–235. Springer, Heidelberg (2000)
9. Coron, J.-S.: Optimal Security Proofs for PSS and Other Signature Schemes. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 272–287. Springer, Heidelberg (2002)
10. Cramer, R., Shoup, V.: Signature schemes based on the strong RSA assumption. ACM Transactions on Information and System Security 3(3), 161–185 (2000)
11. Dwork, C., Naor, M.: An Efficient Existentially Unforgeable Signature Scheme and Its Applications. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 234–246. Springer, Heidelberg (1994)
12. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
13. Gennaro, R., Halevi, S., Rabin, T.: Secure Hash-and-Sign Signatures without the Random Oracle. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 123–139. Springer, Heidelberg (1999)
14. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Dwork, C. (ed.) STOC 2008, pp. 197–206. ACM Press, New York (2008)
15. Goldwasser, S., Micali, S., Rivest, R.: A digital signature scheme secure against adaptive chosen message attacks. SIAM Journal on Computing 17(2), 281–308 (1988)
16. Halevi, S., Krawczyk, H.: Strengthening Digital Signatures Via Randomized Hashing. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 41–59. Springer, Heidelberg (2006)
17. Hess, F.: Efficient Identity Based Signature Schemes Based on Pairings. In: Nyberg, K., Heys, H.M. (eds.) SAC 2002. LNCS, vol. 2595, pp. 310–324. Springer, Heidelberg (2003)

18. Hofheinz, D., Jager, T., Kiltz, E.: Short signatures from weaker assumptions (2011), http://eprint.iacr.org/2011/296.pdf
19. Hofheinz, D., Kiltz, E.: Programmable Hash Functions and Their Applications. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 21–38. Springer, Heidelberg (2008)
20. Hohenberger, S., Waters, B.: Short and Stateless Signatures from the RSA Assumption. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 654–670. Springer, Heidelberg (2009)
21. Kawachi, A., Numayama, A., Tanaka, K., Xagawa, K.: Security of Encryption Schemes in Weakened Random Oracle Models (Extended Abstract). In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 403–419. Springer, Heidelberg (2010)
22. Leurent, G.: MD4 is Not One-Way. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 412–428. Springer, Heidelberg (2008)
23. Liskov, M.: Constructing an Ideal Hash Function from Weak Ideal Compression Functions. In: Biham, E., Youssef, A.M. (eds.) SAC 2006. LNCS, vol. 4356, pp. 358–375. Springer, Heidelberg (2007)
24. Mironov, I.: Collision-Resistant No More: Hash-and-Sign Paradigm Revisited. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 140–156. Springer, Heidelberg (2006)
25. Naito, Y., Wang, L., Ohta, K.: How to construct cryptosystems and hash functions in weakened random oracle models. Cryptology ePrint Archive, Report 2009/550 (2009)
26. Naor, M., Yung, M.: Universal one-way hash functions and their cryptographic applications. In: Johnson, D.S. (ed.) STOC 1989, pp. 33–43. ACM Press, New York (1989)
27. Numayama, A., Isshiki, T., Tanaka, K.: Security of Digital Signature Schemes in Weakened Random Oracle Models. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 268–287. Springer, Heidelberg (2008)
28. Pasini, S., Vaudenay, S.: Hash-and-Sign with Weak Hashing Made Secure. In: Pieprzyk, J., Ghodosi, H., Dawson, E. (eds.) ACISP 2007. LNCS, vol. 4586, pp. 338–354. Springer, Heidelberg (2007)
29. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM 21(2), 120–126 (1978)
30. Rompel, J.: One-way functions are necessary and sufficient for secure signatures. In: Ortiz, H. (ed.) STOC 1990, pp. 387–394. ACM Press, New York (1990)
31. Schnorr, C.P.: Efficient signature generation by smart cards. Journal of Cryptology 4(3), 161–174 (1991)
32. Stevens, M., Lenstra, A.K., de Weger, B.: Chosen-Prefix Collisions for MD5 and Colliding X.509 Certificates for Different Identities. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 1–22. Springer, Heidelberg (2007)
33. Stevens, M., Sotirov, A., Appelbaum, J., Lenstra, A., Molnar, D., Osvik, D.A., de Weger, B.: Short Chosen-Prefix Collisions for MD5 and the Creation of a Rogue CA Certificate. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 55–69. Springer, Heidelberg (2009)
34. Wang, X., Yin, Y.L., Yu, H.: Finding Collisions in the Full SHA-1. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 17–36. Springer, Heidelberg (2005)
35. Wang, X., Yu, H.: How to Break MD5 and Other Hash Functions. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 19–35. Springer, Heidelberg (2005)
36. Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)

A Short Non-delegatable Strong Designated Verifier Signature

Haibo Tian[1], Xiaofeng Chen[2], and Jin Li[3]

[1] School of Information Science and Technology
Sun Yat-Sen University, Guangzhou, Guangdong, 510275, P.R. China
[2] State Key Laboratory of Integrated Service Networks (ISN),
Xidian University, Xi'an 710071, P.R. China
[3] School of Computer Science, Guangzhou University, 510006, P.R. China
{sysutianhb,chenerry,Jinli71}@gmail.com

Abstract. We propose a non-delegatable strong designated verifier signature (SDVS) featured by a two-element signature. Comparatively, current SDVS schemes without delegatability produce at least three-element signatures. The SDVS scheme provably satisfies the unforgeability property under a computational Diffie-Hellman (CDH) problem. Its non-delegatability holds conditioned on a knowledge extraction assumption (KEA). Its privacy of signer's identity (PSI) is reduced to the hardness of a variant of CDH problem. The construction method utilizes a combination of a KEA-based identification protocol, an OR proof technique, and a Fiat-Shamir heuristic.

Keywords: Signature Schemes, Strong Designated Verifier Signature, Non-delegatability.

1 Introduction

Jakobsson et al. [14] proposed the concept of designated verifier signature (DVS). A DVS consists of a proof that either "the signer has signed on a message" or "the signer has the verifier's secret key". If a designated verifier is confident that her/his private key is kept in secret, the verifier makes sure that a signer has signed on a message. No other parties can be convinced by the DVS since the designated verifier can generate it with her/his private key. It is useful in various commercial cryptographic applications, such as e-voting, copyright protection.

A strong DVS (SDVS) is an extension of the DVS. In the appendix, Jakobsson et al. [14] gave a definition of SDVS. It means that a verifier needs to use her/his private key to verify the signature. It considers a situation where a signature is captured before reaching a designated verifier. In this case, an adversary can know who is the real signer as there are only two possibilities. Laguillaumie and Vergnaud [17], and Saeednia [24] both formalized the notion.

Lipmaa et al. [21] proposed the non-delegatability property of a DVS scheme. A real-life scenario about the property is that an adversary at first interacts with a signer to obtain something, and then creates signatures for a particular designated verifier. It is intended to prevent a dishonest signer to sell a specifically constructed secret. The delegatability property is believed undesirable in many applications of DVS schemes, such as a hypothetical e-voting protocol, an e-library application.

This paper focuses on SDVS schemes without delegatability. It reports a scheme providing two-element non-delegatable signatures.

1.1 Contribution

We combine an identification protocol with an OR proof technique. Wu et al. [31] proposed an identification protocol based on a knowledge extraction assumption (KEA) [5]. Basically, the KEA means that if a probabilistic polynomial time algorithm takes as input a group element g^{α} and produces a pair of elements $(g^{\beta}, g^{\alpha\beta})$, it can print the value β, where g is a generator of a cyclic group. In the identification scheme, a verifier produces a value g^{α} as a challenge, and a prover produces a value $g^{\alpha\beta}$ to prove its ownership of the value β. By using the OR proof technique, we obtain the following KEA-based OR protocol.

- A verifier produces a value g^{α}.
- A prover then produces two values $g^{\alpha_1\beta}$ and $g^{\alpha_2\gamma}$ satisfying $\alpha_1 + \alpha_2 = \alpha$ to prove its ownership of β or γ. It returns a tuple $(g^{\alpha_1}, g^{\alpha_1\beta}, g^{\alpha_2\gamma})$ to the verifier.
- However, the verifier cannot verify the tuple directly as it knows nothing about the value α_1 or α_2. To make it workable, a simple method is to use the pairing technique so that anybody can verify the relationship of the values g^{α_1} and $g^{\alpha_1\beta}$.

The KEA-based OR protocol can be turned into a DVS scheme by using the Fiat-Shamir [7] heuristic. Interestingly, if we remove one element from the DVS signature, we turn it into an SDVS signature. For example, if the value β is taken as a verifier's private key, an SDVS signature consists of two elements $(g^{\alpha_1\beta}, g^{\alpha_2\gamma})$.

To make the scheme provable, we use a symmetric encryption algorithm and model it as a random oracle. Although the building blocks include a KEA-based protocol, the unforgeability and PSI properties are reduced to the hardness of CDH problem and its variant, the inverse CDH problem. Only the non-delegatability property needs the KEA assumption.

In summary, we give a non-delegatable two-element SDVS by combining a KEA-based identification protocol, an OR proof technique, and the Fiat-Shamir heuristic.

1.2 Related Works

There are some approaches to construct a non-delegatable DVS scheme:

1. Jakobsson et al. [14] proposed the first DVS scheme. It is a combination of a trapdoor commitment and a Schnorr signature. A signer adds a committed

value of a commitment to a hash value in a signature scheme. A verifier then can simulate a signature by producing a committed value for a given commitment. Lipmaa et al. [21] and Wang [30] proposed a DVS and an identity-based DVS (IBDVS) by using the similar method, separately.
2. Huang et al. [12] proposed a non-delegatable universal DVS scheme. It is a combination of a Schnorr style signature and the OR proof technique. A hash value in the signature scheme is divided into two challenges. A signer answers one challenge by a simulation and the other by using its private key. A verifier does the similar thing to simulate a signature. The schemes proposed in [9–11] are constructed in a similar way.
3. Tian et al. [29] proposed a non-delegatable SDVS scheme. It is a combination of a Schnorr signature, a KEA-based identification scheme, and the OR proof technique. The KEA assumption is used partially to extract a private key in their non-delegatability proof.
4. Feng et al. [8] proposed two general methods to construct SDVS schemes. Their discrete logarithm instantiation seems to be non-delegatable. The instantiation is based on the ring signature. The signature is Schnorr style.

Besides the construction methods, some papers discuss the non-delegatability property of DVS schemes. Lipmaa et al. [21] shows that schemes in [17,24–26] are delegatable. Li et al. [20] shows that schemes in [18,23,27,32] are delegatable. The two papers show that most DVS schemes before 2005 in the literature are delegatable. Huang et al. [11] claims that some identity-based schemes in [3,13, 16] are delegatable. Zhang et al. [33] shows that a scheme in [34] is delegatable. Tian et al. [29] shows that a scheme in [19] is delegatable. Sun et al. [28] shows that a scheme in [15] is delegatable.

Although the non-delegatability property has been a focus of many literatures, it is a controversial concept as it may be undesirable in some applications. In our opinion, the non-delegatable DVS schemes can be viewed as a special category. Schemes in this category have its own application area where the responsibility of a signer is important and cannot be delegated to another entity.

1.3 Organizations

The next section gives some preliminaries about SDVS and mathematical assumptions. Section 3 is our SDVS scheme. The proofs of the scheme are in Section 4. Section 5 compares our scheme and other non-delegatable schemes.

2 Preliminaries

2.1 Assumptions

Let $\mathbb{G}$ be a cyclic group with a large-prime order q. Let g be the generator of $\mathbb{G}$.

– **Computational Diffie-Hellman (CDH) Problem:** Given $g^{\alpha}, g^{\beta} \in \mathbb{G}$, compute $g^{\alpha\beta}$.

– **Inverse Computational Diffie-Hellman (InvCDH) Problem:** Given $g^{\alpha} \in \mathbb{G}$, compute $g^{\alpha^{-1}}$.

Bao et al. [1] have proven the equivalence of the two problems. We use the InvCDH problem for convenience in the proof of privacy of signer's identity. The assumption is that there are no (ϵ, t) algorithms to solve the CDH (InvCDH) problem in time t with a non-negligible probability ϵ if q is big enough.

Knowledge Extractor Assumption version 1 (KEAv1) [5]: Let T denote a polynomial time bounded algorithm which on input (g, g^{α}), produces $(g^{\beta}, g^{\alpha\beta})$, where β is chosen by T. The assumption is that there is another polynomial time bounded algorithm T^*, which takes the same input as T, uses the same coins as T, and produces $(\beta, g^{\beta}, g^{\alpha\beta})$ with a probability $1 - \epsilon$ where ϵ is a negligible value.

Remark 1. The KEAv1 is proven in a generic group model [6]. This gives it an evidence about its plausibility. It is only used in the proof of the non-delegatability property.

2.2 SDVS

We define an SDVS scheme as follows.

– *Setup*: A probabilistic polynomial time algorithm, on input a security parameter $\kappa \in \mathbb{Z}$, produces system parameters *sp*.
– *KeyGen*: A probabilistic polynomial time algorithm, on input the system parameters *sp*, produces key pairs (y_S, x_S) for a signer S, and (y_V, x_V) for a verifier V.
– *Sign*: A probabilistic polynomial time algorithm, on input the system parameters *sp*, a signer's private key x_S, a verifier's public key y_V and a message m, produces a signature δ.
– *Ver*: A deterministic polynomial time algorithm, on input the system parameters *sp*, a public key y_S of a signer, a private key x_V of a verifier, a message m, and a signature δ, produces a verification decision.
– *Sim*: A probabilistic polynomial time algorithm, on input the system parameters *sp*, a public key y_S of a signer, a private key x_V of a verifier, and a message m, produces a signature δ.

Properties

We consider four properties of a SDVS scheme, namely unforgeability, non-transferability, privacy of signer's identity, and non-delegatability. The following definitions mainly refer to [10,29].

– Unforgeability: The unforgeability means that if an adversary can produce a signature related to a signer and a verifier, and if it knows no private keys of the signer or verifier, it can be used as a black box to solve a hard problem. As the hard problem is not easy to be solved, the premise is false so the adversary cannot produce a valid signature. The concept is formally defined by a game between an adversary $\mathcal{A}$ and a simulator $\mathcal{S}$:

- $\mathcal{S}$ provides $\mathcal{A}$ system parameters sp, a public key y_S, and a public key y_V.
- $\mathcal{A}$ adaptively issues queries to the following oracles for polynomially many times:
  - Σ: Given a message m, it returns a valid signature δ with respect to y_S and y_V.
  - Υ: Give a signature δ on a message m, it returns a decision about its validity with respect to y_S and y_V.
- $\mathcal{A}$ produces a forgery δ^* for a message m^*. It wins the game if the signature is valid for m^* with respect to y_S and y_V, and it has not queried the message m^* to oracle Σ.

Definition 1. *An SDVS scheme is (ϵ, t)-unforgeable, if no adversary $\mathcal{A}$ wins the game with a probability at least ϵ in time at most t.*

– Non-transferability: The non-transferability means that given a valid message-signature pair (m, δ) for a designated verifier, it is infeasible for any probabilistic polynomial-time distinguisher to tell the message was signed by a signer or the designated verifier. Then a verifier cannot transfer a signature to a third party to convince the party that the signature was produced by a signer. The concept is formally defined as follows:

Definition 2. *An SDVS scheme is non-transferable if signatures produced by a signer are computationally indistinguishable from those produced by a designated verifier, i.e.*

$$\{Sign(sp, x_S, y_V, m)\} \approx \{Sim(sp, y_S, x_V, m)\}. \tag{1}$$

If the distributions of the two sets are identical, it is perfect non-transferable.

– Privacy of Signer's Identity: It considers two signers who produce signatures for a designated verifier. Basically, it requires that given a message-signature pair (m, δ), a distinguisher without the private key of the designated verifier, cannot tell it is produced by which signer. The concept is formally defined by a game between a distinguisher $\mathcal{D}$ and a simulator $\mathcal{C}$:

- $\mathcal{C}$ provides system parameters sp, two signers' public keys y_{S0} and y_{S1}, and a verifier's public key y_V.
- $\mathcal{D}$ adaptively issues queries to the following oracles for polynomially many times:
  - Σ_0 or Σ_1: Given a message m, it returns a valid signature δ with respect to y_{S0} and y_V or to y_{S1} and y_V.
  - Υ: Given a message m, a signature δ, and an identity $Sd \in \{S0, S1\}$, it returns a decision about its validity with respect to y_{Sd} and y_V.
- $\mathcal{D}$ produces a message m^*. $\mathcal{C}$ then flips a fair coin $b^* \leftarrow \{0, 1\}$, and computes a challenge signature $\delta^* = Sign(sp, x_{Sb^*}, y_V, m^*)$. It returns δ^* to $\mathcal{D}$.

  - $\mathcal{D}$ continues to issue queries as before except that it could not query Υ on input $(m^*, \delta^*, S0)$ or $(m^*, \delta^*, S1)$. Finally, it produces a bit b' and wins the game if $b' = b^*$.

Definition 3. *An SDVS scheme is (ϵ, t) secure about privacy of signer's identity if no distinguisher $\mathcal{D}$ wins the game above with probability that deviates from one-half by more than ϵ in time at most t.*

– Non-delegatability: Tian et al. [29] proposed an instinctive version of the original non-delegatability concept proposed by Lipmaa et al. [21]. Basically, the non-delegatability concept means that if an entity can create a valid signature, it knows the private key of the signer or the verifier. They then proposed a direct model that if an extractor interacted with a signature producer, the extractor could extract a private key of a signer or a verifier. It is formally defined by a game between an extractor $\mathcal{K}$ and a signature producer $\mathcal{F}$.

  - $\mathcal{K}$ produces system parameters sp, and sends it to $\mathcal{F}$.
  - $\mathcal{F}$ produces a public key $y_{S(V)}$, and sends it to $\mathcal{K}$.
  - $\mathcal{K}$ produces the other public key $y_{V(S)}$ and sends it to $\mathcal{F}$.
  - $\mathcal{F}$ produces a valid SDVS for a message m queried by $\mathcal{K}$ with a non-negligible probability. $\mathcal{F}$ accesses to at least a signing oracle and random oracles if $\mathcal{F}$ produces y_V, or accesses to at least signing and verifying oracles and random oracles if $\mathcal{F}$ produces y_S.
  - $\mathcal{K}$ produces $x_{S(V)}$.

Definition 4. *An SDVS scheme is $(\epsilon, t, \epsilon', t')$-non-delegatable if $\mathcal{K}$ can extract the private key in time t with a probability ϵ when $\mathcal{F}$ can produce a signature in time t' with a probability ϵ' in the above game, where $\epsilon > poly_1(\epsilon')$ and $t < poly_2(t')$ for two polynomial functions $poly_1$ and $poly_2$.*

3 The SDVS Scheme

– *Setup*: Let $\mathbb{G}$ and $\mathbb{G}_T$ be two cyclic groups with a large-prime order q. Let g be the generator of $\mathbb{G}$. Let $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ be a bilinear map. Let E be a symmetric encryption algorithm that takes an element in $\mathbb{G}$ as input, and produces a ciphertext in $\mathbb{G}$ with an encryption key $k \in \mathbb{Z}_q$. Let E^{-1} be its corresponding decryption algorithm. Let $H_1 : \{0,1\}^* \rightarrow \mathbb{G}$ and $H_2 : \{0,1\}^* \rightarrow \mathbb{Z}_q$ be secure hash functions. The system parameters are $sp = (\mathbb{G}, \mathbb{G}_T, g, q, e, E, E^{-1}, H_1, H_2)$.
– *KeyGen*: A signer has a private key $x_S \in_R \mathbb{Z}_q$ and a verification key $y_S = g^{x_S}$. A verifier has a private key $x_V \in_R \mathbb{Z}_q$, and a public key $y_V = g^{x_V}$.
– *Sign*: To sign a message m for a designated verifier V, a signer S does as follows:

  1. Choose at random $r \in_R \mathbb{Z}_q$, and compute $\delta_1 = y_V^r$.
  2. Compute $R = H_1(m)$, and $k = H_2(m)$.

3. Compute $\delta_2 = (\frac{R}{E_k(g^r)})^{x_S}$.
4. Set $\delta = (\delta_1, \delta_2)$.

- *Ver*: After receiving a signature $\delta = (\delta_1, \delta_2)$ and a message m from a signer S, a verifier V does as follows:
  1. Compute $R' = H_1(m)$, and $k' = H_2(m)$.
  2. Compute $T = \frac{R'}{E_{k'}(\delta_1^{x_V^{-1}})}$.
  3. Check if $e(g, \delta_2) = e(y_S, T)$.
- *Sim*: To simulate a signature on m, the verifier does as follows:
  1. Choose at random $r \in_R \mathbb{Z}_q$, and compute $\delta_2 = y_S^r$.
  2. Compute $R = H_1(m)$, and $k = H_2(m)$.
  3. Compute $\delta_1 = (E_k^{-1}(\frac{R}{g^r}))^{x_V}$.
  4. Set $\delta = (\delta_1, \delta_2)$.

Remark 2. For a valid signature, in the verification algorithm, it should be $T = \delta_2^{x_S^{-1}}$. Suppose the signature is produced by a signer, then

$$T = \frac{R'}{E_{k'}(\delta_1^{x_V^{-1}})} = \frac{R'}{E_{k'}(y_V^{r\, x_V^{-1}})} = \frac{R}{E_k(g^r)} = \delta_2^{x_S^{-1}}. \tag{2}$$

If the signature is produced by a simulator, then

$$T = \frac{R'}{E_{k'}(\delta_1^{x_V^{-1}})} = \frac{R'}{E_{k'}(((E_k^{-1}(\frac{R}{g^r}))^{x_V})^{x_V^{-1}})} = g^r = \delta_2^{x_S^{-1}}. \tag{3}$$

Remark 3. The symmetric encryption and decryption algorithms E and E^{-1} are used in proofs as random oracles. This is mainly to prove the property of PSI, where random oracles are used to extract a Diffie-Hellman value. We give a sample implementation of the encryption in Appendix A.

4 Proofs

We use the following symbols. Let q_h be the number of query on a hashing oracle H_1. Let q_e be the number of query on an encryption oracle E. Let q_d be the number of query on a decryption oracle E^{-1}. Let q_s be the number of query on a signing oracle Σ. Let q_v be the number of query on a verifying oracle Υ. Let τ_e denote the time of an exponentiation in a group $\mathbb{G}$. Finally, let τ_p denote the time of a pairing evaluation.

4.1 Unforgeability

Proposition 1. *If the CDH problem is (ϵ, t)-holding, and the hashing function, symmetric encryption and decryption algorithms are random oracles, the SDVS scheme is (ϵ', t') unforgeable, where*

$$\epsilon' < 3 q_h q_d \epsilon \tag{4}$$

and

$$t' > t - (q_h + 5q_s + (q_e + q_d)(2q_v + 1))\tau_e - 4q_v(q_e + q_d)\tau_p. \quad (5)$$

Proof. Suppose an adversary $\mathcal{A}$ who can produce valid SDVS signatures. It takes public keys of a signer and a verifier, and queries a signing oracle Σ and a verifying oracle Υ. Suppose a simulator $\mathcal{S}$ that provides inputs and oracles for $\mathcal{A}$. The simulator also provides $\mathcal{A}$ hashing oracles H_1 and H_2, an encryption oracle E, and a decryption oracle E^{-1}. Adversary $\mathcal{A}$ has to ask these oracles when it needs such operations. The simulator tries to solve a CDH problem. It takes an instance (g, g^α, g^β). Then $\mathcal{S}$ plays with $\mathcal{A}$ as follows:

- Simulator $\mathcal{S}$ selects at random $\gamma \in_R \mathbb{Z}_q$, and sets $y_S = g^\alpha$, $y_V = g^{\alpha\gamma}$.
- The simulator provides oracles H_1, H_2, E, and E^{-1} by maintaining four lists H_{list}, K_{list}, E_{list} and E^{-1}_{list}.
  - H_1: On a query m_i, $\mathcal{S}$ finds it in the H_{list}. If the message m_i has not been queried, $\mathcal{S}$ selects at random $r_i \in_R \mathbb{Z}_q$, and computes $R_i = g^{r_i}$. It records (m_i, r_i, R_i) in the H_{list}, and returns R_i. If the message m_i has been queried, it returns R_i in a matching entry directly. For a random selected integer $i^* \in_R \{1, \ldots, q_h\}$, $\mathcal{S}$ simply records $(m_{i^*}, \perp, g^\beta)$ and returns g^β.
  - H_2: On a query m_i, $\mathcal{S}$ finds it in the K_{list}. If the message m_i has not been queried, $\mathcal{S}$ selects at random $k_i \in_R \mathbb{Z}_q$, and records (m_i, k_i) in the K_{list}, and returns k_i. If the message m_i has been queried, it returns k_i in a matching entry directly.
  - E: On a query (k_i, U_i), $\mathcal{S}$ finds it in the E_{list} and E^{-1}_{list}. If the pair (k_i, U_i) has not been queried to the encryption oracle, and the value U_i not been returned as a decryption output by using the key k_i, it selects at random $e_i \in_R \mathbb{Z}_q$, and computes $W_i = g^{e_i}$. It records (k_i, U_i, e_i, W_i) in the E_{list}, and returns W_i. Else, it returns W_i in a matching entry of E_{list}, or the second element of a query in a matching entry of E^{-1}_{list}.
  - E^{-1}: On a query (k_i, P_i), $\mathcal{S}$ finds it in E_{list} and E^{-1}_{list}. If the pair (k_i, P_i) has not been queried to the decryption oracle, and the value P_i not been returned as an encryption output by using the key k_i, it selects at random $d_i \in_R \mathbb{Z}_q$, and computes $Q_i = g^{d_i}$. It records (k_i, P_i, d_i, Q_i) in a list E^{-1}_{list}, and returns Q_i. Else, it returns Q_i in a matching entry of E^{-1}_{list}, or the second element of a query in a matching entry of E_{list}. For a random selected integer $j^* \in_R \{1, \ldots, q_d\}$, $\mathcal{S}$ computes $g^{\gamma^{-1}\beta}$, simply records $(k_{j^*}, P_{j^*}, \perp, g^{\gamma^{-1}\beta})$, and returns $g^{\gamma^{-1}\beta}$.
- Simulator $\mathcal{S}$ provides a signing oracle Σ and a verifying oracle Υ by using the above lists.
  - Σ: On input a message m_i, $\mathcal{S}$ finds it in the H_{list}. If it finds nothing, it selects at random $r_i \in_R \mathbb{Z}_q$, and computes $R_i = g^{r_i}$. It records (m_i, r_i, R_i) in the H_{list}. Else, it uses a matching entry directly. Similarly, $\mathcal{S}$ obtains k_i by using the K_{list}. Then it selects at random $r_l \in_R \mathbb{Z}_q$, and computes $U_i = g^{r_l}$. If (k_i, U_i) is in the E_{list} as a query or the value U_i in the

E^{-1}_{list} as a response by using the decryption key k_i, another r_l is selected. Then it selects at random $e_i \in_R \mathbb{Z}_q$, computes $W_i = g^{e_i}$ and records an entry (k_i, U_i, e_i, W_i) in the E_{list}. Finally, it computes $\delta_1 = y_V^{r_l}$, and $\delta_2 = y_S^{r_i - e_i}$, and returns $\delta = (\delta_1, \delta_2)$.

- Υ: On input a signature $\delta = (\delta_1, \delta_2)$ for a message m_i, $\mathcal{S}$ finds a matching entry in H_{list}, and an entry in K_{list}. If there is no matching in H_{list} or K_{list}, $\mathcal{S}$ claims the signature invalid. Else, suppose that the two matching entries are (m_i, r_i, R_i) and (m_i, k_i). Then $\mathcal{S}$ finds all entries indexed by k_i in lists E_{list} and E^{-1}_{list}. If it finds nothing, it claims the signature invalid. Else,

  1. Among all entries found in E_{list}, $\mathcal{S}$ finds entries containing a value e_i satisfying $y_S^{r_i - e_i} = \delta_2$. Then among all entries containing the value e_i, it finds a value U_i satisfying $e(U_i, y_V) = e(g, \delta_1)$. If it finds such an entry (k_i, U_i, e_i, W_i), it claims the signature valid. Otherwise, it does as follows.
  2. Among all entries found in E^{-1}_{list}, $\mathcal{S}$ finds entries containing a value d_i satisfying $y_V^{d_i} = \delta_1$. Then among all entries containing the value d_i, it finds a value P_i satisfying $e(\frac{R_i}{P_i}, y_S) = e(g, \delta_2)$. If it finds such an entry (k_i, P_i, d_i, Q_i), it claims the signature valid. Otherwise, the signature is invalid.

If $\mathcal{A}$ produces a forged signature for a message m_j, $\mathcal{S}$ will check whether $j = i^*$. If it is not, $\mathcal{S}$ *Fails*. Else, $\mathcal{S}$ finds related entries in H_{list}, K_{list}. Suppose them be $(m_{i^*}, \perp, R_{i^*})$ and (m_{i^*}, k_{i^*}). Then indexed by the value k_{i^*}, $\mathcal{S}$ finds all entries in E_{list}. Among these entries, $\mathcal{S}$ finds a value U_{i^*} satisfying $e(U_{i^*}, y_V) = e(g, \delta_1)$. If it finds such an entry $(k_{i^*}, U_{i^*}, e_{i^*}, W_{i^*})$, it computes an answer $(\delta_2 y_S^{e_{i^*}})$. If the forged signature is valid, it should be $g^{\alpha\beta}$. If $\mathcal{S}$ finds nothing in E_{list}, it checks whether $k_{i^*} = k_{j^*}$ and $e(\frac{R_{i^*}}{P_{j^*}}, y_S) = e(g, \delta_2)$. If it is not, $\mathcal{S}$ *Fails*. Else, $\mathcal{S}$ takes δ_1 as an answer. If the signature is valid, it should be $g^{\alpha\beta}$.

Considering the simulation, it is possible that $\mathcal{A}$ queries a valid signature, and $\mathcal{S}$ claims it invalid. It is the case that $\mathcal{A}$ produced a valid signature while it did not query a hashing oracle, or an encryption oracle, or a decryption oracle. The probability of the case is at most $1/q$. Then with a probability at least $1 - q_v/q$, the simulation is perfect, and $\mathcal{A}$ gives a forgery with a probability ϵ'.

When $\mathcal{A}$ gives a forgery, the case happens at most $1/q$ that it does not query oracles H_1, H_2, E or E^{-1}. Then the case $j = i^*$ happens with a probability $1/q_h$. Suppose $\mathcal{S}$ may find a matching entry in E_{list} with a probability $\eta \in [0, 1]$. Then with a probability $(1 - \eta)$, $\mathcal{A}$ may use the E^{-1} oracle for the forgery. Then with a probability $1/q_d$, $\mathcal{S}$ can take advantage over $\mathcal{A}$. Finally, the successful probability of $\mathcal{S}$ is:

$$\begin{aligned} \epsilon &\geq (1 - q_v/q)\epsilon'(1 - 1/q)(1/q_h)[\eta + (1 - \eta)(1/q_d)] \\ &> \epsilon'(1 - q_v/q)(1 - 1/q)(1/(q_h q_d)) \\ &> \frac{\epsilon'}{3q_h q_d} \end{aligned} \tag{6}$$

where $q > 2q_v$, and $q > 3$.

For each new query on H_1, $\mathcal{S}$ needs an exponentiation. For a query on E or E^{-1}, it also needs that operation. Then the time is $(q_h+q_e+q_d)\tau_e$ for simulating random oracles. For each signing query, $\mathcal{S}$ needs at most five exponentials. For a verifying query, $\mathcal{S}$ needs to check entries in E_{list} and E_{list}^{-1} if needed. Each check needs at most two exponentiations and four pairing evaluations. Then the time is less than $5q_s\tau_e + q_v(q_e+q_d)(2\tau_e+4\tau_p)$ for $\mathcal{S}$ to simulate signing and verifying oracles. Finally, the runtime of $\mathcal{S}$ is:

$$\begin{aligned} t &< t' + (q_h+q_e+q_d)\tau_e + 5q_s\tau_e + q_v(q_e+q_d)(2\tau_e+4\tau_p) \\ &< t' + (q_h+5q_s+(q_e+q_d)(2q_v+1))\tau_e + 4q_v(q_e+q_d)\tau_p \end{aligned} \tag{7}$$

4.2 Non-transferability

Proposition 2. *The SDVS scheme is perfect non-transferable.*

Proof. We prove that for any valid signature $\hat{\delta} = (\hat{\delta_1}, \hat{\delta_2})$ on a message m with respect to a signer and a designated verifier, it may be produced by a signing algorithm or a simulation algorithm with the same probability. Then, the distribution of a signature ensemble produced by the signing algorithm is the same as an ensemble produced by the simulation algorithm.

If a signature $\delta = (\delta_1, \delta_2)$ is produced by a signing algorithm, it is

$$\begin{pmatrix} \delta_1 \\ \delta_2 \end{pmatrix} = \begin{pmatrix} y_V^r \\ \left(\frac{H_1(m)}{E_{H_2(m)}(g^r)}\right)^{x_S} \end{pmatrix} \tag{8}$$

for a random selected $r \in_R \mathbb{Z}_q$. The case $\hat{\delta} = \delta$ happens with a probability $1/q$.

If a signature $\delta_s = (\delta_{s1}, \delta_{s2})$ is produced by a simulation algorithm, it is

$$\begin{pmatrix} \delta_{s1} \\ \delta_{s2} \end{pmatrix} = \begin{pmatrix} \left(E_{H_2(m)}^{-1}\left(\frac{H_1(m)}{g^r}\right)\right)^{x_V} \\ y_S^r \end{pmatrix} \tag{9}$$

for a random selected $r \in_R \mathbb{Z}_q$. The case $\hat{\delta} = \delta_s$ happens with the same probability $1/q$.

4.3 Privacy of Signer's Identity (PSI)

Proposition 3. *If the InvCDH problem is (ϵ, t)-holding, and the hashing function, symmetric encryption and decryption algorithms are random oracles, the SDVS scheme is (ϵ', t') secure about the PSI property, where*

$$\epsilon' < 6q_{max}\epsilon \tag{10}$$

and

$$t' > t - (2+q_h+5q_s+(q_e+q_d)(2q_v+1))\tau_e - 4q_v(q_e+q_d)\tau_p \tag{11}$$

where $q_{max} \le max\{q_e, q_d\}$.

Proof. Suppose a distinguisher $\mathcal{D}$ to distinguish a real signer of a given SDVS signature. $\mathcal{D}$ takes two signer's public keys and a verifier's public key as input, and queries two signing oracles Σ_0 and Σ_1, and a verifying oracle Υ. Suppose a simulator $\mathcal{C}$ provides these oracles and inputs to $\mathcal{D}$. It also provides two hashing oracles H_1, H_2, an encryption oracle E, and a decryption oracle E^{-1}. $\mathcal{D}$ has to consult $\mathcal{C}$ when it needs these operations. $\mathcal{C}$ tries to solve an InvCDH problem. It takes an instance (g, g^{α}). Then $\mathcal{C}$ plays with $\mathcal{D}$ as follows.

- $\mathcal{C}$ selects at random $\beta_0, \beta_1 \in_R \mathbb{Z}_q$, and computes $y_{S0} = g^{\alpha\beta_0}$, and sets $y_{S1} = g^{\alpha\beta_1}$. It sets $y_V = g^{\alpha}$, and feeds all public keys to $\mathcal{D}$.
- $\mathcal{C}$ simulates hashing oracles H_1 and H_2, an encryption oracle E, and a decryption oracle E^{-1} by using four lists H_{list}, K_{list}, E_{list}, and E^{-1}_{list}.
  - H_1: On a query m_i, $\mathcal{C}$ finds it in the H_{list}. If the message m_i has not been queried, $\mathcal{C}$ selects at random $r_i \in_R \mathbb{Z}_q$, and computes $R_i = g^{r_i}$. It records (m_i, r_i, R_i) in the H_{list}, and returns R_i. If the message m_i has been queried, it returns R_i in a matching entry directly.
  - H_2 and E: $\mathcal{C}$ simulates them in the same way as $\mathcal{S}$ in the unforgeability proof.
  - E^{-1}: On a query (k_i, P_i), $\mathcal{S}$ finds it in E_{list} and E^{-1}_{list}. If the pair (k_i, P_i) has not been queried to the decryption oracle, and the value P_i not been returned as an encryption output by using the key k_i, it selects at random $d_i \in_R \mathbb{Z}_q$, and computes $Q_i = g^{d_i}$. It records (k_i, P_i, d_i, Q_i) in a list E^{-1}_{list}, and returns Q_i. Else, it returns Q_i in a matching entry of E^{-1}_{list}, or the second element of a query in a matching entry of E_{list}.
- $\mathcal{C}$ provides $\mathcal{D}$ signing oracles Σ_0 and Σ_1, and a verifying oracle Υ in the same way as $\mathcal{S}$ in the unforgeability proof with an exception that the signer's public key is y_{S0} for Σ_0, y_{S1} for Σ_1, and y_{id} for Υ, where $id \in \{S0, S1\}$ is an indicator from $\mathcal{D}$.
- When $\mathcal{D}$ submits a message m^*, $\mathcal{C}$ randomly selects a value $\phi^* \in_R \mathbb{Z}_q$. And it flips a fair coin $b^* \in \{0, 1\}$. It also selects at random r^*, computes $R^* = g^{r^*}$, and adds an entry (m^*, r^*, R^*) to H_{list}. Then it randomly selects k^* and adds an entry in (m^*, k^*) in K_{list}. Then with a probability 1/2, it chooses one case from the two:
  1. It sets $\delta_1^* = g^{\phi^*}$. It then randomly selects e^*, computes $W^* = g^{e^*}$, and adds an entry $(k^*, \perp, e^*, W^*)$ in E_{list}. It computes $\delta_2 = y_{Sb^*}^{r^*-e^*}$. Finally, it returns $\delta^* = (\delta_1^*, \delta_2^*)$.
  2. It sets $\delta_2^* = g^{\phi^*}$. Then it randomly selects d^*, computes $Q^* = g^{d^*}$, and adds an entry $(k^*, \perp, d^*, Q^*)$ in E^{-1}_{list}. It computes $\delta_1^* = y_V^{d^*}$. Finally, it returns $\delta^* = (\delta_1^*, \delta_2^*)$.
- Then $\mathcal{D}$ continues to query various oracles. The behaviors of oracles H_1 and H_2 keep unchanged. The oracles Σ_0 and Σ_1 now add an extra mark for each entry added by them to E_{list} or E^{-1}_{list}. The oracle Υ does not respond to a query with the signature δ^* on the message m^* with an indicator $id \in \{S0, S1\}$. The behaviors of E and E^{-1} have the following changes:

- E: If a query is (k^*, Q^*), $\mathcal{C}$ tries to find matching entries in E_{list}^{-1} except the entry $(k^*, \perp, d^*, Q^*)$. If it finds nothing, $\mathcal{C}$ *Fails*. If a query begins with k^*, say (k^*, U_j), and $U_j \neq Q^*$, $\mathcal{C}$ finds matching entries in E_{list} and E_{list}^{-1}. If it finds nothing, it will add a new entry (k^*, U_j, e_j, W_j) and return W_j, where W_j is selected at random and $W_j \neq W^*$. Suppose there are q_e^* queries beginning with k^* and no matching entries in both E_{list} and E_{list}^{-1}. Then for a random selected integer $j^* \in_R \{1, \ldots, q_e^*\}$, $\mathcal{C}$ sets $e_{j^*} = e^*$ and $W_{j^*} = W^*$.
- E^{-1}: If a query is (k^*, W^*), $\mathcal{C}$ finds matching entries in E_{list} except the entry $(k^*, \perp, e^*, W^*)$. If it finds nothing, $\mathcal{C}$ *Fails*. If a query begins with k^*, say (k^*, P_j), and $P_j \neq W^*$, $\mathcal{C}$ finds matching entries in E_{list}^{-1} and E_{list}. If it finds nothing, it will add a new entry (k^*, P_j, d_j, Q_j) and returns Q_j. Suppose there are q_d^* queries beginning with k^* and no matching entries in both E_{list} and E_{list}^{-1}. Then for a random selected integer $j_d^* \in_R \{1, \ldots, q_d^*\}$, $\mathcal{C}$ sets $d_{j_d^*} = d^*$ and $Q_{j_d^*} = Q^*$.

When $\mathcal{D}$ produces a bit b', $\mathcal{C}$ finds an entry (k^*, U_{j^*}, e^*, W^*) in E_{list} or an entry (k^*, P_{j^*}, d^*, Q^*) in E_{list}^{-1}. If it finds nothing, $\mathcal{C}$ *Fails*. If it finds such an entry in E_{list}, $\mathcal{C}$ returns $U_{j^*}^{\phi^{*-1}}$ as an answer. If it finds such an entry in E_{list}^{-1} , it computes $\left((\frac{R^*}{P_{j^*}})^{\phi^{*-1}}\right)^{\beta_{b'}}$ as an answer.

Similar to the analysis in the unforgeability proof about the verifying oracle Υ, the simulation about the oracle is perfect with a probability $1 - q_v/q$. When $\mathcal{D}$ gives a meaningful guess bit, the event happens with a probability $1/q$ that $\mathcal{D}$ queries neither (k^*, U_{j^*}) nor (k^*, P_{j^*}). The reason is that W^* or Q^* are chosen randomly by $\mathcal{C}$. When $\mathcal{D}$ did query E or E^{-1}, there are two cases:

1. $\mathcal{C}$ finds an entry (k^*, U_{j^*}, e^*, W^*) in E_{list}. Suppose this event happens with a probability $\eta \in [0, 1]$. Then with a probability $1/q_e^*$, the simulation is perfect and $\mathcal{D}$ can produce a meaningful output. No matter $\mathcal{D}$'s guess, $\mathcal{C}$ can return a right answer if $\mathcal{C}$ produced the challenge signature by setting $\delta_1^* = g^{\phi^*}$.
2. $\mathcal{C}$ finds an entry (k^*, P_{j^*}, d^*, Q^*) in E_{list}^{-1}. This event happens with a probability $1 - \eta$. Then with a probability $1/q_d^*$, the simulation is perfect and $\mathcal{D}$ can produce a meaningful output. When $\mathcal{D}$ guessed correctly, $\mathcal{C}$ can return a right answer if $\mathcal{C}$ produced the challenge signature by setting $\delta_2^* = g^{\phi^*}$.

In summary, the successful probability of $\mathcal{C}$ is

$$\begin{aligned}\epsilon &\geq (1 - q_v/q)(1 - 1/q)[\eta(1/q_e^*)(1/2) + (1 - \eta)(1/q_d^*)\epsilon'(1/2)] \\ &> \epsilon'/6[\eta(1/q_e^*) + (1 - \eta)(1/q_d^*)] \\ &\geq \epsilon'/6[\eta(1/q_{max}) + (1 - \eta)(1/q_{max})] \\ &= \frac{\epsilon'}{6q_{max}}\end{aligned} \tag{12}$$

where $q > 2q_v$, $q > 3$ and $q_{max} = max\{q_e^*, q_d^*\}$.

The runtime analysis is similar to the unforgeability proof except that there is an extra two exponentiations for a challenge signature generation. It is

$$\begin{aligned}t &< t' + (q_h + q_e + q_d)\tau_e + 5q_s\tau_e + q_v(q_e + q_d)(2\tau_e + 4\tau_p) + 2\tau_e \\ &< t' + (2 + q_h + 5q_s + (q_e + q_d)(2q_v + 1))\tau_e + 4q_v(q_e + q_d)\tau_p\end{aligned} \tag{13}$$

4.4 Non-delegatability

Proposition 4. *If the KEAv1 assumption holds with a probability* $1-\epsilon''$, *the SDVS scheme is* $(\epsilon, t, \epsilon', t')$ *non-delegatable, where*

$$\epsilon > \frac{1-\epsilon''}{3}\epsilon' \tag{14}$$

and

$$t < t' + (q_h + 5q_s + (q_e + q_d)(2q_v + 1))\tau_e + 4q_v(q_e + q_d)\tau_p. \tag{15}$$

Proof. Suppose an adversary $\mathcal{F}$ that may obtain some partial secrets to produce a valid signature with a probability ϵ' in time t'. To confirm that $\mathcal{F}$ indeed has signer's secret key or verifier's secret key, suppose an extractor $\mathcal{K}$ that provides random oracles, signing oracles and verifying oracles. If $\mathcal{F}$ gives a signature for a message m, $\mathcal{K}$ tries to print signer's or verifier's secret key.

- Case 1:
  1. $\mathcal{K}$ provides $\mathcal{F}$ system parameters $(\mathbb{G}, \mathbb{G}_T, g, q, e)$.
  2. $\mathcal{F}$ produces y_S and sends it to $\mathcal{K}$.
  3. $\mathcal{K}$ selects at random $\gamma \in \mathbb{Z}_q$, computes $y_V = y_S^{\gamma}$ and sends it to $\mathcal{F}$.
  4. $\mathcal{K}$ provides hashing oracles, an encryption oracle and a decryption oracle in a similar way as $\mathcal{S}$ in the unforgeability proof except that no special values i^* or j^* are selected. $\mathcal{K}$ provides a signing oracle and a verifying oracle in the same way as $\mathcal{S}$.
  5. $\mathcal{K}$ sends a message m^* to $\mathcal{F}$.
  6. $\mathcal{F}$ produces a signature δ^* for the message m^*.
  7. $\mathcal{K}$ finds the entry (m^*, r^*, R^*) in H_{list}, and (m^*, k^*) in K_{list}. Then, there are two cases:

     - Among all entries found from E_{list} by indexing k^*, $\mathcal{K}$ finds entries containing a value e^* satisfying $y_S^{r^*-e^*} = \delta_2^*$. Then among all entries containing the value e^*, it finds a value U^* satisfying $e(U^*, y_V) = e(g, \delta_1^*)$. If it finds such an entry (k^*, U^*, e^*, W^*) in E_{list}, $\mathcal{K}$ believes the signature valid, and takes $\mathcal{F}$ as an algorithm with an input $g^{r^*-e^*}$ and outputs $(y_S, y_S^{r^*-e^*})$. $\mathcal{K}$ then builds another algorithm $\mathcal{F}^*$ with the same random tape and inputs. According to the KEAv1 assumption, it is expected that $\mathcal{F}^*$ produces outputs $(x_S, y_S, y_S^{r^*-e^*})$. However, if $\mathcal{K}$ does not find such an entry in E_{list}, it does as follows.
     - Among all entries found in E_{list}^{-1} by indexing k^*, $\mathcal{K}$ finds entries containing a value d^* such that $y_V^{d^*} = \delta_1^*$. Then among all entries containing the value d^*, it finds a value P^* satisfying $e(\frac{R^*}{P^*}, y_S) = e(g, \delta_2^*)$. If it finds such an entry (k^*, P^*, d^*, Q^*) in E_{list}^{-1}, it believes the signature valid, and takes $\mathcal{F}$ as an algorithm with input $g^{\gamma d^*}$ and outputs (y_S, δ_1^*). $\mathcal{K}$ then builds another algorithm $\mathcal{F}^*$ with the same random tape and inputs. According to the KEAv1 assumption, it is expected that $\mathcal{F}^*$ produces outputs (x_S, y_S, δ_1^*).

- Case 2:
  1. $\mathcal{K}$ provides $\mathcal{F}$ system parameters $(\mathbb{G}, \mathbb{G}_T, g, q, e)$.
  2. $\mathcal{F}$ produces y_V and sends it to $\mathcal{K}$.
  3. $\mathcal{K}$ selects at random $\gamma \in \mathbb{Z}_q$, computes $y_S = y_V^\gamma$ and sends it to $\mathcal{F}$.
  4. $\mathcal{K}$ queries $\mathcal{F}$ and provides it various oracles in the same way as in the case 1.
  5. It is similar to last step of the case 1. The difference is that if $\mathcal{K}$ finds a satisfying entry in E_{list}, it takes $\mathcal{F}$ as an algorithm with an input $g^{\gamma(r^*-e^*)}$ and outputs $(y_V, y_V^{\gamma(r^*-e^*)})$. If $\mathcal{K}$ finds an entry in E_{list}^{-1}, it takes $\mathcal{F}$ as an algorithm with an input g^{d^*} and outputs $(y_V, y_V^{d^*})$.

The simulation of $\mathcal{K}$ is perfect with a probability at least $1-q_v/q$ as it is similar to $\mathcal{S}$'s simulation in the unforgeability proof. $\mathcal{F}$ may produce a signature without querying E or E^{-1} with a probability $1-1/q$. After $\mathcal{F}$ produced a signature, $\mathcal{K}$ can extract a private key if and only if the KEAv1 assumption holds. The probability is $1-\epsilon''$. In summary, for any public key produced by $\mathcal{F}$, $\mathcal{K}$ can extract its corresponding private key with a probability

$$\begin{aligned}\epsilon &\geq (1-q_v/q)\epsilon'(1-1/q)(1-\epsilon'') \\ &> \tfrac{1}{3}\epsilon'(1-\epsilon'').\end{aligned} \tag{16}$$

The runtime of $\mathcal{K}$ to extract a private key is the same as $\mathcal{S}$'s runtime in the unforgeability proof. It is

$$t < t' + (q_h + 5q_s + (q_e + q_d)(2q_v + 1))\tau_e + 4q_v(q_e + q_d)\tau_p. \tag{17}$$

5 Comparison

We compare our scheme with non-delegatable DVS schemes mentioned in Section 1.2. There are nine related schemes, including DVS, SDVS, IBDVS schemes, and a universal DVS (UDVS) scheme. We show their signature size, signing cost and verifying cost. The proof models of these schemes are mainly random oracle (RO) model. An exception is the scheme in [21] whose proof model is non-programmable random oracle (NPRO) model. The unforgeability of most schemes are reduced to the hardness of a CDH problem or its elliptic curve version (ECDHP). The "DDH" denotes the decisional Diffie-Hellman problem. It is the weakest hard problem in the hard problem column. The "DL" denotes the discrete logarithm problem. It is the strongest hard problem in that column. The "GBDH" means the gap bilinear Diffie-Hellman problem. As the scheme in [14] had no proofs, we use the symbol "-" to denote the fact. The scheme in [8] is just an instantiation of their ring signature based construction. Although it has no proofs, we can deduce its proof model and hard problem from their general proof.

As the signature size and computation cost are related to a scheme's system parameters, we list some system parameters below. There are three kinds of parameters:

1. Let $(\mathbb{G}, \mathbb{G}_T, g, q, e)$ be symbols defined in this paper.
2. Let p' and q' be large primes such that $q'|p'-1$. Let $\mathbb{G}'$ be a subgroup with an order q' of a group $\mathbb{Z}_{p'}^*$.

3. Let $(E, \mathbb{G}_E, P, n)$ be parameters for a group $\mathbb{G}_E$ on a non-supersingular elliptic curve E with a generator point P and a large-prime order n.

The symbols related to the comparison of signature size are as follows. Let $|\cdot|$ denote the bit-length of the element "$\cdot$". Considering a cryptographic strength of approximate 80 security bits, by using the parameter set in [2], we have that $|g| = 154$, $|q| = 151$, and $|g_T| \approx 923$ for an element $g_T \in \mathbb{G}_T$. For the same security level, $|p'| = 1024$, $|q'| = 160$, and $|n| = 160$. If an element is in $\mathbb{G}_E$, its bit-length is deemed as 161.

The symbols related to computation cost are as follows. As we use the parameter set in [2], the symbol τ_e is the time of a scalar multiplication. We use another symbol τ_n for the time of a scalar multiplication in the group $\mathbb{G}_E$. The symbol τ_p still denotes the time of a paring evaluation. A new symbol τ' is used to denote the time of an exponentiation in $\mathbb{G}'$ or $\mathbb{G}_T$. Note that the computation of multi-exponentiation has accelerative algorithms. From a repot in [22], for $g_1, g_2, g_3 \in \mathbb{G}$ and $e_1, e_2, e_3 \in \mathbb{Z}_q$, the computation of $g_1^{e_1} g_2^{e_2}$ is about as costly as an exponentiation, and the computation of $g_1^{e_1} g_2^{e_2} g_3^{e_3}$ is about 1.5 times of an exponentiation.

Table 1. Comparison between non-delegatable DVS schemes

Scheme	Type	Signature-Size (bits)	Sign-Cost	Verify-Cost	Model	Hard Problem
[14]	DVS	2528 ($\mathbb{Z}_{q'}^3 \times \mathbb{G}'^2$)	$3\tau'$	$5\tau'$	-	-
[21]	DVS	640 ($\mathbb{Z}_{q'}^4$)	$3\tau'$	$3\tau'$	NPRO	DDH
[30]	IBDVS	1228 ($\mathbb{Z}_q \times \mathbb{G}^2 \times \mathbb{G}_T$)	$6\tau_e + 2\tau_p$	$2\tau_e + 3\tau_p$	RO	GBDH
[12]	UDVS	607 ($\mathbb{Z}_q^3 \times \mathbb{G}$)	$3\tau_e + \tau_p$	$\tau_e + 2\tau_p + \tau'$	RO	CDH
[10]	SDVS	640 ($\mathbb{Z}_{q'}^4$)	$3\tau'$	$3\tau'$	RO	DL
[9]	IBDVS	758 ($\mathbb{Z}_q^4 \times \mathbb{G}$)	$\tau_e + 3\tau_p + 3\tau'$	$4\tau' + 4\tau_p$	RO	CDH
[11]	IBSDVS	2607 ($\mathbb{Z}_q^3 \times \mathbb{G}^2 \times \mathbb{G}_T^2$)	$\tau_e + 4\tau' + 4\tau_p$	$4\tau' + 5\tau_p$	RO	CDH
[8]	SDVS	1504($\mathbb{Z}_{q'}^3 \times \mathbb{G}'$)	$2\tau'$	$2.5\tau'$	RO	DL
[29]	SDVS	481 ($\mathbb{Z}_n^2 \times \mathbb{G}_\mathbb{E}$)	$2\tau_n$	$2\tau_n$	RO	ECDHP
Ours	SDVS	308 ($\mathbb{G}^2$)	$2\tau_e$	$\tau_e + 2\tau_p$	RO	CDH

From the Table 1, we observe the following points:

– The signature size of our scheme is short. It is the only scheme consists of two elements in a signature. Only one scheme in [29] produces a signature with three elements. All other schemes produce signatures consisting of at least four elements.
– The signing and verifying costs of our scheme are moderate due to the cost of the scalar multiplication on elliptic curves.

Acknowledgment. This work is supported by the National Natural Science Foundation of China (Grant Nos. 60970144, 60803135, 61070168, 61003244, 61100224, 61103119), Fundamental Research Funds for the Central Universities (Grant Nos.10lgpy31, 11lgpy71, 11lgzd06, 10lgzd14), Doctoral Fund of Ministry of Education of China for New Teachers (Grant No. 20090171120006). We are grateful to the anonymous referees for their invaluable suggestions.

References

1. Bao, F., Deng, R.H., Zhu, H.: Variations of Diffie-Hellman Problem. In: Qing, S., Gollmann, D., Zhou, J. (eds.) ICICS 2003. LNCS, vol. 2836, pp. 301–312. Springer, Heidelberg (2003)
2. Boneh, D., Lynn, B., Shacham, H.: Short Signatures from the Weil Pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001)
3. Cao, F., Cao, Z.: An identity based universal designated verifier signature scheme secure in the standard model. Journal of Systems and Software 82(4), 643–649 (2009)
4. Cramer, R., Damgård, I.B., Schoenmakers, B.: Proof of Partial Knowledge and Simplified Design of Witness Hiding Protocols. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 174–187. Springer, Heidelberg (1994)
5. Damgård, I.: Towards Practical Public Key Systems Secure against Chosen Ciphertext Attacks. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 445–456. Springer, Heidelberg (1992)
6. Dent, A.W., Galbraith, S.D.: Hidden Pairings and Trapdoor DDH Groups. In: Hess, F., Pauli, S., Pohst, M. (eds.) ANTS 2006. LNCS, vol. 4076, pp. 436–451. Springer, Heidelberg (2006)
7. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
8. Feng, D., Xu, J., Chen, W.: Generic Constructions for Strong Designated Verifier Signature. Journal of Information Processing Systems 7(1), 159–172 (2011)
9. Huang, Q., Susil, W., Wong, D.: Non-delegatable Identity-based Designated Verifier Signature. Cryptology ePrint Archive: Report 2009/367 (2009)
10. Huang, Q., Yang, G., Wong, D., Susilo, W.: Efficient Strong Designated Verifier Signature Schemes without Random Oracles or Delegatability. Cryptology ePrint Archive: Report 2009/518 (2009)
11. Huang, Q., Yang, G., Wong, D., Susilo, W.: Identity-based strong designated verifier signature revisited. Journal of Systems and Software 84(1), 120–129 (2011)
12. Huang, X., Susilo, W., Mu, Y., Wu, W.: Universal Designated Verifier Signature Without Delegatability. In: Ning, P., Qing, S., Li, N. (eds.) ICICS 2006. LNCS, vol. 4307, pp. 479–498. Springer, Heidelberg (2006)
13. Huang, X., Susilo, W., Mu, Y., Zhang, F.: Short Designated Verifier Signature Scheme and Its Identity-based Variant. International Journal of Network Security 6(1), 82–93 (2008)
14. Jakobsson, M., Sako, K., Impagliazzo, R.: Designated Verifier Proofs and Their Applications. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 143–154. Springer, Heidelberg (1996)

15. Kancharla, P., Gummadidala, S., Saxena, A.: Identity Based Strong Designated Verifier Signature Scheme. Journal of Informatica 18(2), 239–252 (2007)
16. Kang, B., Boyd, C., Dawson, E.: A novel identity based strong designated verifier signature scheme. Journal of Systems and Software 82(2), 270–273 (2009)
17. Laguillaumie, F., Vergnaud, D.: Designated Verifier Signatures: Anonymity and Efficient Construction from *Any* Bilinear Map. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 105–119. Springer, Heidelberg (2005)
18. Laguillaumie, F., Vergnaud, D.: Multi-designated Verifiers Signatures. In: López, J., Qing, S., Okamoto, E. (eds.) ICICS 2004. LNCS, vol. 3269, pp. 495–507. Springer, Heidelberg (2004)
19. Lee, J., Chang, J.: Comment on Saeednia et al.'s strong designated verifier signature scheme. Journal of Computer Standards & Interfaces - CSI 31(1), 258–260 (2009)
20. Li, Y., Lipmaa, H., Pei, D.: On Delegatability of Four Designated Verifier Signatures. In: Qing, S., Mao, W., López, J., Wang, G. (eds.) ICICS 2005. LNCS, vol. 3783, pp. 61–71. Springer, Heidelberg (2005)
21. Lipmaa, H., Wang, G., Bao, F.: Designated Verifier Signature Schemes: Attacks, New Security Notions and a New Construction. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 459–471. Springer, Heidelberg (2005)
22. Möller, B.: Algorithms for Multi-exponentiation. In: Vaudenay, S., Youssef, A.M. (eds.) SAC 2001. LNCS, vol. 2259, pp. 165–180. Springer, Heidelberg (2001)
23. Ng, C., Susilo, W., Mu, Y.: Universal Designated Multi Verifier Signature Schemes. In: Xu, C., Yang, L. (eds.) SNDS 2005, pp. 305–309. IEEE, Fukuoka (2005)
24. Saeednia, S., Kramer, S., Markovitch, O.: An Efficient Strong Designated Verifier Signature Scheme. In: Lim, J.I., Lee, D.H. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 40–54. Springer, Heidelberg (2004)
25. Steinfeld, R., Bull, L., Wang, H., Pieprzyk, J.: Universal Designated-Verifier Signatures. In: Laih, C.S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 523–542. Springer, Heidelberg (2003)
26. Steinfeld, R., Wang, H., Pieprzyk, J.: Efficient Extension of Standard Schnorr/RSA Signatures into Universal Designated-Verifier Signatures. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 86–100. Springer, Heidelberg (2004)
27. Susilo, W., Zhang, F., Mu, Y.: Identity-Based Strong Designated Verifier Signature Schemes. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 313–324. Springer, Heidelberg (2004)
28. Sun, X., Li, J., Hu, Y., Chen, G.: Delegatability of an Identity Based Strong Designated Verifier Signature Scheme. INFORMATICA 21(1), 117–122 (2010)
29. Tian, H., Chen, X., Jiang, Z., Du, Y.: Non-delegatable Strong Designated Verier Signature on Elliptic Curves. In: ICISC 2011, Seoul, Korea (November 2011)
30. Wang, B.: A non-delegatable identity-based strong designated verifier signature scheme. Cryptology ePrint Archive: Report /2008/507 (2008)
31. Wu, J., Stinson, D.: An Efficient Identification Protocol and the Knowledge-of-Exponent Assumption. Cryptology ePrint Archive: Report 2007/479 (2007)
32. Zhang, R., Furukawa, J., Imai, H.: Short Signature and Universal Designated Verifier Signature Without Random Oracles. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 483–498. Springer, Heidelberg (2005)
33. Zhang, J., Geng, Q.: On the Security of Group Signature Scheme and Designated Verifier Signature Scheme. In: NAS 2008, pp. 351–358. IEEE, Chongqing (2008)
34. Zhang, J., Mao, J.: A novel ID-based designated verifier signature scheme. Information Sciences 178(3), 766–773 (2008)

A Encryption Algorithm Implementation

Note that the SDVS scheme uses an encryption in both the *Sign* and *Ver* algorithms. It uses a decryption algorithm in the *Sim* algorithm that produces a signature to be verified by a *Ver* algorithm. So it is not the case that one encrypts something that is to be decrypted by another entity. We may design an algorithm specially for the SDVS scheme.

We use a parameter set in BLS signature [2]. The parameter set includes an elliptic curve (EC) that is defined over a field $\mathbb{F}_{3^l}$ for a positive integer l. An additive group $\mathbb{G}$ is generated by a point on the EC with a large-prime order q. The total number of points on the EC is denoted by n. Let a point C on the curve is expressed as (c_x, b) where $c_x \in \mathbb{F}_{3^l}$ and $b \in \{0, 1\}$. Let $|3^l|$ be the binary bit-length of the value 3^l. Let the symbol $||$ denote bit concatenation. Let $(X)_2$ be the binary representation of the value X.

As a random number in $\mathbb{F}_{3^l}$ is a quadratic residue with a probability about $1/2$, we need an additional parameter i to be part of a signature. The length of the value i is a balance of the successful probability of the algorithm and the communication bandwidth. We set $i \in [0, 2^{32} - 1]$. Let an entity firstly set a value i at random. Then we define an function $f_k(C, i)$ as follows.

- $f_k(C, i)$:
  1. Set a value $t = 0$.
  2. Use AES counter model to produce a bit string s_i with length $|3^l| - 1$. The encryption key is k. The initial counter number is i.
  3. Set $(c'_x)_2 = (0||s_i) \oplus (c_x)_2$.
  4. If c'_x is a x-coordinate of a point on the EC, do
     (a) Choose the point $C' = (c'_x, b)$ in a compressed expression.
     (b) If $C' \in \mathbb{G} \setminus \{0\}$, it produces an output (C', i) and stops.
  5. Otherwise, $t = t + 1$ and if $t < MAX$, the counter i is selected at random from the interval $[0, 2^{32} - 1]$, and returns step 2. Else it returns an error and stops.

The variable "MAX" is related to the successful probability of the algorithm. There are two conditions for a successful run. One is that c'_x is a x-coordinate of a point on the EC. The other is that the point $C' \in \mathbb{G}$. The first condition happens with a probability $1/2$. The second condition happens with a probability q/n. So it is expected that $MAX \geq 2n/q$. As $i \in [0, 2^{32} - 1]$, we have at most $MAX < 2^{32}$. We suggest $MAX = 2^{16}$ as a balance of the runtime and successful probability.

Then for a signer, it sets $i \in [0, 2^{32} - 1]$ at random, and uses $f_k(C, i)$ as an encryption algorithm. For a verifier, it sets i as a received value, and uses $f_k(C, i)$ as an encryption algorithm. If a verifier tries to simulate an SDVS, it sets $i \in [0, 2^{32} - 1]$ at random, and uses $f_k(C, i)$ as a decryption algorithm.

Note that the value i should be a random value. If i is set to be fixed, say 0, and increased linearly, the non-transferability property may be lost. Note that the property has no limitations on the knowledge of an adversary. Suppose then an adversary has a verifier's private key. Given a (δ_1, δ_2, i), it works as follows.

- Compute $U = \delta_1^{x_V^{-1}}$.
- Compute $(W, i') = f_k(U, 0)$. If $i' = i$, a signature is produced by a signer. Otherwise, it is simulated by a verifier.

Note that if $(C', i) = f(C, 0)$, it is $(C, i) = f(C', i)$ and $(C'', j) = f(C', 0)$ where $C' \neq C''$ with a high probability.

Finally, it is expected that one could give a more efficient implementation to avoid the probabilistic algorithm and the extra communication cost.

Deterministic Identity Based Signature Scheme and Its Application for Aggregate Signatures

S. Sharmila Deva Selvi, S. Sree Vivek, and C. Pandu Rangan

Theoretical Computer Science Laboratory,
Department of Computer Science and Engineering,
Indian Institute of Technology Madras, Chennai, India
{sharmila,svivek}@cse.iitm.ac.in, prangan@iitm.ac.in

Abstract. Since the introduction of identity based cryptography in 1984 by Adi Shamir, several identity based signature schemes were reported. However, there are only two deterministic identity based signature schemes available in the literature and both of them use probabilistic private key generation and uses bilinear pairing. Moreover, these signatures consist of either two or more group elements and hence they are not 'short'. Thus an interesting and challenging open question is to design a deterministic signature scheme which does not use randomness either in the key generation phase or in the signing phase, avoid bilinear pairing and having a 'short' signature-where the signature consists of only one element. While this problem is addressed by BLS scheme in the PKI based setting, this has been an open problem in the identity based setting since 1984. This paper settles the open problem affirmatively. Specifically, we propose a fully deterministic identity based signature scheme, without using bilinear pairing. The signature consists of just one group element of a composite order group and its security is related to strong RSA problem in the random oracle model. Our security reduction is tight as one need not use forking lemma during security reduction for fully deterministic signature schemes. The major and important consequence of our scheme is its use for aggregate signature scheme. Our scheme leads to the first full aggregate identity based signature scheme with no prior communication among different signers. Besides our aggregate signature scheme does not employ any computation that goes through several rounds.

Keywords: Identity Based Deterministic Signature, Aggregate Signature, Full Aggregation, Random Oracle Model, Provable Security.

1 Introduction

The concept of Identity Based Cryptography (IBC) was introduced by Adi Shamir [21] in 1984. The distinguishing characteristic of identity based cryptography is the ability to use any string, that uniquely identifies a user in the system as the public key. In particular, this string may be the email address, telephone number, or combination of any of these parameter that is unique to that user. The corresponding private key can only be derived by a trusted Private Key Generator (PKG) who uses a master secret key, for deriving the private key of users.

The private key generation algorithm executed by the PKG may be viewed as a PKI based signature scheme. The signing algorithm executed by the user to generate the signed document is also a PKI based signature scheme. An identity based signature scheme is said to be probabilistic if the signing algorithm (run by the user) and/or the key generation algorithm (run by the PKG) are probabilistic. An identity based signature scheme is said to be partially deterministic if the signing algorithm is deterministic but key generation algorithm is probabilistic. The schemes reported in [8,2,19,11] are all probabilistic in nature. The schemes reported in [14,20] are the only known partially deterministic schemes in the literature. The scheme in [20] is the most efficient deterministic(partial) identity based signature in the literature till now.

An identity based signature scheme is said to be fully deterministic if both the key generation and signing algorithms are deterministic. There is no fully deterministic identity based signature scheme available in the literature. This paper presents the first of such a system. While the signatures of all other schemes contain two or more components, where at-least one element is a group element, the signature of our scheme contains just one component (a group element). Because it is a deterministic signature scheme, there is no need for using forking lemma and the reduction is tight, which results in compact key size that contributes for efficiency in practice. Even though computation of bilinear pairing has become efficient, finding out pairing friendly curves are difficult [10] and most of the efficient curves and means of compressing are patented. Thus, we have only a hand full of elliptic curves that support pairing for designing cryptosystem. Besides, since the RSA patent expired in the year 2000, designing cryptographic schemes based on RSA assumption gets more attention these days. Hence, the research in pairing free protocol is a very important and worthwhile effort.

The major application of our scheme is its implication in aggregation of signatures. In several real-life situations, it is advantageous to handle a collection of signed documents together rather than handling them in isolation. In applications such as e-Banking, legal document processing (archiving and communicating) in a legal firm, digital attestation related application and so on. In all the above applications, generating, storing and transmitting a large number of signed documents arise naturally. An *Aggregate Signature Scheme* combines several signed documents, say $\sigma_1, \ldots, \sigma_t$ on messages $m_1, \ldots, m_t$ by users $U_1, \ldots, U_t$ and produces a single signed document σ_{agg} where size of σ_{agg} is expected to be substantially smaller than sum of the sizes of σ_i's. Thus, the communication cost can be significantly reduced if we transmit σ_{agg} instead of transmitting $\sigma_1, \ldots, \sigma_t$ individually. A similar remark holds good even for storage requirements when we archive σ_{agg} (instead of $\sigma_1, \ldots, \sigma_t$). If $|\sigma_{agg}|$ depends on the number of signatures or number of messages (or both) it is called *partial aggregation* and if $|\sigma_{agg}|$ is independent of both the number of messages and signatures, then it is called *full aggregation*.

In a PKI based system, in order to verify an aggregate signature, the verifier needs the public keys of all the signers. In most applications, these public keys are transmitted along with the aggregate signature, defeating the primary purpose of aggregate signature schemes which is to reduce communication complexity. This is inevitable in order to support verification. Moreover, public keys come with a certificate, which is a signature from the certification authority (CA) and the CA's public key. Altogether, these sum up to huge bandwidth requirement than actually required to authenticate the messages. In an identity-based signature scheme, since the public key is just the users identity, e.g. e-mail address, IP address, Social Security Number (SSN) and Unique Identification Number (UID), individual public keys become no longer useful, and this removes the need for explicit certification and all associated costs. These facts make the identity-based paradigm more attractive or interesting for use in the design of aggregate signatures.

If the basic identity based signature scheme is probabilistic the aggregation will call for prior communication among signers to achieve full aggregation, where they have to propagate the local randomness used in generating their respective signatures. The computation will also proceed in several rounds. If the identity based signature is partly deterministic then the aggregation will lead to partial aggregation. However, if the basic signature scheme is fully deterministic, very efficient full aggregation is possible with no prior communication or rounds.

Related Work: Many well known PKI based aggregate signatures are available in the literature [7,16,3,17,18]. However, as the focus of this paper is on identity based aggregate signature scheme, we will not compare the PKI based schemes with ours. Most of the efficient aggregate signature schemes in the PKI setting are deterministic [7,3,17]. The first identity based aggregate signature scheme that achieves full aggregation was proposed in [9] by Cheng et al. Their scheme uses bilinear pairing and requires large setup cost because the signers essentially broadcast their individual random values to form a single random value. Moreover, the fact that the signature cannot be generated until all of the signers contribute in the first round, makes the scheme less practical. Gentry et al. proposed another scheme in [12], based on bilinear maps and the security of the scheme relies on the Gap Diffie Hellman problem. The weakness of the scheme is that, the signers of a given aggregate signature must agree on a common random value which was never used by any of the users before to generate a signature. The weaknesses in [12] is reported in [22]. Recently, Boldyreva et al. [5] proposed a sequential aggregate signature scheme (in-fact, the security of their original schemes [4] and [6] were flawed due to the assumption used to prove them was actually not hard to solve in polynomial time, as pointed out by Hwang et al. [15]). Their new scheme [5] was based on the hardness of a CDH-type problem that raised from their scheme and uses bilinear pairings. The first RSA based identity based aggregate signature scheme was proposed by Bagherzandi et al. [1]. This scheme uses two rounds of communication between the signers to generate a full aggregate signature, where the first round is to commit the random value shares (again by broadcasting the individual commitments as in [9]) and the second round is the aggregate signature generation round. Their scheme uses equivocable commitments and hence looses its generality and becomes less practical because of the overhead involved in broadcasting the commitments. The table **Table-1** summarizes the current state of the art in the research of identity based aggregate signatures which achieves full aggregation and are provably secure in the random oracle model. Aggregate signature scheme achieving partial aggregation without pairing is proposed by Sharmila et al. in [22].

However, aggregating the signatures generated by our scheme does not require any communication among users as there is no random value used in our signatures, and this settles the open problem posed in [15].

Table 1. State of the art survey of IBAS schemes

Schemes	Agg Sign Len	Sign Cost (/ user)	Agg Verify (t users)	Hard Prob	Sign Type (D/ND)	Rounds	Agg Mode (G/S)
[9]	$2\lvert\mathbb{G}\rvert$	3[M]	2[P]+t[M]	CDH	ND	2	G
[1]	$2\lvert\mathbb{Z}_n^*\rvert + \lvert\kappa\rvert + \lvert log(l)\rvert$	2[E]	t[E]	RSA	ND	2	G
[5]	$3\lvert\mathbb{G}\rvert$	7[M]	6[P]+t[M]	CDH-type	ND	-	S
[12]	$2\lvert\mathbb{G}\rvert+\lvert\mathbb{Z}_q^*\rvert$	5[M]	3[P]+t[M]	GAP-DH	ND	-	S
IBAS	$\lvert\mathbb{Z}_n^*\rvert$	2[E]	$(2t+1)$[E]	strong RSA	D	-	G

The terms used in **Table-1** are explained here. ND - Nondeterministic Signature, D- Deterministic Signature, G- General aggregation, S- Sequential Signature, κ is the security parameter of the scheme, [E]-Exponentiation in $\mathbb{Z}_n^*$, [M]- Scalar Point Multiplication in $\mathbb{G}$, [P]- Bilinear Pairing Operation, Gap-DH- Gap-Diffie-Hellman, [$\mathbb{G}_T$M]- Exponentiation in $\mathbb{G}_T$.

Computational Assumption: We review the computational assumption related to the protocols we discuss.

The Strong RSA Problem: Given a randomly chosen RSA modulus n and a random $c \in \mathbb{Z}_n^*$, finding $b > 1$ and $a \in \mathbb{Z}_n^*$, such that $c \equiv a^b \bmod n$ is the strong RSA problem.

The Strong RSA Assumption: The advantage of any probabilistic polynomial time algorithm $\mathcal{F}$ in solving the strong RSA problem in $\mathbb{Z}_n^*$ is defined as:

$$Adv_{\mathcal{F}}^{sRSA} = Pr\left[\mathcal{F}(n, c) \rightarrow \{a, b\} \mid (a \in \mathbb{Z}_n^*, b > 1) \wedge (c \equiv a^b \, mod \, n)\right]$$

The *strong RSA Assumption* is that, for any probabilistic polynomial time algorithm $\mathcal{F}$, the advantage $Adv_{\mathcal{F}}^{sRSA}$ is negligibly small.

2 Generic Model

In this section, we describe the generic frame work for a identity based signature scheme and an aggregate signature scheme. An identity based aggregate signature scheme (IBAS) consists of following six algorithms. The framework for a deterministic identity based signature scheme (D-IBS) consists of the first four algorithms described below, namely **Setup**, **Extract**, **Sign** and **Verify**. If the signature scheme is deterministic the signature for a message is always the same for every of invocation of the sign algorithm.

Setup: The private key generator (PKG) provides the security parameter κ as the input to this algorithm, generates the system parameters *params* and the master private key *msk*. PKG publishes *params* and keeps *msk* secret.

Extract: The user provides his identity ID to the PKG. The PKG runs this algorithm with identity ID, *params* and *msk* as the input and obtains the private key D. The private key D is sent to user through a secure channel.

Sign: For generating a signature on a message m, the user provides his identity ID, his private key D, *params* and the message m as input. This algorithm generates a valid signature σ on message m by the user.

Verify: This algorithm on input a signature σ on message m by the user with identity ID, checks whether σ is a valid signature on message m by ID. If true it outputs "*Valid*", else it outputs "*Invalid*".

AggregateSign: On receiving the various signatures $(\sigma_i)_{i=1\,to\,t}$ from different users $(U_i)_{i=1\,to\,t}$, any third party or one of the signers can run this algorithm and generate the aggregate signature σ_{agg} for the set of $\langle message, identity \rangle$ pairs $(m_i, ID_i)_{i=1\,to\,t}$.

Note: For sequential aggregation, each user contribute in the generation of aggregate signature by aggregating his own signature to the aggregate signature generated by the signers so far.

AggregateVerify: This algorithm on input of an aggregate signature σ_{agg}, the list for $(m_i, ID_i)_{i=1\,to\,t}$ and the *params* checks whether σ_{agg} is a valid aggregate signature on m_i by ID_i for all $i = 1$ to t. If true, it outputs "*Valid*", else outputs "*Invalid*".

3 Security Model

3.1 Existential Unforgeability of D-IBS

We define the security model for the existential unforgeability of a deterministic identity based signature scheme under adaptive chosen identity and message attack in this section. A D-IBS scheme is secure against existential forgery, under adaptive chosen identity and message attack if no probabilistic polynomial time forger $\mathcal{F}$ has non-negligible advantage in the following game:

Setup Phase: The challenger $\mathcal{C}$ runs the setup algorithm and generates the system parameters *params* and the master secret key *msk*. Now, $\mathcal{C}$ gives *params* to the forger $\mathcal{F}$ and keeps *msk* secret.
Training Phase: After the setup is done, $\mathcal{F}$ starts interacting with $\mathcal{C}$ by querying the various oracles provided by $\mathcal{C}$ in the following way:

- **Extract Oracle:** When $\mathcal{F}$ makes a query with an identity ID as input, $\mathcal{C}$ outputs D, the private key of ID to $\mathcal{F}$.

– **Signing Oracle:** When $\mathcal{F}$ makes a signing query with identity ID and message m, $\mathcal{C}$ outputs a valid signature σ on m by ID.

Forgery Phase: $\mathcal{F}$ outputs a signature σ, with ID_S as signer, and on a message m^*. $\mathcal{F}$ wins the game if σ is a valid signature and $\mathcal{F}$ has not queried for the signature corresponding to (ID_S, m^*) pair from the sign oracle and private key corresponding to ID_S. The advantage of $\mathcal{F}$ is given by,

$$Adv_{\mathcal{F}}^{D-IBS} = \{Pr[\mathcal{F}(Verify(\sigma) = valid)\}$$

3.2 Existential Unforgeability of IBAS

We define the security model for the existential unforgeability of an IBAS scheme under adaptive chosen identity and message attack in this section. An IBAS scheme is secure against existential forgery under adaptive chosen identity and message attack if no probabilistic polynomial time algorithm $\mathcal{F}$ has non-negligible advantage in the following game.

Setup Phase: The challenger $\mathcal{C}$ runs the setup algorithm and generates the system public parameters *params* and the master secret key *msk*. Now, $\mathcal{C}$ gives *params* to the forger $\mathcal{F}$ and keeps *msk* secret.

Training Phase: After the setup is done, $\mathcal{F}$ starts interacting with $\mathcal{C}$ by querying the oracles provided by $\mathcal{C}$ in the following way:

– **Extract Oracle:** When $\mathcal{F}$ makes a query with an identity ID as input, $\mathcal{C}$ outputs D, the private key of ID to $\mathcal{F}$.
– **Signing Oracle:** When $\mathcal{F}$ makes a signing query with identity ID and message m, $\mathcal{C}$ outputs a valid signature σ on m by ID.

  Note: It should be noted that Aggregate sign oracle is not required for the adversary because aggregation is a public process and any third party who has t signatures can combine all the signatures to form an aggregate signature. Thus, the forger $\mathcal{F}$ can always generate the aggregate signature after querying t individual signatures. However, in a sequential aggregation $\mathcal{F}$ sends a so far aggregated signature σ_{agg} along with a message, identity pair (m_i, ID_i) and requests for the aggregate signature. $\mathcal{C}$ generates the current aggregate signature σ_{agg} and sends it to $\mathcal{F}$.

Forgery Phase: $\mathcal{F}$ outputs an aggregate signature σ_{agg} for signatures $(\sigma_i)_{i=1\,to\,t}$ from the users $(ID_i)_{i=1\,to\,t}$ on messages $(m_i)_{i=1\,to\,t}$ where, at least one identity in the list of identities is the say $ID_S \in \{ID_i\}_{i=1\,to\,t}$, for which the private key was not queried by $\mathcal{F}$ and let m_S the message corresponding to ID_S. The forger $\mathcal{F}$ wins the game if σ_{agg} is a valid aggregate signature and $\mathcal{F}$ has not queried for the signature corresponding to (ID_S, m_S) pair from the sign oracle.

$$Adv_{\mathcal{F}}^{IBAS} = \{Pr[\mathcal{F}(Verify(\sigma_{agg}) = valid)\}$$

4 Deterministic Identity Based Signature Scheme (D-IBS)

In this section, we propose a new deterministic identity based signature scheme and also prove the scheme to be existentially unforgeable under adaptive chosen message and adaptive chosen identity attack in the random oracle model. The deterministic identity based signature scheme consists of the following algorithms:

- **Setup(κ):** Given κ as input, the PKG generates *params* and *msk* by performing the following:
  - Chooses two primes p and q of size κ, such that $p = 2p' + 1$ and $q = 2q' + 1$ where p' and q' are also primes.
  - Computes $n = pq$ and the Euler's totient function $\phi(n) = (p-1)(q-1)$. Therefore $|n| = 2\kappa$ and $|\phi(n)| = 2\kappa$.
  - Chooses e, such that $|e| = \kappa/4$ and computes d such that $ed \equiv 1 \bmod \phi(n)$.
  - It also chooses three cryptographic hash functions $H_0 : \{0,1\}^* \times \{0,1\} \rightarrow \mathbb{Z}_n^*$, $H_1 : \{0,1\}^{l_m} \times \{0,1\}^{l_1} \times \{0,1\} \rightarrow \{0,1\}^{\kappa/2}$ and $H_2 : \{0,1\}^{l_m} \times \{0,1\}^{l_1} \times \{0,1\} \rightarrow \{0,1\}^{\kappa/2}$. Here l_m is the size of message and l_1 is the size of identity of a user.
  - Now, PKG publicizes the system parameters, $params = \langle \kappa, n, e, H_0, H_1, H_2 \rangle$ and keeps the factors of n, namely p, q and the secret inverse d as the master secret key *msk*.
- **Extract(ID):** The user provides his identity ID to PKG. The PKG performs the following to find out the private key of the corresponding user:
  - Compute $g_0 = H_0(ID, 0)$ and $g_1 = H_0(ID, 1)$.
  - Compute $d_0 = (g_0)^d \ mod\ n$ and $d_1 = (g_1)^d \ mod\ n$.
  - The private key $D = \langle d_0, d_1 \rangle$ is sent to the corresponding user through a secure and authenticated channel.
- **Sign(m, ID, D):** To generate a deterministic signature on a message m, the user with identity ID performs the following:
  - Picks $\beta \in_R \{0,1\}$
  - Computes $h_1 = H_1(m, ID, \beta)$ and $h_2 = H_2(m, ID, \beta)$.
  - Computes $\sigma = (d_0)^{h_1}(d_1)^{h_2} \ mod\ n$.

  Now, $\langle \sigma, \beta \rangle$ is the signature on m by user with identity ID.

  It should be noted that β is random from others view but fixed with respect to the signer. As in [13], in order to avoid maintaining a record of all previous message/signature pairs, the signer can generate β as $\beta = PRF(D, ID, m)$, where $PRF()$ is a private random function (private to the signer). Thus, for a corresponding private key D, identity ID and message m, there is only one possibility of β.
- **Verify(m, σ, β, ID):** In order to verify the validity of a signature $\langle \sigma, \beta \rangle$ with respect to the identity ID and message m, the verifier performs the following:
  - Computes $g_0 = H_0(ID, 0)$ and $g_1 = H_0(ID, 1)$.
  - Computes $h_1' = H_1(m, ID, \beta)$ and $h_2' = H_2(m, ID, \beta)$.

- Checks whether $\sigma^e \ mod \ n \overset{?}{=} (g_0)^{h'_1}(g_1)^{h'_2} \ mod \ n$
- If the above check holds, outputs "*Valid*", otherwise outputs "*Invalid*".

Correctness of Verification

$$L.H.S = \sigma^e = ((d_0)^{h'_1}(d_1)^{h'_2})^e = ((g_0^d)^{h'_1}(g_1^d)^{h'_2})^e = (g_0)^{h'_1}(g_1)^{h'_2} = R.H.S$$

4.1 Existential Unforgeability of D-IBS

Theorem 1. *The identity based signature scheme (D-IBS) is EUF-D-IBS-CMA secure in the random oracle model under adaptive chosen message and adaptive chosen identity attack, if the strong RSA problem is assumed to be hard in $\mathbb{Z}_n^*$, where $n = pq$, and p, q, $(p-1)/2$ and $(q-1)/2$ are large prime numbers.*

Proof: Suppose a forger $\mathcal{F}$ is capable of breaking the EUF-D-IBS-CMA security of the D-IBS scheme and a challenger $\mathcal{C}$ is challenged with an instance of the strong RSA problem say $\langle n, c \in \mathbb{Z}_n^* \rangle$, where n is a composite number with two big prime factors p and q, such that $(p-1)/2$ and $(q-1)/2$ are also primes. $\mathcal{C}$ can make use of $\mathcal{F}$ to compute a and b such that $c \equiv a^b \bmod n$, by playing the following interactive game with $\mathcal{F}$. (Note that if *mod* operation is not specified then the computation is pure integer computation.)

Setup: $\mathcal{C}$ begins the game by setting up the system parameters as in the D-IBS scheme. $\mathcal{C}$ takes n from the instance of the strong RSA problem, chooses e such that $|e| = \kappa/4$, $e = xy$ for some arbitrary x and y, and $|x| = |y| = \kappa/8$. $\mathcal{C}$ chooses w such that $|w| = \kappa/8$. $\mathcal{C}$ sends the public parameters $params = \langle n, e \rangle$ to $\mathcal{F}$. $\mathcal{C}$ stores w, x and y for future use in $\mathcal{O}_{H_0}$ and $\mathcal{O}_{H_2}$ oracles. $\mathcal{C}$ also designs the three hash functions H_0, H_1 and H_2 as random oracles $\mathcal{O}_{H_0}$, $\mathcal{O}_{H_1}$ and $\mathcal{O}_{H_2}$. $\mathcal{C}$ maintains three lists L_{H_0}, L_{H_1} and L_{H_2} in order to consistently respond to the queries to the random oracles $\mathcal{O}_{H_0}$, $\mathcal{O}_{H_1}$ and $\mathcal{O}_{H_2}$ respectively.

Training Phase: $\mathcal{F}$ performs a series of queries to the oracles provided by $\mathcal{C}$. The descriptions of the oracles and the responses given by $\mathcal{C}$ to the corresponding oracle queries by $\mathcal{F}$ are described below. For the sake of simplicity, we assume that $\mathcal{O}_{H_0}(.)$ oracle is queried with ID and both 0 and 1 as inputs, before any other oracle is queried with the corresponding identity as the input parameters.

Oracle $\mathcal{O}_{H_0}(ID, l \in \{0,1\})$*:* We make a simplifying assumption that $\mathcal{A}$ queries the $\mathcal{O}_{H_0}$ oracle with distinct inputs in each query. If the oracle was queried with ID as input for $l = 0$ first, the next query with the same identity can be made with $l = 1$. There is no loss of generality due to this assumption, because, if the an identity is repeated with the same l value, by definition the oracle consults the list L_{H_0} and gives the same response. Thus, we assume that $\mathcal{A}$ asks $2q_{H_0}$ distinct queries for q_{H_0} distinct identities. Among these q_{H_0} identities, a random identity has to be selected as target identity and it is done as follows.

$\mathcal{C}$ selects a random index T, where $1 \leq T \leq 2q_{H_0}$. $\mathcal{C}$ does not reveal T to $\mathcal{A}$. When $\mathcal{A}$ generates the T^{th} query on ID_T, $\mathcal{C}$ decides to fix ID_T as target identity for the forgery phase. Moreover, $\mathcal{C}$ responds to $\mathcal{A}$ as follows:

- If a tuple of the form $\langle ID, l, d_l, g_l, *\rangle$ exists in list L_{H_0}, then it returns g_l as response.
- If the tuple of the form $\langle ID, l, d_l, g_l, *\rangle$ does not exist in list L_{H_0}, then it does the following.
  - If it is not the T^{th} query i.e. $i \neq T$, then $\mathcal{C}$ performs the following:
    - Chooses $d_0, d_1 \in_R \mathbb{Z}_n^*$ and sets $H_0(ID, j) = g_j = (d_j)^e \ mod \ n$ for j=0,1.
    - Stores the tuple $\langle ID_i, j, d_j, g_j, -\rangle$(for j=0,1) to list L_{H_0} and returns g_l as the response.
  - If it is the T^{th} query i.e. $i = T$, then $\mathcal{C}$ performs the following:
    - Chooses r_0 and r such that $|r_0| = \kappa/2$ and $|r| = \kappa/4$.
    - Sets $H_0(ID_T, 0) = g_0 = c^{xw} \ mod \ n$
    - Sets $H_0(ID_T, 1) = g_1 = c^{x+re} \ mod \ n$
      Note: c is taken from the strong RSA instance, x is chosen by $\mathcal{C}$ during setup.
    - Here d_0, d_1 are not known to $\mathcal{C}$ and hence $\mathcal{C}$ sets $d_j =$ " $-$ " for j=0,1.
    - Stores the tuples $\langle ID, j, -, g_j, -\rangle$ (for j=0,1) to list L_{H_0} and returns g_l as the response.

Note: For $\lambda \in \{0,1\}$, $\bar{\lambda}$ represents the negation of λ.

Oracle $\mathcal{O}_{H_1}(m, ID, \beta)$: When this query is made by $\mathcal{F}$, $\mathcal{C}$ does the following:

- If a tuple of the form $\langle m, ID, \lambda, \texttt{useful}, s_{\lambda 1}^{(m)}, t_{\lambda 1}^{(m)}, u^{(m)}, h_{1\lambda}^{(m)}\rangle$, where $\lambda = \beta$ exists in the list L_{H_1} then return $h_{1\lambda}^{(m)}$.
- If a tuple of the form $\langle m, ID, \bar{\lambda}, \texttt{useful}, s_{\bar{\lambda} 1}^{(m)}, t_{\bar{\lambda} 1}^{(m)}, v^{(m)}, h_{1\bar{\lambda}}^{(m)}\rangle$, where $\bar{\lambda} = \beta$ exists in the list L_{H_1} then return $h_{1\bar{\lambda}}^{(m)}$.
- Else, performs the following:
  - Chooses $\lambda \in_R \{0,1\}$.
  - For λ, perform the following:
    - Choose $s_{\lambda 1}^{(m)}, t_{\lambda 1}^{(m)}, s_{\lambda 2}^{(m)}, t_{\lambda 2}^{(m)}, u^{(m)} \in_R \{0,1\}^{\kappa/4}$.
    - Compute $h_{1\lambda}^{(m)} = u^{(m)} + s_{\lambda 1}^{(m)} e + t_{\lambda 1}^{(m)} y$.
    - Compute $h_{2\lambda}^{(m)} = -u^{(m)} w + s_{\lambda 2}^{(m)} e + t_{\lambda 2}^{(m)} y + 1$.
    - Set $\texttt{useful} = 1$.
    - Store the tuple $\langle m, ID, \lambda, \texttt{useful}, s_{\lambda 1}^{(m)}, t_{\lambda 1}^{(m)}, u^{(m)}, h_{1\lambda}^{(m)}\rangle$ in list L_{H_1}.
    - Store the tuple $\langle m, ID, \lambda, \texttt{useful}, s_{\lambda 2}^{(m)}, t_{\lambda 2}^{(m)}, u^{(m)}, h_{2\lambda}^{(m)}\rangle$ in list L_{H_2}.
  - For $\bar{\lambda}$, where $\bar{\lambda} = \neg\lambda$ perform the following:
    - Choose $s_{\bar{\lambda} 1}^{(m)}, t_{\bar{\lambda} 1}^{(m)}, s_{\bar{\lambda} 2}^{(m)}, t_{\bar{\lambda} 2}^{(m)}, v^{(m)} \in_R \{0,1\}^{\kappa/4}$.
    - Compute $h_{1\bar{\lambda}}^{(m)} = v^{(m)} + s_{\bar{\lambda} 1}^{(m)} e + t_{\bar{\lambda} 1}^{(m)} y$.
    - Compute $h_{2\bar{\lambda}}^{(m)} = -v^{(m)} w + s_{\bar{\lambda} 2}^{(m)} e + t_{\bar{\lambda} 2}^{(m)} y$.
    - Set $\texttt{useful} = 0$.
    - Store the tuple $\langle m, ID, \bar{\lambda}, \texttt{useful}, s_{\bar{\lambda} 1}^{(m)}, t_{\bar{\lambda} 1}^{(m)}, u^{(m)}, h_{1\bar{\lambda}}^{(m)}\rangle$ in list L_{H_1}.
    - Store the tuple $\langle m, ID, \bar{\lambda}, \texttt{useful}, s_{\bar{\lambda} 2}^{(m)}, t_{\bar{\lambda} 2}^{(m)}, u^{(m)}, h_{2\bar{\lambda}}^{(m)}\rangle$ in list L_{H_2}.
  - If $\beta = \lambda$, output $h_{1\lambda}$.
  - If $\beta = \bar{\lambda}$, output $h_{1\bar{\lambda}}$.

Oracle $\mathcal{O}_{H_2}(m, ID, \beta)$: When this query is made by $\mathcal{F}$, $\mathcal{C}$ does the following:

- If a tuple of the form $\langle m, ID, \lambda, \texttt{useful}, s^{(m)}_{\lambda 2}, t^{(m)}_{\lambda 2}, u^{(m)}, h^{(m)}_{2\lambda}\rangle$, where $\lambda = \beta$ exists in the list L_{H_2} then return $h^{(m)}_{2\lambda}$.
- If a tuple of the form $\langle m, ID, \bar{\lambda}, \texttt{useful}, s^{(m)}_{\bar{\lambda} 2}, t^{(m)}_{\bar{\lambda} 2}, v^{(m)}, h^{(m)}_{2\bar{\lambda}}\rangle$, where $\bar{\lambda} = \beta$ exists in the list L_{H_2} then return $h^{(m)}_{2\bar{\lambda}}$.
- Else, perform the following:
  - Chooses $\lambda \in_R \{0, 1\}$.
  - For λ, perform the following:
    - Choose $s^{(m)}_{\lambda 1}, t^{(m)}_{\lambda 1}, s^{(m)}_{\lambda 2}, t^{(m)}_{\lambda 2}, u^{(m)} \in_R \{0, 1\}^{\kappa/4}$.
    - Compute $h^{(m)}_{1\lambda} = u^{(m)} + s^{(m)}_{\lambda 1} e + t^{(m)}_{\lambda 1} y$.
    - Compute $h^{(m)}_{2\lambda} = -u^{(m)} w + s^{(m)}_{\lambda 2} e + t^{(m)}_{\lambda 2} y + 1$.
    - Set $\texttt{useful} = 1$.
    - Store the tuple $\langle m, ID, \lambda, \texttt{useful}, s^{(m)}_{\lambda 1}, t^{(m)}_{\lambda 1}, u^{(m)}, h^{(m)}_{1\lambda}\rangle$ in list L_{H_1}.
    - Store the tuple $\langle m, ID, \lambda, \texttt{useful}, s^{(m)}_{\lambda 2}, t^{(m)}_{\lambda 2}, u^{(m)}, h^{(m)}_{2\lambda}\rangle$ in list L_{H_2}.
  - For $\bar{\lambda}$, where $\bar{\lambda} = \neg\lambda$ perform the following:
    - Choose $s^{(m)}_{\bar{\lambda} 1}, t^{(m)}_{\bar{\lambda} 1}, s^{(m)}_{\bar{\lambda} 2}, t^{(m)}_{\bar{\lambda} 2}, v^{(m)} \in_R \{0, 1\}^{\kappa/4}$.
    - Compute $h^{(m)}_{1\bar{\lambda}} = v^{(m)} + s^{(m)}_{\bar{\lambda} 1} e + t^{(m)}_{\bar{\lambda} 1} y$.
    - Compute $h^{(m)}_{2\bar{\lambda}} = -v^{(m)} w + s^{(m)}_{\bar{\lambda} 2} e + t^{(m)}_{\bar{\lambda} 2} y$.
    - Set $\texttt{useful} = 0$.
    - Store the tuple $\langle m, ID, \bar{\lambda}, \texttt{useful}, s^{(m)}_{\bar{\lambda} 1}, t^{(m)}_{\bar{\lambda} 1}, u^{(m)}, h^{(m)}_{1\bar{\lambda}}\rangle$ in list L_{H_1}.
    - Store the tuple $\langle m, ID, \bar{\lambda}, \texttt{useful}, s^{(m)}_{\bar{\lambda} 2}, t^{(m)}_{\bar{\lambda} 2}, u^{(m)}, h^{(m)}_{2\bar{\lambda}}\rangle$ in list L_{H_2}.
  - If $\beta = \lambda$, output $h_{2\lambda}$.
  - If $\beta = \bar{\lambda}$, output $h_{2\bar{\lambda}}$.

Oracle $\mathcal{O}_{Extract}(ID)$: $\mathcal{C}$ checks whether tuples of the form $\langle ID, 0, d_0, g_0, -\rangle$ and $\langle ID, 1, d_1, g_1, r\rangle$ exists in the list L_{H_0}, if so returns the corresponding d_0 and d_1 as the private keys corresponding to the identity ID. However, if $ID = ID_T$, $\mathcal{C}$ aborts.

Oracle $\mathcal{O}_{Sign}(m, ID)$: $\mathcal{C}$ checks whether $ID \stackrel{?}{=} ID_T$ and performs the following to generate the signature on m by ID:

- If $ID \neq ID_T$, then $\mathcal{C}$ knows the private key corresponding to ID, so $\mathcal{C}$ chooses $\beta \in_R \{0, 1\}$ and performs the signing as per the protocol and generates σ, after querying $\mathcal{O}_{H_1}(m, ID, \beta)$ and $\mathcal{O}_{H_2}(m, ID, \beta)$.
- If $ID = ID_T$, then $\mathcal{C}$ checks whether a tuple corresponding to $(m, ID_T, -)$ is found in L_{H_1} and H_{H_2}. If it does not exist, $\mathcal{C}$ invokes the $\mathcal{O}_{H_1}(m, ID_T, 0)$ and $\mathcal{O}_{H_2}(m, ID_T, 0)$ oracles. Then, $\mathcal{C}$ simulates the signing algorithm as follows: (because, $\mathcal{C}$ does not know the private key corresponding to ID_T):
  - $\mathcal{C}$ retrieves the tuples corresponding to m, ID_T from the lists L_{H_1} and L_{H_2}, for which the flag $\texttt{useful}$=0. Let $\langle m, ID, \gamma, \texttt{useful} = 0, s^{(m)}_{\gamma 1}, t^{(m)}_{\gamma 1}, v^{(m)}, h^{(m)}_{1\gamma}\rangle$ be the tuple in list L_{H_1} and $\langle m, ID, \gamma, \texttt{useful} = 0, s^{(m)}_{\gamma 2}, t^{(m)}_{\gamma 2}, v^{(m)}, h^{(m)}_{2\gamma}\rangle$ be the tuple in list L_{H_2}.

  - Set $\beta = \gamma$.
  - Computes $\sigma = c^{xws_{\gamma 1}^{(m)}} c^{wt_{\gamma 1}^{(m)}} c^{xs_{\gamma 2}^{(m)}} c^{t_{\gamma 2}^{(m)}} c^{-rv^{(m)}w} c^{rs_{\gamma 2}^{(m)}e} c^{rt_{\gamma 2}^{(m)}y} \ mod\ n$.
- Sends σ, β as the signature on the message m by identity ID_T.

The verification of a signature is done by checking whether $\sigma^e \overset{?}{=} (g_0)^{h_1}(g_1)^{h_2}$. We need to verify only the case when $ID = ID_T$ because in other cases, the signature is generated as per the protocol.

Forgery Phase: At the end of the ***Training Phase***, $\mathcal{F}$ produces a forged signature σ^*, β^* on the message m^* as if signed by the user with identity ID_S. If σ^* is a valid signature on m^* and if σ^* satisfies all the constraints given below, then $\mathcal{C}$ can solve the hard problem.

- $ID_S = ID_T$
- Private key of ID_T is not queried to Extract oracle.
- Signature on m^*, ID_T is not queried to Sign Oracle. (This is forbidden in the model because it is a deterministic signature scheme).
- The tuple $\langle m^*, ID_T, \beta^*, \texttt{useful}, s_{\beta^*1}^{(m^*)}, t_{\beta^*1}^{(m^*)}, v^{(m^*)}, h_{1\beta^*}^{(m^*)} \rangle$ corresponding to m^*, ID_T, in list L_{H_1} and the tuple $\langle m^*, ID_T, \beta^*, \texttt{useful}, s_{\beta^*2}^{(m^*)}, t_{\beta^*2}^{(m^*)}, v^{(m^*)}$, $h_{2\beta^*}^{(m^*)} \rangle$ corresponding to m^*, ID_T, in list L_{H_2} has the flag "$\texttt{useful}$=1".

Now, if the above constraints are satisfied, then $\mathcal{C}$ obtains a and b such that $c = a^b \ mod\ n$ by performing the following:

- The public keys corresponding to ID_T, i.e. $H_0(ID_T, 0)$ and $H_0(ID_T, 1)$, are set to be $\langle g_0 = c^{xw}$ and $g_1 = c^{x+re} \rangle$ by $\mathcal{C}$ while $\mathcal{F}$ performed the $\mathcal{O}_{H_0}(ID_T, .)$ queries (Note that c was taken from the strong RSA problem instance).
- Let d be such that $d \equiv e^{-1} \bmod \phi(n)$, where e is the master public key.

  Note: Now, in terms of the public keys and the value d, the private keys corresponding to ID_T are $d_0 = g_0^d = c^{xwd}$ and $d_1 = g_1^d = c^{xd+r}$, **implicitly**. However, these values cannot be computed explicitly by $\mathcal{C}$, because $\mathcal{C}$ has no way of computing d. That is why $\mathcal{C}$ has set " $-$ " for these unknowns in $\mathcal{O}_{H_0}$ oracle queries. Thus, the values d_0 and d_1 used in the proof are only hypothetical.
- Since we have the condition that "$\texttt{useful} = 1$", $\mathcal{C}$ should have set the $h_{1\beta^*}^{(m^*)} = \mathcal{O}_{H_2}(m^*, ID_T, \beta^*) = u^{(m^*)} + s_{\beta^*1}^{(m^*)}e + t_{\beta^*1}^{(m^*)}y$ and $h_{2\beta^*}^{(m^*)} = \mathcal{O}_{H_2}(m^*, ID_T, \beta^*)$ $= u^{(m^*)}w + s_{\beta^*2}^{(m^*)}e + t_{\beta^*2}^{(m^*)}y + 1$ respectively.
- Now, $\sigma^* = (d_0)^{h_{1\beta^*}^{(m^*)}} (d_1)^{h_{2\beta^*}^{(m^*)}} = (c^{xwd})^{h_{1\beta^*}^{(m^*)}} (c^{xd+r})^{h_{2\beta^*}^{(m^*)}}$ and hence the signature verification holds good for a proper forgery, because $(\sigma^*)^e = ((c^{xwd})^{h_{1\beta^*}^{(m^*)}}$ $(c^{xd+r})^{h_{2\beta^*}^{(m^*)}})^e = (c^{xw})^{h_{1\beta^*}^{(m^*)}} (c^{x+re})^{h_{2\beta^*}^{(m^*)}} = g_0^{h_{1\beta^*}^{(m^*)}} g_1^{h_{2\beta^*}^{(m^*)}}$.
- Hence, the forgery σ^*, submitted by $\mathcal{F}$ is of the following form. Now, for the sake of simplicity, we rename the values $v^{(m^*)} = v$, $s_{\beta^*1}^{(m^*)} = s_1$, $t_{\beta^*1}^{(m^*)} = t_1$, $s_{\beta^*2}^{(m^*)} = s_2$ and $t_{\beta^*1}^{(m^*)} = t_2$ in the following equations.

$$
\begin{aligned}
\sigma^* &= (d_0)^{h_{1\beta^*}^{(m^*)}}(d_1)^{h_{2\beta^*}^{(m^*)}} = (c^{xwd})^{h_{1\beta^*}^{(m^*)}}(c^{xd+r})^{h_{2\beta^*}^{(m^*)}} \\
&= (c^{xwd})^{(v^{(m^*)}+s_{\beta^*1}^{(m^*)}e+t_{\beta^*1}^{(m^*)}y)}(c^{xd+r})^{(-v^{(m^*)}w+s_{\beta^*2}^{(m^*)}e+t_{\beta^*2}^{(m^*)}y+1)} \\
&= (c^{xwd})^{(v+s_1e+t_1y)}(c^{xd})^{(-vw+s_2e+t_2y+1)}(c^r)^{(-vw+s_2e+t_2y+1)} \\
&= c^{(xwdv+xwds_1e+xwdt_1y)}c^{(-xdvw+xds_2e+xdt_2y+xd)}c^{(-rvw+rs_2e+rt_2y+r)} \\
&= c^{(xwdv+xwds_1e+wdt_1e)}c^{(-xwdv+xds_2e+dt_2e+xd)}c^{(-rvw+rs_2e+rt_2y+r)} \\
&\qquad\qquad (\text{Since } xy = e) \\
&= c^{(xwds_1e+wdt_1e+xds_2e+dt_2e-rvw+rs_2e+rt_2y+r)}c^{xd} \\
&= c^{(xws_1+wt_1+xs_2+t_2-rvw+rs_2e+rt_2y+r)}c^{xd} \ (\text{Since } ed \equiv 1 \ mod \ \phi(n))
\end{aligned}
$$

– Let, $z = c^{(xws_1+wt_1+xs_2+t_2-rvw+rs_2e+rt_2y+r)}$ and z can be computed by $\mathcal{C}$ because $\mathcal{C}$ knows the values $r, s_1, s_2, t_1, t_2, v, w, x$ and y.
– Therefore, $\sigma^* = zc^{xd} \Rightarrow c^{xd} = \sigma^*/z \Rightarrow c^{xd} = c^{x(x^{-1}y^{-1})} = c^{y^{-1}} \Rightarrow \sigma^* = zc^{y^{-1}}$. Thus, $\mathcal{C}$ can obtain the solution for the equation $c = a^b \ mod \ n$ with $a = \sigma^*/z$ and $b = y$.

Probability Analysis: *The analysis is given in the full version of this paper*

5 Identity Based Aggregate Signature Scheme from RSA (IBAS)

We propose the new identity based aggregate signature (IBAS) scheme in this section and prove the existential unforgeability of the scheme.

Deterministic General IBAS: Our scheme is a deterministic identity based aggregate signature scheme, which supports full aggregation, i.e. the size of the aggregate signature is one group element along with the message and the list of identities. The scheme consists of six algorithms, out of which **Setup**, **Extract**, **Sign** and **Verify** are identical to that of D-IBS scheme. We explain the **AggregateSign** and **AggregateVerify** algorithms below:

– **AggregateSign:** This algorithm takes as input a set of t signatures $\{\sigma_i, \beta_i\}_{i=1\,to\,t}$ and the corresponding message identity pairs $\langle m_i, ID_i\rangle$, such that $\forall i = 1$ to t, $\langle\sigma_i, \beta_i\rangle$ is the valid signature by the user with identity ID_i on message m_i. The aggregation is done as follows:

$$\sigma_{agg} = \prod_{i=1}^{t} \sigma_i$$

The identity based aggregate signature is σ_{agg} and the corresponding list of message, identity and β's is $\mathcal{L} = \{m_i, ID_i, \beta_i\}_{i=1\,to\,t}$.

– **AggregateVerify:** This algorithm takes the identity based aggregate signature σ_{agg} and the corresponding list of message identity pairs, $\mathcal{L} = \{m_i, ID_i, \beta_i\}_{i=1\,to\,t}$ and performs the verification as follows:
  • For all i=1 to t
    Compute $g_{i0} = H_0(ID_i, 0)$
    Compute $g_{i1} = H_0(ID_i, 1)$
    Compute $h'_{i1} = H_1(m_i, ID_i, \beta_i)$ and
    Compute $h'_{i2} = H_2(m_i, ID_i, \beta_i)$

- If $\sigma^e_{agg} \overset{?}{=} \prod_{i=1}^{t} ((g_{i0})^{h'_{i1}} (g_{i1})^{h'_{i2}})$, then outputs "$Valid$" else outputs "$Invalid$".

The correctness of verification is straight forward.

Security Proof for IBAS

Theorem 2. *Our identity based aggregate signature scheme (IBAS) is EUF-IBAS-CMA secure in the random oracle model under adaptive chosen message and adaptive chosen identity attack, if the strong RSA problem is assumed to be hard in $\mathbb{Z}^*_n$, where p, q, $(p-1)/2$ and $(q-1)/2$ are large prime numbers.*

Proof is given in the full version of this paper.

6 Conclusion

This paper presents the first fully deterministic identity based signature scheme whose signature consists of just one group element. This scheme leads to an identity based aggregate signature scheme that is most efficient and achieves full aggregation without any prior communication among signers and this settles the open problem raised in [15]. Our deterministic signature scheme has additional attractive property that it does not employ any bilinear pairing based computations. We have proved the security in random oracle model and the natural open question is to design a fully deterministic identity based signature scheme, secure in the standard model.

References

1. Bagherzandi, A., Jarecki, S.: Identity-Based Aggregate and Multi-Signature Schemes Based on RSA. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 480–498. Springer, Heidelberg (2010)
2. Barreto, P.S.L.M., Libert, B., McCullagh, N., Quisquater, J.-J.: Efficient and Provably-Secure Identity-Based Signatures and Signcryption from Bilinear Maps. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 515–532. Springer, Heidelberg (2005)
3. Bellare, M., Namprempre, C., Neven, G.: Unrestricted Aggregate Signatures. In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 411–422. Springer, Heidelberg (2007)
4. Boldyreva, A., Gentry, C., O'Neill, A., Yum, D.H.: Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing. In: ACM Conference on Computer and Communications Security, CCS 2007, pp. 276–285. ACM (2007)
5. Boldyreva, A., Gentry, C., O'Neill, A., Yum, D.H.: Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing. Cryptology ePrint Archive, Report 2007/438 (2007), http://eprint.iacr.org/ (revised on February 21, 2010)
6. Boldyreva, A., Gentry, C., O'Neill, A., Yum, D.H.: New multiparty signature schemes for network routing applications. ACM Transactions on Information and System Security (TISSEC) 12(1), 1–39 (2008)

7. Boneh, D., Gentry, C., Lynn, B., Shacham, H.: Aggregate and Verifiably Encrypted Signatures from Bilinear Maps. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 416–432. Springer, Heidelberg (2003)
8. Cha, J.C., Cheon, J.H.: An Identity-Based Signature from Gap Diffie-Hellman Groups. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 18–30. Springer, Heidelberg (2002)
9. Cheng, X., Liu, J., Wang, X.: Identity-Based Aggregate and Verifiably Encrypted Signatures from Bilinear Pairing. In: Gervasi, O., Gavrilova, M.L., Kumar, V., Laganá, A., Lee, H.P., Mun, Y., Taniar, D., Tan, C.J.K. (eds.) ICCSA 2005, Part IV. LNCS, vol. 3483, pp. 1046–1054. Springer, Heidelberg (2005)
10. Freeman, D., Scott, M., Teske, E.: A taxonomy of pairing-friendly elliptic curves. Journal of Cryptology 23(2), 224–280 (2010)
11. Galindo, D., Garcia, F.D.: A Schnorr-Like Lightweight Identity-Based Signature Scheme. In: Preneel, B. (ed.) AFRICACRYPT 2009. LNCS, vol. 5580, pp. 135–148. Springer, Heidelberg (2009)
12. Gentry, C., Ramzan, Z.: Identity-Based Aggregate Signatures. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 257–273. Springer, Heidelberg (2006)
13. Goh, E.-J., Jarecki, S., Katz, J., Wang, N.: Efficient signature schemes with tight reductions to the diffie-hellman problems. Journal of Cryptology 20(4), 493–514 (2007)
14. Herranz, J.: Deterministic identity-based signatures for partial aggregation. The Computer Journal 49(3), 322–330 (2006)
15. Hwang, J.Y., Lee, D.H., Yung, M.: Universal forgery of the identity-based sequential aggregate signature scheme. In: Computer and Communications Security, ASIACCS 2009, pp. 157–160. ACM (2009)
16. Lu, S., Ostrovsky, R., Sahai, A., Shacham, H., Waters, B.: Sequential Aggregate Signatures and Multisignatures Without Random Oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 465–485. Springer, Heidelberg (2006)
17. Ma, D.: Practical forward secure sequential aggregate signatures. In: Proceedings of the 2008 ACM Symposium on Information, Computer and Communications Security, ASIACCS 2008, pp. 341–352. ACM (2008)
18. Neven, G.: Efficient Sequential Aggregate Signed Data. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 52–69. Springer, Heidelberg (2008)
19. Sakai, R., Ohgishi, K., Kasahara, M.: Cryptosystems based on pairing. In: The 2000 Symposium on Cryptography and Information Security, Okinawa, Japan, pp. 135–148 (January 2000)
20. Sharmila Deva Selvi, S., Sree Vivek, S., Pandu Rangan, C.: Identity-Based Deterministic Signature Scheme without Forking-Lemma. In: Iwata, T., Nishigaki, M. (eds.) IWSEC 2011. LNCS, vol. 7038, pp. 79–95. Springer, Heidelberg (2011)
21. Shamir, A.: Identity-Based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
22. Sree Vivek, S., Sharmila Deva Selvi, S., Shriram, J., Pandu Rangan, C.: Identity based partial aggregate signature scheme without pairing. Accepted in 35th IEEE Sarnoff Symposium (2012) full version, http://eprint.iacr.org/2010/461

Fully Leakage-Resilient Signatures with Auxiliary Inputs

Tsz Hon Yuen, Siu Ming Yiu, and Lucas C.K. Hui

Department of Computer Science, The University of Hong Kong, Hong Kong
{thyuen,smyiu,hui}@cs.hku.hk

Abstract. The auxiliary input model for leakage-resilient encryption considers the leakage of a computationally hard-to-invert function, which can capture a wide class of possible side channel attacks. To avoid the trivial attack that the leakage function simply outputs the forged signature, we propose a new *selective auxiliary input* model for signatures. This model captures side channel attacks that are based on the physical implementation of the cryptosystem regardless of the underlying public parameters chosen.

We provide the first generic construction of fully leakage-resilient signatures, allowing polynomial leakage of the signing key and all intermediate randomness used, under this *selective auxiliary input* model. We then demonstrate an efficient instantiation of it, thus solving an open problem mentioned by Boyle *et al.* (Eurocrypt 2011).

1 Introduction

To guarantee the security of a cryptosystem, we usually define an attack model which captures and limits the adversarial behavior of an attacker. The system is only guaranteed to be secure under the defined model if the adversary follows the restrictions described in the model. One of the most fundamental restriction of most traditional models is that the adversary should not be able to obtain the secret key of the system[1]. In most cases, such as the IND-CCA model for encryption or the EUF-CMA model for signatures, the adversary is not even allowed to obtain one bit of information of the secret key. However, in reality, there are a number of side-channel attacks (e.g. timing, power, etc.) which help the attacker to obtain partial information about the secret key. Therefore we do not have the security guarantee provided by the traditional security model, if the attacker obtains some side-channel information about the secret key.

Leakage-resilient cryptography was proposed to capture such side-channel attacks, and provides security guarantees even if the adversary can obtain partial information about the secret key. In this paper, we will focus on leakage-resilient signature schemes. We consider memory attacks, i.e., the adversary can learn arbitrary information about the secret state of a system, including the signing

[1] There are certain exceptions, such as the unconditional anonymity of ring signatures, where the signer remains anonymous even if the adversary obtains all signing keys.

key sk and the internal randomness r. More precisely, the power of an adversary is modeled by an efficiently computable function f applied to sk and r, under certain restrictions. The power of the attacker depends on the restrictions. The *relative leakage* model [1] only allows the leakage function f to output at most l bits, where l is smaller than the size of the secret key. The bit size restriction is later relaxed to the entropy constraint, where f is restricted to lower the entropy of the secret key by at most l bits [2]. These two models consider l as a fraction of the secret key. On the other hand, the *bounded retrieval model* (e.g., see [3,4]) considers l as a system parameter and increases the size of the secret key to accommodate l bits of leakage. It will not affect the size of the public key, the signature size and the efficiency of verification in the case of signatures.

A drawback of the length/entropy-bounded leakage model is that, if a system is used for a sufficiently long time, the real world attacker may eventually obtain leakages larger than the upper bound of the leakage permitted in the security model. If the secret key is shared among various cryptosystems, or the randomness is reused in different cryptosystems, the attacker may collect enough information through different channels for a sufficiently long time. There are two directions to solve this problem. The first approach is to refresh the secret key periodically, while bounding the leakage between updates. This model is known as the *continual leakage* model [5,6]. The second approach is to further relax the restriction on f by the *auxiliary input* model proposed by Dodis *et al.* [7]. This model allows the leakage of any function f which cannot be inverted by any polynomial time adversary with non-negligible probability. For example, the adversary can obtain a one-way permutation of the secret key in the auxiliary input model, which is not allowed in the length/entropy-bounded leakage model since it information-theoretically reveals the entire secret key. Therefore the auxiliary input model allows a larger classes of leakage functions.

Leakage-Resilient Signatures. Alwen *et al.* [8] gives a leakage-resilient signature (LR-Sig) in the bounded-retrieval model, which tolerates leakage of up to half the secret key. Katz and Vaikuntanathan [9] proposed a generic construction of LR-Sig in the standard model. It tolerates leakage of $n - n^{\epsilon}$ bits of information about the n-bit secret key for any $\epsilon > 0$, but does not allow leakage of the internal state. They also proposed an efficient leakage-resilient one-time signature scheme in the standard model based on one-way functions, which tolerates leakage of the secret key and the internal state. The notion of *fully leakage-resilient* was proposed in [9] to represent the resilient to leakage on all intermediate values the signer used throughout the lifetime of the system.

Continuous LR-Sig were proposed independently by Dodis *et al.* [5] and Brakerski *et al.* [6]. The first scheme in [5] is an extension from [9], and is resilient to leakage of the signing key only. The second scheme in [5] is secure in the random oracle model, which tolerates leakage of up to $n/2$ bits of the entire current secret state of the signer. Brakerski *et al.* [6] also proposed two LR-Sig in the continuous (bounded) leakage model. The first scheme is also an extension from [9], and is resilient to leakage of the signing key only. The second scheme is fully leakage-resilient in the random oracle model and tolerates $(1 - o(1))n$ bits leakage, but requires refreshing the signing key after every few invocation of the signing algorithm.

Fully leakage resilient signatures were proposed independently by Malkin *et al.* [10] and Boyle *et al.* [11], in the continuous (bounded) leakage model, which tolerates $(1 - o(1))n$ bits leakage of the entire secret state. Boyle *et al.* [11] mentioned that fully leakage-resilient signatures with auxiliary input is an open problem. In this paper, we try to solve this open problem, and also extend the auxiliary input model to support continuous updates.

A recent and independent paper by Faust *et al.* [12] also tried to solve this open problem by proposing a LR-Sig scheme with auxiliary input. They proposed two different security models and gave a secure construction in each model. They extended their schemes to give auxiliary input secure identification schemes.

Challenge on the Model. There are two main challenges for LR-Sig with auxiliary input: the security model and the construction. Modeling hard-to-invert leakage for signature schemes is an important open problem in [11]. In the bounded-leakage model, it requires that the signing key still has a certain amount of min-entropy even after seeing the leakage (of the signing key and the randomness used in past signatures). Firstly, observe that the EUF-CMA model for signatures cannot be trivially combined with the auxiliary input model for public key encryption. It is because in the auxiliary input model for encryption, the adversary is allowed to obtain an arbitrary polynomial-time computable hard-to-invert leakage f. It does not restrict the adversary to model the function f to be *the signing algorithm for the challenge message* M^*. If such leak oracle query is allowed for LR-Sig, the adversary can always forge a signature. Therefore, a new security model is needed for LR-Sig with auxiliary input.

Recently, Faust *et al.* [12] proposed two methods to solve this problem. Firstly, they proposed a *random message unforgeability* model, where the challenge message is randomly chosen from the message space instead of chosen by the adversary. Therefore, the adversary cannot ask for the leakage of a signature on the challenge message. Secondly, they proposed an auxiliary input model which only allows exponentially hard-to-invert leakage. Since the signing algorithm can be viewed as a polynomially hard-to-invert leakage, the signing algorithm is excluded from the set of allowed leakage in this model. Their first model is less useful in signatures, since the adversary may be able to sign a specific message even though he may not be able to sign a random message. Their second model is more suitable for signatures, but the set of allowed leakage functions is greatly reduced by the exponentially hard-to-invert restriction. It is desirable to define a security model for the standard chosen message attack with polynomially hard-to-invert leakage, and yet prevents the adversary to obtain a forged signature on the challenge message from the leakage.

Our Contributions. We give the *first* chosen message attack model that captures the leakage of the polynomially hard-to-invert function, called the *selective auxiliary input*. In this model, the adversary has to specify the leakage function f *prior to* the announcement of the public parameters (the public key, generator of the group used, modulus, etc.). As a result, the adversary cannot embed the signing algorithm into f without having the public parameters. It is similar to the selective ID model for identity-based encryption [13], where the adversary must commit ahead of time to the identity that it intends to attack. The advantage of the selective auxiliary input model is that it captures the standard chosen message attack, while we still capture the polynomially hard-to-invert leakage. This model is more suitable to capture real-world side channel attacks where the leakage method is independent of the public parameters. For example, for the power analysis of the CPU, the attacker may obtain certain bits after a few clock cycles of the CPU, which is not related to the public parameters of the underlying cryptosystem used. In general, side channel attacks are attacks that are based on the physical implementation of a cryptosystem, rather than the weakness of the public parameters that have been chosen. Therefore, the selective auxiliary input is a reasonable security model without having the problems of the security models in [12].

Challenge on the Construction. We would like to construct the *first* fully leakage-resilient signatures with selective auxiliary inputs. Under the new security model, the adversary can even leak the entire entropy of the secret key (e.g. one-way permutation of the secret key), which is not allowed in most LR-Sig schemes ([8,9,5,6,11]).

The first idea we considered is to use the recent leakage-resilient identity-based encryption (IBE) with auxiliary input [14]. It is well-known that a secure IBE implies a secure signature scheme [15]. However, this idea does not work since the model of leakage-resilient IBE only allows the leakage of identity-based secret keys, but not the randomness computing them. Therefore, it cannot be applied to the fully leakage-resilient signature model.

The second approach is to construct it from the Boyle *et al.* scheme [11]. The main challenge is that [11] used the entropy bound for the security proof. For the auxiliary input model, we only restrict that the leakage oracle cannot reveal the secret key. Therefore restricting the entropy loss cannot be used to construct a LR-Sig with auxiliary input. Furthermore, using an encryption scheme in a signature is not efficient. Generally speaking, a CPA-secure encryption scheme in the standard model can have a ciphertext size as double the message size. On the other hand, an efficient signature scheme in the standard model can have a signature size close to the message size. Therefore, it is desirable to design a LR-Sig scheme without using encryption.

Our Approach. We give a generic construction secure in the auxiliary input model by improving the schemes in [9,11]. The original purpose of the use of lossy encryption in [11] is to enable the simulator in the security proof to switch between the real witness (x, ω) and the fake witness (x', ω'). The distributions of the lossy encryption of both witnesses are statistically close. Therefore the simulator can leak the fake witness instead of the real one. The idea to construct a LR-Sig with auxiliary input is to replace the lossy encryption with a statistically hiding commitment scheme. The first observation is that we only need to hide the witness statistically. The decryption algorithm is only needed in the proof in [11]. Therefore, we only need a knowledge extractor for the commitment scheme with respect to the challenge message only. We use the statistically hiding commitment scheme with extraction since it can incorporate with the SNIWI proof system. The signature of a message m is the SNIWI proof for the following language L':

$$L' = \{(s, y, ck, m, H, C'') : \exists x, \omega'' \text{ s.t. } f_s(x) = y \wedge C'' = Commit_{ck}^{H(m)}(x; \omega'')\},$$

where $(KGen, Commit, Decommit)$ is a statistically hiding commitment scheme with extraction, $ck \leftarrow KGen(1^\lambda)$ and ω'' is the randomness used in $Commit_{ck}$ with the tag $H(m)$. There are a few advantages of using commitment scheme over encryption. Firstly, using encryption in a signature is somewhat counter-intuitive and seems inefficient as we discuss previously. Using commitment scheme in signature is common in many signature schemes. Our proposal is more efficient than the existing LR-Sig scheme, since the commitment scheme can be reused in the SNIWI proof. Secondly, when we use a public key encryption scheme, the corresponding decryption key is generated with the public key (unless the public key can be sampled uniformly from the public key space, without affecting the security). Although this decryption key is not stored by the signer, the key generation process may still be exposed by the side channel attack. Therefore, this decryption key is an extra source of weakness of the system and may reduce the security of the system. On the other hand, the binding key of the commitment scheme only appears in the security proof, since it is generated by another indistinguishable key generation algorithm appeared in the security proof. There is no real-world attack on the binding key.

2 Security Models for Leakage-Resilient Signatures

In this section, we define the security model for LR-Sig with auxiliary inputs.

A signature scheme consists of three polynomial-time algorithms:

`KeyGen`: On input a security parameter 1^λ, it generates a signing key sk and a verification key vk.

`Sign`: On input sk and a message m from a message space $\mathcal{M}$, it outputs a signature σ.

`Verify`: On input vk, m and σ, it outputs a bit 1 or 0 symbolizing the validity of the signature.

Correctness. $\forall m \in \mathcal{M}$, $1 \leftarrow \mathtt{Verify}(\mathsf{vk}, m, \mathtt{Sign}(\mathsf{sk}, m))$, where $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathtt{KeyGen}(1^\lambda)$.

2.1 Selective Auxiliary Input Model for Unforgeability

We consider the following existential unforgeability game against adaptive chosen message attacks (EUF-CMA) for unforgeability, together with the leakage-resilient with selective auxiliary inputs.

We write $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathtt{KeyGen}(1^\lambda; r)$, where r is the randomness used in `KeyGen`. We write $\sigma \leftarrow \mathtt{Sign}(\mathsf{sk}, m_i; r_i)$, where r_i is the randomness used in `Sign`. Denote

$\mathfrak{state}$ as a set of random coins used. Denote a polynomial-time (in λ) computable function family $\mathcal{F}$. The game $\mathsf{LR\text{-}Sig}$ is defined as follows.

1. Select. The adversary $\mathcal{A}$ gives a set of leakage functions $\mathbb{F} \subset \mathcal{F}$ to the challenger.
2. Setup. The challenger samples $r \leftarrow \{0,1\}^*$ and runs $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathtt{KeyGen}(1^\lambda; r)$. The challenger gives vk to $\mathcal{A}$ and sets $\mathfrak{state} = \{r\}$.
3. Query. The following oracles can be queried by $\mathcal{A}$:
   – Signing Oracle $\mathcal{SO}(m_i)$: On input a message $m_i \in \mathcal{M}$, it samples $r_i \leftarrow \{0,1\}^*$ and returns the signature $\sigma_i \leftarrow \mathtt{Sign}(\mathsf{sk}, m_i; r_i)$. It sets $\mathfrak{state} := \mathfrak{state} \cup \{r_i\}$.
   – Leak Oracle $\mathcal{LO}(f_i)$: On input a polynomial-time computable function $f_i \in \mathbb{F}$, it returns $f_i(\mathfrak{state})$.
4. Output. $\mathcal{A}$ returns a message-signature pair (m^*, σ^*).

$\mathcal{A}$ wins the game if $1 \leftarrow \mathtt{Verify}(\mathsf{vk}, m^*, \sigma^*)$ and m^* was not queried to the signing oracle.

A signature scheme is EUF-CMA secure w.r.t. selective auxiliary inputs from $\mathcal{F}$ if there is no PPT $\mathcal{A}$ winning the game above with non-negligible probability.

2.2 Classes of Auxiliary Input Functions

We define two classes of function families as Dodis *et al.* [7]. For future convenience, we will parametrize these families by the min-entropy k_u of the signing key, as opposed to the security parameter 1^λ (Note, in our schemes the signing key will be random, so k_u is simply the length of the signing key.). Denote the number of leak oracle query as q_l. Denote $\mathfrak{state}_i$ as the value of $\mathfrak{state}$ where the i-th leak oracle is queried.

The first family $\mathcal{F}_{bdd}$ is the length-bounded family studied by the prior works (e.g. [9,11]), while the second family, $\mathcal{F}_{vk-ow}$, is similar to the auxiliary input families introduced by Dodis *et al.* [7]. It only assumes that the signing key is "hard to compute" given the leakage.

– Let $\mathcal{F}_{bdd}(\ell_u(k_u))$ be the class of all polynomial-time computable functions f of constant output size. We have further restriction on the security model such that for all $f_1, \ldots, f_{q_l}$ queried in the leak oracle, $\sum_{i=1}^{q_l} |f_i(\mathfrak{state}_i)| \leq \ell_u$, where ℓ_u is the number of bits the attacker is allowed to learn about the signing key. The trivial leakage upper bound is $\ell_u \leq k_u$, which means that the total leakage is less than the min-entropy of the signing key.
– Let $\mathcal{F}_{vk-ow}(g^{\mathrm{u}}(k_u))$ be the class of all polynomial-time computable functions f; and let $\mathcal{S}$ denote a set of q_s signing oracle output, such that for $f_1, \ldots, f_{q_l} \in \mathcal{F}_{vk-ow}$, given $\mathsf{vk}, \mathcal{S},$ and $\{f_i(\mathfrak{state}_i)\}_{i \in [1, q_l]}$, (for $(\mathsf{sk}, \mathsf{vk}, r, \{r_i\}_{i \in [1, q_l]})$ that is randomly generated), no PPT algorithm can find sk with probability greater than $g^{\mathrm{u}}(k_u)$, where $g^{\mathrm{u}}(k_u) \geq 2^{-k_u}$ is the hardness parameter. Our goal is to make $g^{\mathrm{u}}(k_u)$ as large (i.e., as close to $\mathrm{negl}(k_u)$) as possible.

Differences from Previous Definitions. Our model bears some differences from the existing model of *public key encryption (PKE) with auxiliary input* [7], *identity-based encryption (IBE) with auxiliary input* [14], *signatures with entropy-bounded leakage* [11], and *signatures with auxiliary input* [12].

1. We models the leakage as an adaptive oracle, with auxiliary function as the input. The adversary can modify its query according to the previous oracle output, as in [16,11]. (In [7], since the CPA security for PKE only has the leak oracle query, they modeled it as a single oracle query.) Although the adversary has to select the set $\mathbb{F}$ at the beginning, he can still change the sequence of the query f_i according to the changes of the internal state of the challenger.
2. In [7,14], they only define the CPA security, where no decryption oracle is provided to the adversary. For EUF-CMA, we have to provide the signing oracle. Similar to the decryption oracle for CCA security, the signing oracle in EUF-CMA uses the secret key to calculate the oracle output. As a result, the output from the signing oracle (or decryption oracle in the case of PKE/IBE) contains some information about the secret key. Therefore, they have to be considered when defining the classes of function families. The models in [11,12] do not need to consider this restriction.
3. Our model is similar to the EU-CMAA model in [12]. After removing step 1 of our security game, and restricting leakage family to be exponentially hard-to-invert, we obtain the EU-CMAA model.

Definition 1. *A signature scheme is said to be*

– *$(\ell_u(k_u))$-sLB-EUF-CMA (selective length-bounded EUF-CMA) secure if it is EUF-CMA secure w.r.t. family $\mathcal{F}_{bdd}(\ell_u(k_u))$.*
– *$(g^{\mathrm{u}}(k_u))$-sAI-EUF-CMA (selective auxiliary input EUF-CMA) secure if it is EUF-CMA secure w.r.t. family $\mathcal{F}_{vk-ow}(g^{\mathrm{u}}(k_u))$.*

3 Building Blocks

We use the notation $\mathsf{negl}(\cdot)$ to refer to some *negligible* function. We use PPT stands for probabilistic polynomial-time.

3.1 Second-Preimage Resistant

A family of second-preimage resistant (SPR) functions is a pair of polynomial time algorithms:

– KGen: on input 1^λ, outputs a description $s \in \{0,1\}^*$ of a function F.
– F: on input s and $x \in \{0,1\}^{\mu(\lambda)}$, outputs $y \in \{0,1\}^{\kappa(\lambda)}$.

The second-preimage resistant property is that if given a randomly chosen $x \in \{0,1\}^{\mu(\lambda)}$ and a description of a randomly chosen function $s \leftarrow \mathsf{KGen}(1^\lambda)$, it is computationally infeasible to find an input $x' \in \{0,1\}^{\mu(\lambda)}$ such that $x' \neq x$ and

$\mathsf{F}(s,x) = \mathsf{F}(s,x')$. For simplicity, we usually write $\mathsf{F}(s,x)$ as $\mathsf{F}_s(x)$. We further say that $\mathcal{F} = (\mathsf{KGen}, \mathsf{F})$ is a family of *public-coin* SPR functions, if it satisfies the definition above even when $\mathcal{A}$ takes as input also the internal randomness that was used by $\mathsf{KGen}(1^\lambda)$ for sampling the function.

3.2 Statistical Non-interactive Witness-Indistinguishable Proof

For a language L with witness relation R_L, the proof system is a triplet of algorithms (CRSGen, P, V), where CRSGen is an algorithm generating common reference strings crs, P and V are the prover and verifier algorithms, respectively.

– CRSGen: on input 1^λ, outputs a common reference string crs.
– P: on input crs, x and w, outputs a proof π if $(x,w) \in R_L$.
– V: on input crs, x and π, outputs 1 for accept and 0 for reject.

We require that the proof system is complete, sound, and statistically witness indistinguishable given the crs.

3.3 Admissible Hash Functions

Admissible hash function [17] allows the partition of message space into two subsets, which we will label as red and blue. We follows the definition in [18].

For $K \in \{0,1,\perp\}^{\tau(\lambda)}$, we define the function $F_K : \{0,1\}^{\tau(\lambda)} \to \{\mathsf{Red}, \mathsf{Blue}\}$:

$$F_K(y) := \begin{cases} \mathsf{Red} & \text{if } \forall i \in \{1,\ldots,\tau(\lambda)\} : K_i = y_i \text{ or } K_i = \perp, \\ \mathsf{Blue} & \text{otherwise.} \end{cases}$$

For any $u = u(\lambda) < \tau(\lambda)$, let $\mathcal{K}_{u,\lambda}$ denote the uniform distribution over $\{0,1,\perp\}^{\tau(\lambda)}$ conditioned on exactly u positions having $\perp$ values. Let $\mathcal{H} = \{\mathcal{H}_\lambda\}$ be a hash function ensemble, where each $H \in \mathcal{H}_\lambda$ is a polynomial-time computable function $H : \{0,1\}^* \to \{0,1\}^{\tau(\lambda)}$. For each $H \in \mathcal{H}_\lambda$, we define the function $F_{K,H} : \{0,1\}^* \to \{\mathsf{Red}, \mathsf{Blue}\}$, which colors the space $\{0,1\}^*$ according to $F_{K,H}(x) = F_K(H(x))$. We can further define $\mathcal{H}$ as a *public-coin* admissible hash function ensemble as in [11].

3.4 Commitment Scheme

A commitment scheme allows one to commit to a value while keeping it hidden, with the ability to reveal the committed value later. A commitment scheme for a message space $\mathcal{M}$ has the following syntax:

– $\mathsf{KeyGen}(1^\lambda)$: It takes as input the security parameter λ and outputs a commitment key ck.
– $\mathsf{Commit}(ck, m)$: It takes as input ck and a message $m \in \mathcal{M}$, and outputs (c, d) where c is the commitment value and d is the decommitment value.
– $\mathsf{Decommit}(ck, c, d, m)$: It takes as input ck, c, d, and a message m, output 1 or 0 indicating if c is a valid commitment to m.

A commitment scheme has Correctness, Hiding and Binding properties. We denote $ck_h \leftarrow \mathsf{KeyGen}_h(1^\lambda)$ as the commitment key generated by the challenger of the hiding game. We denote $(ck_b, bk) \leftarrow \mathsf{KeyGen}_b(1^\lambda)$ as the commitment key and the binding key generated by the challenger of the binding game.

Statistically Hiding Commitment Scheme with Extraction has the following properties.

- **Indistinguishable Key Generation.** The distribution of hiding keys and the distribution of binding keys are computationally indistinguishable.
- **Statistically Hiding.** $(\mathsf{KeyGen}_h, \mathsf{Commit}, \mathsf{Decommit})$ is a statistically hiding commitment scheme. The distribution of the commitment values generated by different messages are statistically indistinguishable, if the commitment key is generated by KeyGen_h.
- **Extraction.** There exists an algorithm Dec such that $(\mathsf{KeyGen}_b, \mathsf{Commit}, \mathsf{Dec})$ is a public key encryption scheme with errorless decryption.

Tag-Based Commitment Scheme. There are a number of commitment schemes [19,20,21,22] that use extra tags in commit and decommit. For simulation-sound commitment scheme, we further assume that commitments are labeled with a tag *tag*. Therefore, the algorithms Commit_{tag} and $\mathsf{Decommit}_{tag}$ use *tag* as an extra input. The tag-based commitment, for a message space $\mathcal{M}$ and a tag space $\{0,1\}^n$, has the following syntax:

- $\mathsf{KeyGen}(1^\lambda)$: It takes as input the security parameter λ and outputs a commitment key ck.
- $\mathsf{Commit}_{tag}(ck, m)$: It takes as input the commitment key ck, a message $m \in \mathcal{M}$ and a $tag \in \{0,1\}^n$, and outputs (c, d) where c is the commitment value and d is the decommitment value.
- $\mathsf{Decommit}_{tag}(ck, c, d, m)$: It takes as input ck, c, d, a message m and a *tag*, output 1 or 0 indicating if c is a valid commitment to m for *tag*.

We can similarly define the correctness, hiding and binding properties for the same *tag*. If the hiding property holds for some tags, we call them hiding tags. If the binding property holds for some tags, we call them binding tags.

3.5 Statistically Hiding Tag-based Commitment Scheme with Extraction

Our construction uses a commitment scheme which has the following properties defined above: statistically hiding for hiding tags; computationally binding for binding tags; extraction for binding tags; and indistinguishable Key Generation. We also need the oblivious sampling of KeyGen: The distribution of ck is statistically-close to the uniform distribution.

This commitment scheme is similar to some existing primitives in the literature. For $\mathcal{R}$-Lossy PKE [11], the *lossiness under lossy tags* (resp. *decryption under injective tags*, decryption algorithm) is similar to the statistically hiding for hiding tags (resp. computationally binding for binding tags, extraction algorithm) of the commitment scheme. Both primitives requires the indistinguishable key generation algorithms and oblivious sampled public keys. On the other hand, these two primitives still have some differences. For $\mathcal{R}$-Lossy PKE, there are different ways to define the decryption relations, such as $\mathcal{R}^{\mathbf{EQ}}$ for equality relation, and $\mathcal{R}^{\mathbf{BM}}$ for bit matching relation. For the commitment scheme, no explicit decryption algorithm is defined. Such algorithm is only defined in the extraction property and does not have other decryption relations.

For selective opening commitment [23], it considers the case that the commitment is still hiding even if a number of commitments are opened (the decommitment values are given). This security requirement is important in zero-knowledge proofs, especially in the concurrent composition of zero-knowledge proofs. For our construction, we need the property that the commitment is still *binding* even if a number of commitments are opened. Therefore we use the tag to partition which commitments can be opened and which commitment is binding.

4 Construction

In this section, we present a fully leakage-resilient signature with auxiliary input. We use the following primitives in our construction:

- Let $\mathcal{F} = (\mathsf{KeyGen}_{\mathsf{SPR}}, \mathsf{F})$ be a family of public-coin second pre-image resistant functions $\mathsf{F}_s : \{0,1\}^{\mu(\lambda)} \to \{0,1\}^{\kappa(\lambda)}$ for some $\kappa(\lambda) < \mu(\lambda)$.
- Let $\mathcal{C} = (\mathsf{KeyGen}_{\mathsf{Com}}, \mathsf{Commit}, \mathsf{Open})$ be a family of statistically hiding tag-based commitment scheme with extraction.
- Let $\mathcal{H}$ be a public-coin admissible hash function ensemble, which maps from the message space to the tag space of $\mathcal{C}$.
- Let $\mathit{\Pi} = (\mathsf{CRSGen}, \mathsf{P}, \mathsf{V})$ be a SNIWI proof system for the language

$$L = \{(s, y, t, \boldsymbol{ck}, C) : \exists (x, \omega) \ \text{ st } \ C = \mathsf{Commit}_t(\boldsymbol{ck}, x; \omega) \quad \text{and} \quad \mathsf{F}_s(x) = y\}.$$

Now we give our signature scheme as follows:

- $\mathsf{KeyGen}(1^\lambda)$: On input 1^λ, it samples
  - a uniformly distributed $x \leftarrow \{0,1\}^*$;
  - a function description $s \leftarrow \mathsf{KeyGen}_{\mathsf{SPR}}(1^\lambda)$ from the SPR family;
  - a description of an admissible hash function $H \leftarrow \mathcal{H}_\lambda$;
  - $\boldsymbol{ck} \leftarrow \{0,1\}^*$ to be used as commitment keys of the statisically hiding tag-based commitment scheem with extraction $\mathcal{C}$;
  - $\mathsf{crs} \leftarrow \{0,1\}^*$ to be used as a common reference string of the SNIWI proof system $\mathit{\Pi}$.

  It computes $y = \mathsf{F}_s(x)$. It outputs the signing key $sk = x$ and the verification key $vk = (s, y, H, \boldsymbol{ck}, \mathsf{crs})$.
- $\mathsf{Sign}(sk, M)$: On input the signing key x and a message M, it picks some randomness ω and computes $C = \mathcal{C}.\mathsf{Commit}_{H(M)}\ (\boldsymbol{ck}, x; \omega)$. It obtains a proof $\pi \leftarrow \mathsf{P}(\mathsf{crs}, (s, y, H(M), \boldsymbol{ck}, C), (x, \omega))$ and outputs the signature (C, π).

– $\mathsf{Verify}(vk, M, \sigma)$: On input vk, the message M and the signature $\sigma = (C, \pi)$, it outputs 1 if and only if $\mathsf{V}(\mathsf{crs}, (s, y, H(M), \boldsymbol{ck}, C), \pi) = 1$.

Theorem 1. *The signature is* $(\mathsf{negl}(\lambda))$*-sAI-EUF-CMA assuming the existence of the schemes* $\mathcal{F}, \mathcal{H}, \mathcal{C}$ *and* Π *with properties described above.*

The proof of the theorem will be given in the full version of the paper.

Remark. We can prove the security of our scheme under the auxilary input model by [12], with exponentially hard-to-invert leakage. The security proof is very similar and is omitted. Note that our sAI-EUF-CMA model is incomparable to the security model in [12].

5 Efficient Instantiation

We show that our generic construction can be efficiently instantiated based on the Decision Linear Assumption [24]: Given (u, v, h, u^a, v^b, h^c) in a group $\mathbb{G}$, it is hard to decide if $c = a + b$.

5.1 Building Blocks

Let $\mathcal{G}(1^\lambda)$ be a PPT algorithms which outputs $(\mathbb{G}, \mathbb{G}_T, p, g, \hat{e})$, where $\mathbb{G}$ and $\mathbb{G}_T$ are groups of prime order p, g is the generator of $\mathbb{G}$, and $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear maps.

Family of SPR Functions. The linear-based SPR function family $\mathcal{F} = (\mathsf{KeyGen}_{\mathsf{SPR}}, \mathsf{F})$ is defined as [5]:

– $\mathsf{KeyGen}_{\mathsf{SPR}}$: On input 1^λ, it samples $(\mathbb{G}, \mathbb{G}_T, p, g, \hat{e}) \leftarrow \mathcal{G}(1^\lambda)$. Then it samples $s = (h_1, \ldots, h_{k(\lambda)}, h'_1, \ldots, h'_{k(\lambda)}) \leftarrow \mathbb{G}^{2k(\lambda)}$ uniformly at random and outputs the description s.
– F: On input a description s and an input $(g_1, \ldots, g_{k(\lambda)}) \in \mathbb{G}^{k(\lambda)}$, it outputs:

$$\mathsf{F}(s, (g_1, \ldots, g_{k(\lambda)})) = \left(\prod_{i=1}^{k(\lambda)} \hat{e}(h_i, g_i), \prod_{i=1}^{k(\lambda)} \hat{e}(h'_i, g_i) \right).$$

Admissible Hash Function. Boneh and Boyen [17] showed how to construct admissible hash functions from collision-resistant hash functions. Collision-resistant hash functions can be constructed from decision linear assumption [17].

Statistically Hiding Tag-Based Commitment Scheme with Extraction. We give an efficient instantiation for the message space $\mathcal{M} = \mathbb{G}$ and the tag space $[1, n]$ as follows. It is similar to the linear-based commitment of the NIWI proof system of Groth and Sahai [25].

– KeyGen: On input 1^λ, it samples $(\mathbb{G}, \mathbb{G}_T, p, g, \hat{e}) \leftarrow \mathcal{G}(1^\lambda)$. It picks a random set $\mathcal{S} \subset [1, n]$ and runs the following subroutines for $j \in [1, n]$:

- KeyGen_h: If $j \notin \mathcal{S}$, it chooses 3 independently and uniformly distributed elements $u_{0,j}, u_{1,j}, u_{2,j} \leftarrow \mathbb{G}$ and sets $\boldsymbol{u_{1,j}} = (u_{0,j}, u_{1,j}, 1)$ and $\boldsymbol{u_{2,j}} = (u_{0,j}, 1, u_{2,j})$. Then it samples $\boldsymbol{u_{0,j}} \leftarrow \mathbb{G}^3$ at random, such that $\boldsymbol{u_{0,j}}, \boldsymbol{u_{1,j}}, \boldsymbol{u_{2,j}}$ are linearly independent. It sets $\boldsymbol{ck_j} = (\boldsymbol{u_{0,j}}, \boldsymbol{u_{1,j}}, \boldsymbol{u_{2,j}})$.
- KeyGen_b: If $j \in \mathcal{S}$, it samples a random $u_{0,j} \leftarrow \mathbb{G}$, two random $\alpha_j, \beta_j \in \mathbb{Z}_p$ and sets $u_{1,j} = u_{0,j}^{\alpha_j}, u_{2,j} = u_{0,j}^{\beta_j}$. It computes $\boldsymbol{u_{1,j}} = (u_{0,j}, u_{1,j}, 1)$ and $\boldsymbol{u_{2,j}} = (u_{0,j}, 1, u_{2,j})$. Then it samples $r, s \in \mathbb{Z}_p$ uniformly at random and sets $\boldsymbol{u_{0,j}} = (u_{0,j}^{r+s}, u_{1,j}^{r}, u_{2,j}^{s})$. It sets $\boldsymbol{ck_j} = (\boldsymbol{u_{0,j}}, \boldsymbol{u_{1,j}}, \boldsymbol{u_{2,j}})$ and $bk_j = (\alpha_j, \beta_j)$.

Finally, it outputs $\boldsymbol{ck} = (\boldsymbol{ck_1}, \ldots, \boldsymbol{ck_n})$.

– Commit: On input $(\boldsymbol{ck}, m, tag)$, it samples $(d_0, d_1, d_2) \leftarrow \mathbb{Z}_p^3$ uniformly at random and computes $\boldsymbol{c} = (m, 1, 1) \prod_{j=0}^{2} \boldsymbol{u_{j,tag}}^{d_j}$. The commitment value is $\boldsymbol{c}$ and the decommitment value is $\boldsymbol{d} = (\boldsymbol{u_{0,tag}}^{d_0}, \boldsymbol{u_{1,tag}}^{d_1}, \boldsymbol{u_{2,tag}}^{d_2})$.
– $\mathsf{Decommit}$: On input $(\boldsymbol{ck}, \boldsymbol{c}, \boldsymbol{d}, m, tag)$, it outputs 1 if $\boldsymbol{c} = (m, 1, 1) \prod_{j=0}^{2} \boldsymbol{u_{j,tag}}^{d_j}$.
– Dec: On input $(\boldsymbol{ck}, \boldsymbol{c}, bk, tag)$, it parses $\boldsymbol{c} = (c_0, c_1, c_2)$ and $bk_{tag} = (\alpha_{tag}, \beta_{tag})$. If $tag \notin \mathcal{S}$, it returns $\perp$. Otherwise, it outputs $m = c_0 \cdot c_1^{-\frac{1}{\alpha_{tag}}} \cdot c_2^{-\frac{1}{\beta_{tag}}}$.

We prove that the above scheme has the desired properties.

$\mathsf{Indistinguishable\ Key\ Generation}$. The only difference between KeyGen_h and KeyGen_b is how $\boldsymbol{u_{0,j}}$ is chosen. If the decision linear assumption holds, no PPT adversary can distinguish between them.

$\mathsf{Statistically\ Hiding}$. For the hiding tags $tag \notin \mathcal{S}$, $\boldsymbol{u_{0,tag}}, \boldsymbol{u_{1,tag}}, \boldsymbol{u_{2,tag}}$ are linearly independent. So they form a basis for $\mathbb{G}^3$ and the commitment is perfectly hiding, which implies statistically hiding.

$\mathsf{Extraction}$. For the binding tags $tag \in \mathcal{S}$, the commitment $\boldsymbol{c} = (m \cdot u_{0,tag}^{d_1+d_2+d_0(r+s)}, u_{1,tag}^{d_1+rd_0}, u_{2,tag}^{d_2+sd_0})$ is a linear encryption [24] of the message m with $bk = (\alpha_{tag}, \beta_{tag})$ as the decryption key.

$\mathsf{Oblivious\ Sampling}$. The output of KeyGen_h is statistically-close to the uniform distribution.

$\mathsf{Computationally\ Binding}$. For the binding tags $tag \in \mathcal{S}$, the adversary cannot output a commitment which can be decommit by two different messages. It is because by the extraction property, the simulator can use bk to run Dec on the commitment and obtain the original message. If the commitment can be decommit by two different messages, it breaks the linear encryption scheme [24].

SNIWI Proof System. The NIWI proof system of Groth and Sahai [25] can be instantiated by the decision linear assumption. Since it is perfectly hiding, it is also statistically hiding.

5.2 Efficiency Analysis

Summarizing the previous instantiation under the decision linear assumption, we can see that the secret key size is $2k$ elements in $\mathbb{G}$ (for $k \geq 3$), and the signature size is $6(k+1)$ elements in $\mathbb{G}$. The signing time is dominated by the $9k$

exponentiation in $\mathbb{G}$. The verification time is dominated by the $3(k+2)$ pairing operations.

If one is also willing to make the double pairing assumption as in [5], the signature scheme can be more efficient by setting $k = 1$. Using the SXDH-based commitment and SNIWI proofs by Groth and Sahai [25] can also give a more efficient construction.

5.3 Fully Leakage-Resilient Signatures with Selective Continuous Auxiliary Input

We extend the fully leakage-resilient signatures with selective auxiliary input to withstand attacks with continuous leakage. We only briefly describe the intuition here due to the space limit. We generalize the leakage-resilient one-way function [5] to the selective continuous auxiliary leakage setting. Compare with the continuous leakage-resilient one-way relation [11], we firstly added the support of selective auxiliary input leakage, instead of the entropy-bounded leakage. Secondly, we also allow the leakage of the randomness used in the KeyGen and Update. The generic construction of the fully leakage-resilient signatures with selective continuous auxiliary input is similar to [11]. For instantiation, we can follow the continuous leakage-resilient identity-based encryption by Yuen *et al.* [14]. We use the "update master key" algorithm in [14] as the refresh algorithm in the one-way relation.

References

1. Akavia, A., Goldwasser, S., Vaikuntanathan, V.: Simultaneous Hardcore Bits and Cryptography against Memory Attacks. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 474–495. Springer, Heidelberg (2009)
2. Naor, M., Segev, G.: Public-Key Cryptosystems Resilient to Key Leakage. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 18–35. Springer, Heidelberg (2009)
3. Alwen, J., Dodis, Y., Naor, M., Segev, G., Walfish, S., Wichs, D.: Public-Key Encryption in the Bounded-Retrieval Model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 113–134. Springer, Heidelberg (2010)
4. Chow, S.S.M., Dodis, Y., Rouselakis, Y., Waters, B.: Practical leakage-resilient identity-based encryption from simple assumptions. In: Al-Shaer, E., Keromytis, A.D., Shmatikov, V. (eds.) CCS 2010, pp. 152–161. ACM (2010)
5. Dodis, Y., Haralambiev, K., López-Alt, A., Wichs, D.: Cryptography against continuous memory attacks. In: FOCS 2010, pp. 511–520. IEEE Computer Society (2010)
6. Brakerski, Z., Kalai, Y.T., Katz, J., Vaikuntanathan, V.: Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage. In: FOCS 2010, pp. 501–510. IEEE Computer Society (2010)
7. Dodis, Y., Goldwasser, S., Tauman Kalai, Y., Peikert, C., Vaikuntanathan, V.: Public-Key Encryption Schemes with Auxiliary Inputs. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 361–381. Springer, Heidelberg (2010)

8. Alwen, J., Dodis, Y., Wichs, D.: Leakage-Resilient Public-Key Cryptography in the Bounded-Retrieval Model. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 36–54. Springer, Heidelberg (2009)
9. Katz, J., Vaikuntanathan, V.: Signature Schemes with Bounded Leakage Resilience. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 703–720. Springer, Heidelberg (2009)
10. Malkin, T., Teranishi, I., Vahlis, Y., Yung, M.: Signatures Resilient to Continual Leakage on Memory and Computation. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 89–106. Springer, Heidelberg (2011)
11. Boyle, E., Segev, G., Wichs, D.: Fully Leakage-Resilient Signatures. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 89–108. Springer, Heidelberg (2011)
12. Faust, S., Hazay, C., Nielsen, J.B., Nordholt, P.S., Zottarel, A.: Signature schemes secure against hard-to-invert leakage. Cryptology ePrint Archive, Report 2012/045 (2012), http://eprint.iacr.org/
13. Boneh, D., Boyen, X.: Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)
14. Yuen, T.H., Chow, S.S.M., Zhang, Y., Yiu, S.M.: Identity-Based Encryption Resilient to Continual Auxiliary Leakage. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 117–134. Springer, Heidelberg (2012)
15. Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
16. Lewko, A., Rouselakis, Y., Waters, B.: Achieving Leakage Resilience through Dual System Encryption. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 70–88. Springer, Heidelberg (2011)
17. Boneh, D., Boyen, X.: Secure Identity Based Encryption Without Random Oracles. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 443–459. Springer, Heidelberg (2004)
18. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai Trees, or How to Delegate a Lattice Basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer, Heidelberg (2010)
19. Dolev, D., Dwork, C., Naor, M.: Nonmalleable cryptography. SIAM J. Comput. 30(2), 391–437 (2000)
20. MacKenzie, P.D., Yang, K.: On Simulation-Sound Trapdoor Commitments. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 382–400. Springer, Heidelberg (2004)
21. Lin, H., Pass, R., Venkitasubramaniam, M.: Concurrent Non-malleable Commitments from Any One-Way Function. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 571–588. Springer, Heidelberg (2008)
22. Zhang, Z., Cao, Z., Ding, N., Ma, R.: Non-malleable Statistically Hiding Commitment from Any One-Way Function. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 303–318. Springer, Heidelberg (2009)
23. Dwork, C., Naor, M., Reingold, O., Stockmeyer, L.J.: Magic functions. J. ACM 50(6), 852–921 (2003)
24. Boneh, D., Boyen, X., Shacham, H.: Short Group Signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
25. Groth, J., Sahai, A.: Efficient Non-interactive Proof Systems for Bilinear Groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008)

Adaptive CCA Broadcast Encryption with Constant-Size Secret Keys and Ciphertexts

Duong-Hieu Phan[1,2,3], David Pointcheval[2,3], Siamak F. Shahandashti[1,2], and Mario Strefler[2,3]

[1] LAGA, University of Paris 8
[2] ENS / CNRS
[3] INRIA
{phan,pointche,fshahand,strefler}@di.ens.fr

Abstract. We consider designing broadcast encryption schemes with constant-size secret keys and ciphertexts, achieving chosen-ciphertext security. We first argue that known CPA-to-CCA transforms currently do not yield such schemes. We then propose a scheme, modifying a previous selective CPA secure proposal by Boneh, Gentry, and Waters. Our proposed scheme has constant-size secret keys and ciphertexts and we prove that it is selective chosen-ciphertext secure based on standard assumptions. Our scheme has ciphertexts that are shorter than those of the previous CCA secure proposals. Then we propose a second scheme that provides the functionality of both broadcast encryption and revocation schemes simultaneously using the same set of parameters. Finally we show that it is possible to prove our first scheme adaptive chosen-ciphertext secure under reasonable extensions of the bilinear Diffie-Hellman exponent and the knowledge of exponent assumptions. We prove both of these extended assumptions in the generic group model. Hence, our scheme becomes the first to achieve constant-size secret keys and ciphertexts (both asymptotically optimal) and adaptive chosen-ciphertext security at the same time.

1 Introduction

A *broadcast encryption* is a cryptographic scheme that enables encryption of broadcast content such that only a set of target users, selected at the time of encryption, can decrypt the content. Apparent applications include group communication, pay TV, content protection, file system access control, and geolocation.

A crucial aspect of any cryptographic scheme, which arguably decides its fate of being used in practice, is its efficiency. Since one of the most prominent applications of broadcast encryption is real-time broadcasting, ciphertext size is at the heart of efficiency measures for such schemes, and constructions with constant-size ciphertexts are desirable. Indeed, if one allows the ciphertext size to grow linearly with the number of target users, construction of secure broadcast encryption becomes trivial. Other important measures of efficiency for broadcast encryption include the secret and public key sizes and the encryption and decryption times.

A broadcast encryption scheme can be *static* or *dynamic*, depending on if the system users need to be fixed once and for all at the setup stage or if it supports new users joining the system at an arbitrary time, incurring only incremental parameter changes. Evidently, dynamic schemes are more flexible and hence more desirable in practical applications.

An important security paradigm for broadcast encryption schemes is that of *adaptive* security. This paradigm captures the fact that an adversary might choose to compromise keys in the system adaptively, based on its acquired knowledge of the system parameters and previously compromised keys and ciphertexts. Such a definition is widely accepted as the proper notion of security for broadcast encryption schemes and there are schemes proposed in the literature that provably achieve security against adaptive adversaries.

On the other hand, security against chosen ciphertext attacks (CCA) is a fundamental notion of security for any encryption scheme, broadcast encryption included. Although there have been a number of proposed broadcast encryption schemes that are secure against chosen plaintext attacks (CPA), the CPA-to-CCA transformations in the literature do not seem to yield CCA secure broadcast encryption schemes with constant-size ciphertexts.

Adaptive and CCA security, and constant-size ciphertexts, have all three been separately achieved for broadcast encryption. However, there has not been any proposal that achieves all three simultaneously. In this paper, we propose a broadcast encryption with constant-size ciphertexts and prove it adaptive CCA secure under assumptions that are reasonable generalizations of previous assumptions in the literature.

The literature on broadcast encryption mainly considers two categories of such schemes and each work usually provides solutions that are efficient only for one of the two cases, depending on whether the content is broadcast to a very small or a very large proportion of registered users. The party who encrypts the content, hence either determines their intended set of target users or that of revoked users, respectively, as an input to the encryption algorithm. Consequently, the latter category of schemes are sometimes called *revocation* schemes.

Consider the pay-TV application in which the content of the broadcast consists of several TV channels. Normally, there are a number of basic channels that are usually bundled together and provided to most of the customers in different packages, and also there are a number of more specialized channels (e.g., pay-per-view) that are of the interest of a small proportion of customers. Hence we face a scenario in which both of the above categories of schemes are simultaneously needed to broadcast the content. Nevertheless, there has been no proposal in the literature that provides both functionalities efficiently, and hence the existing efficient solution to the above scenario is to set up two parallel schemes, each covering part of the broadcast content. In this paper, we propose a scheme that can handle both cases efficiently, providing a solution to the above scenario that does not require maintaining two parallel sets of system parameters.

1.1 Related Work

Broadcast encryption was first formalized by Fiat and Naor [16]. Their scheme is a private key scheme and proved secure against an upper bounded number of colluders. Fully collusion secure (private-key) broadcast encryption was first proposed in [20], which introduced the subset cover framework that became the basis for many subsequent proposals, including [15] which proposed the first public key broadcast encryption.

Boneh, Gentry, and Waters [4] were first to propose a fully collusion-resistant public key broadcast encryption in which the ciphertext size is constant. In all the previous schemes, the size of the ciphertext is linear in the size of the target set. In this paper we limit our attention to such schemes. They proposed two schemes, respectively CPA and CCA secure, both in the selective model of security. Dynamic broadcast encryption was proposed in [11] where they designed CPA secure schemes that were only partially adaptive secure. Strictly speaking, their scheme is a *revocation* scheme, in which the set of revoked users is selected at the time of encryption, and in turn, any user outside of the revoked set is able to decrypt. [10] proposed identity-based broadcast encryption and gave a selective CPA secure scheme.

Adaptive security was proposed by [17] where they gave several schemes achieving adaptive CPA security, including two broadcast encryption schemes and two identity-based broadcast encryption (IBBE) schemes, one of each achieving constant-size ciphertexts in the random oracle model. The schemes proposed in [30] and [19], respectively a broadcast encryption and a revocation scheme, are the only schemes secure under static assumptions (as opposed to the so called q-based ones). The latter work also proposes an identity-based revocation scheme which is proved selective CPA secure. Recently, the first adaptive CCA secure schemes were proposed by [24], although their schemes do not have constant-size ciphertexts.

1.2 Our Contributions

We propose an efficient dynamic broadcast encryption scheme (called OurBE) and prove that it is selective CCA secure assuming the widely-used bilinear Diffie-Hellman exponent (BDHE) assumption and a universal one-way hash function (UOWHF). The proposed scheme has constant-size ciphertexts (only two group elements), constant-size secret keys (only one group element), and a public key which grows linearly with the number of users in the system. We construct our scheme by modifying a selective CPA secure scheme (dubbed BGW_1 from now on) by Boneh, Gentry, and Waters [4]. Our modification is minimal in the sense that our scheme has exactly the same ciphertext and secret key sizes as that of BGW_1, and is proved secure under the same assumption, plus the comparatively weak UOWHF assumption. The minor difference is that our scheme has one extra element in the linearly-growing public key. The only other CCA secure scheme with constant-size ciphertexts is a modified version of BGW_1 by the same authors (dubbed BGW_2 from now on), which has ciphertexts that are double the size of our scheme (i.e., four group elements vs. our two). BGW_2 is proved selective CCA secure under BDHE, plus the assumption that a signature scheme used in the construction is strongly unforgeable, which is an assumption of comparable strength as UOWHF.

We also propose an *inclusive-exclusive* broadcast encryption scheme which can act as both a broadcast encryption and a revocation scheme at the same time, as it allows the flexibility to specify either the target set or the revoked set at the time of encryption. The ciphertext and the secret key are still only two and one group elements, respectively, but we need to add one group element per user to the already linearly-growing public key which results in a public key which is 1.5 times that of BGW_1.

Next, we show that it is possible to prove OurBE adaptive CCA secure under generalized versions of existing assumptions. Particularly, we propose generalized versions of the BDHE and the knowledge-of-exponent (KEA) assumptions, and prove that both hold in the generic group model. We argue that both of these are intuitive and reasonable generalizations of accepted assumptions, and in turn, enable achieving the highest level of security with highly-efficient parameters. Namely, OurBE is provably adaptive CCA secure with constant-size ciphertexts and secret keys, and it is the first scheme to achieve such properties.

2 Preliminaries

In this section we review the notation we use, the BDHE and GBDHE assumptions, and the notions of security for dynamic broadcast encryption and universal one-way hash function.

Notation. We use the following typefaces: Roman X for constants, italic X for variables, sans serif X for algorithms, and calligraphic $\mathcal{X}$ for oracles. Let $\mathbb{G}$ and $\mathbb{G}_\mathrm{T}$ be groups of order p, and $e : \mathbb{G} \times \mathbb{G} \mapsto \mathbb{G}_\mathrm{T}$ be a bilinear map. Let g be a generator of $\mathbb{G}$ and $g_\mathrm{T} = e(g, g)$.

2.1 Dynamic Broadcast Encapsulation

Broadcast encryption is conventionally formalized as *broadcast encapsulation* in which, instead of a ciphertext, a session key is produced, which is required to be indistinguishable from random. Such a scheme can provide public encryption functionality in combination with a symmetric encryption through the hybrid encryption (a.k.a. KEM-DEM) paradigm [7]. We hence use the terms encryption and encapsulation interchangeably.

Following [11], we define (public-key) *dynamic* broadcast encapsulation as a tuple of four algorithms $\mathsf{BE} = (\mathsf{Setup}, \mathsf{Join}, \mathsf{Encaps}, \mathsf{Decaps})$ where:

- $\mathsf{Setup}(1^k)$ outputs $(\mathit{MSK}, \mathit{EK})$ containing the master secret key and the (initial) encryption key;
- $\mathsf{Join}(\mathit{MSK}, i)$ outputs the key pair (sk_i, pk_i) for user i, and appends pk_i to EK;
- $\mathsf{Encaps}(\mathit{EK}, S)$ for a set of users S outputs (H, K) containing a ciphertext (a.k.a. header) and a session key; and
- $\mathsf{Decaps}(\mathit{EK}, sk_i, S, H)$ outputs K if $i \in S$ and $\perp$ otherwise.

Adaptive CCA security for BE is defined via the following experiments for $b \in \{0, 1\}$ between the challenger C and the adversary A:

1. *Setup:* C runs $\mathsf{Setup}(1^k)$ and gives EK to A;
2. *Query:* A arbitrarily issues the following oracle queries:
   - *join* oracle query $\mathcal{J}oin(i)$: C runs $\mathsf{Join}(\mathit{MSK}, i)$ and gives pk_i to A;
   - *corruption* oracle query $\mathcal{C}or(i)$: C gives sk_i to A;
   - *decapsulation* oracle query $\mathcal{D}ec(i, S, H)$: C runs $\mathsf{Decaps}(\mathit{EK}, sk_i, S, H)$ and gives K to A;
3. *Challenge:* A outputs a set S^* on which it wants to be challenged; C runs $\mathsf{Encaps}(\mathit{EK}, S^*)$ and gets (H^*, K^*), then sets $K = K^*$ if $b = 0$ or picks a random K if $b = 1$, and finally gives (H^*, K) to A;
4. *Query:* A issues further oracle queries as the previous query phase;
5. *Guess:* A outputs a guess b'. The experiment outputs 1 if $b' = b$ and there is no $i^* \in S^*$ for which there has been a $\mathcal{C}or(i^*)$ or $\mathcal{D}ec(i^*, S^*, H^*)$ query. The experiment outputs 0 otherwise.

For any adversary A, we define its *advantage* against BE in an adaptive CCA attack to be the difference between the probability that the above experiment for $b = 0$ outputs 1 and the probability that the experiment for $b = 1$ outputs 1. The scheme is said to be adaptive CCA secure if for any adversary A its advantage against BE in an adaptive CCA attack is negligible in k.

Selective security is defined via similar games with the difference that A commits to the set S^* before the setup phase. For CPA security, A does not get to query the decryption oracle. We sometimes use SCPA, SCCA, ACPA, and ACCA as shorthands referring to selective CPA, selective CCA, adaptive CPA, and adaptive CCA security.

Note that the above definition (which is based on that of [25][1]) is stronger than that of [4] since they require that the adversary does not make any decryption oracle query with $i \in S^*$ for which $H = H^*$, but we relax the constraint and only require no query with $i \in S^*$ for which $(S, H) = (S^*, H^*)$.

2.2 The BDHE and GBDHE Assumptions

Let us define the sets of polynomials $P = (p_1, \ldots, p_s)$ and $Q = (q_1, \ldots, q_t)$, with $p_1 = q_1 = 1$, and a polynomial f, where $\forall i, k : p_i, q_k, f \in \mathbb{F}_p[X_1, \ldots, X_n]$.

[1] Note that, in comparison with [25], we ignore the *Reg* parameter here as it can be regarded as part of *EK*.

Let $g^P = (g^{p_1}, \dots, g^{p_s})$. We say that f is independent of (P, Q) if it cannot be written as $f = \sum_{i,j=1}^{s} a_{i,j} p_i p_j + \sum_{k=1}^{t} b_k q_k$ for constants $a_{i,j}$ and b_k.

The *generalized decision bilinear Diffie-Hellman exponent (GBDHE)* problem [3] is defined as follows: given the input $g^{P(x_1,\dots,x_n)}$ and $g_{\mathrm{T}}^{Q(x_1,\dots,x_n)}$ for random choices of $x_1, \dots, x_n \in \mathbb{F}_p$, decide between $g_{\mathrm{T}}^{f(x_1,\dots,x_n)}$ and a random $T \in \mathbb{G}_{\mathrm{T}}$. The GBDHE assumption says that it is hard to solve the GBDHE problem if f is independent of (P, Q).

The *decision bilinear Diffie-Hellman exponent* assumption (parameterized by n and denoted by n-BDHE), which is an instance of the GBDHE assumption, says that given the input $g, h, \{g^{\alpha^k}\}_{k \in \{1,\dots,2n\} \setminus \{n+1\}}$ for random $h \in \mathbb{G}$ and $\alpha \in \mathbb{Z}_p$, it is hard to decide between $e(g,h)^{\alpha^{n+1}}$ and a random $T \in \mathbb{G}_{\mathrm{T}}$.

2.3 Universal One-Way Hash Function

Consider a keyed hash function H. H is called a universal one-way hash function (UOWHF) if there is no efficient adversary winning the following security game. First, the adversary chooses a message and outputs it. Then, the challenger chooses a random key for H and gives it to the adversary. Finally, the adversary outputs a second message and terminates. The adversary wins if the two messages are different, but their hashes under the chosen key are the same. This notion was first proposed in [21], and is shown to be strictly weaker than collision resistance [29,26]. In fact, one-way functions are shown to be sufficient for UOWHF [27], whereas collision resistant hash functions are only known to be constructed assuming claw-free permutations [8] or lattice-based assumptions [18].

Note. In the full version of this paper [23], we consider two types of standard model CPA-to-CCA transforms, namely Naor-Yung [22] and Canetti-Halevi-Katz [6] (along with their extensions), and argue that applying these transforms to the proposed broadcast encryption schemes in the literature does not give us CCA security and constant-size ciphertexts.

3 An Efficient Selective CCA Broadcast Encryption

Let $\mathsf{H}_\kappa : \mathbb{G} \mapsto \mathbb{Z}_p$ be a hash family indexed by κ. We define a broadcast encryption scheme OurBE in the following. We describe the system for (at most) $n-1$ users to be notationally consistent with the original scheme of [4], on which the system is based. The system for n users can be defined accordingly.

– $\mathsf{Setup}(1^k, n-1)$ picks a random generator $g \in \mathbb{G}$, two random quantities $\alpha, \gamma \in \mathbb{Z}_p$, and a random index κ for hash function H, computes $v = g^\gamma$, and outputs $MSK = (\alpha, \gamma)$ and $EK = (g, v, \kappa)$.
– $\mathsf{Join}(MSK, i)$ computes $g_k = g^{(\alpha^k)}$ for $k = i, i+1, n+1-i$, and $n+1+i$, and $d_i = g_i^\gamma$, and outputs $sk_i = d_i$ and $pk_i = (g_i, g_{i+1}, g_{n+1-i}, g_{n+1+i})$. The secret key sk_i is given to the user, and EK is updated by appending pk_i.

– $\mathsf{Encaps}(EK, S)$ picks a random $t \in \mathbb{Z}_p$ and sets $K = e(g_{n+1}, g)^t$, which can be computed as $K = e(g_{n+1-i}, g_i)^t$ for any i, computes H as follows, and outputs (H, K).

$$H = \langle g^t, (v \cdot g_1^{\mathsf{H}_\kappa(g^t)} \cdot \prod_{j \in S} g_{n+1-j})^t \rangle.$$

– $\mathsf{Decaps}(EK, sk_i, S, H)$ parses the header as $H = (C_0, C_1)$, checks if the following equation holds:

$$e(C_1, g) = e(v \cdot g_1^{\mathsf{H}_\kappa(C_0)} \cdot \prod_{j \in S} g_{n+1-j}, C_0), \tag{1}$$

and if it does, then calculates the session key as follows:

$$K = \frac{e(C_1, g_i)}{e(d_i \cdot g_{1+i}^{\mathsf{H}_\kappa(C_0)} \cdot \prod_{j \in S \setminus \{i\}} g_{n+1-j+i},\ C_0)}.$$

In the following we bring a theorem which states that if the hash function H is a universal one-way hash function, then the proposed scheme satisfies selective CCA security under the same assumption as that of the original scheme, namely n-BDHE. Intuitively, the main modification we make in (the encryption algorithm of) the original scheme is the introduction of $g_1^{\mathsf{H}_\kappa(g^t)}$. If this element is not present, as it is in the original scheme, given a header $H = (C_0, C_1)$ corresponding to a key K, one can compute the header (C_0^r, C_1^r) that corresponds to the key K^r, and hence the scheme is malleable. We show that a UOWHF is sufficient to eradicate malleability and get CCA security. This modification is inspired by a similar technique in [5] which, in contrast, was shown to be applicable to an *identity-based* scheme. Here we show that a similar idea is applicable to BGW_1. The proof of the following theorem can be found in the full version of this paper [23]. In the proof we use the structure of the keys in the scheme to simulate decryption queries.

Theorem 1. *The above scheme is selective CCA secure if the n-BDHE decision problem is hard and* H *is a universal one-way hash function.*

On Dynamicity. Note that the bound on the number of users in OurBE does not prevent the system from being able to handle more than $n - 1$ users. That is, as long as the system "jumps over" the users number n and $n + 1$ (i.e., after user number $n - 1$, the next user is numbered $n + 2$), the system can handle polynomially many users more than $n - 1$ and remains secure. The security of the scheme with more than $n - 1$ users can be proved based on the following assumption: given the input h, and $\{g_k = g^{\alpha^k}\}$ for $k \in \{n + 1 - m, \ldots, n + 1 + m\} \setminus \{n + 1\}$ for random $g, h \in \mathbb{G}$ and $\alpha \in \mathbb{Z}_p$, it is hard to decide between $e(g_{n+1}, h)$ and a random $T \in \mathbb{G}_T$. It is not hard to see that this assumption is equivalent to the following assumption: given the input g, h, and $\{g_k = g^{\alpha^k}\}$ for

$k \in \{1, \ldots, 2m\} \setminus \{m\}$ for random $h \in \mathbb{G}$ and $\alpha \in \mathbb{Z}_p$, it is hard to decide between $e(g_m, h)$ and a random $T \in \mathbb{G}_\mathrm{T}$. Here $m \geq n+2$ is the last user number to join. This assumption is comparable to the m-BDHE assumption. In fact, like the BDHE assumption, it is an instance of the GBDHE assumption. In view of this observation, OurBE is a *dynamic* broadcast encryption in the sense that: (1) the system setup and the ciphertext size are independent of the upper bound on the number of users; (2) a new user can join anytime without incurring modification of other user secret keys; and (3) the encryption key is incrementally updated by an operation of O(1) complexity.

Comparison. The only broadcast encryption scheme in the literature that provides CCA security with constant-size ciphertexts is BGW_2. It has similar secret and public key sizes as our scheme. However, OurBE has a shorter ciphertext size and is based on universal one-way hash functions which are generally more efficient that signatures on which $\mathsf{BGW}_2$is based. For a more comprehensive comparison see the full version of this paper [23].

4 Inclusive-Exclusive Broadcast Encryption

In this section we show that OurBE can be slightly modified to provide both the broadcast encryption and the revocation functionality simultaneously; that is, we propose a scheme in which the encrypter may choose to determine either a target set S or a revoked set R of users at the time of encryption, without the need to set up two parallel systems. The decryption naturally goes ahead only if the user is either in S or not in R. In the following we (ab)use the notation "S/R" to indicate "either S or R" as input to the encapsulation and decapsulation algorithms. In practice this can be implemented using the first bit of the input to indicate the inclusive or exclusive mode of operation.

- $\mathsf{Setup}(1^k, n-1)$ picks random $g \in \mathbb{G}$, $\alpha, \gamma \in \mathbb{Z}_p$, and κ for H, computes $v = g^\gamma$, sets $\pi_0 = g^{\alpha(\alpha^n - 1)/(\alpha - 1)}$, and outputs $MSK = (\alpha, \gamma)$ and $EK = (g, v, \pi_0, \kappa)$.
- $\mathsf{Join}(MSK, i)$ computes g_i, g_{i+1}, g_{n+1-i}, g_{n+1+i}, and $d_i = g_i^\gamma$, sets $\pi_i = \pi_0^{\alpha^i}/g_{n+1}$, and outputs $sk_i = d_i$ and $pk_i = (g_i, g_{i+1}, g_{n+1-i}, g_{n+1+i}, \pi_i)$. Now, sk_i is given to the user, and EK is updated by appending pk_i.
- $\mathsf{Encaps}(EK, S/R)$ picks a random $t \in \mathbb{Z}_p$ and sets $K = e(g_{n+1}, g)^t$, computes H as either of the following accordingly, and outputs (H, K).

$$H = \langle g^t, (v \cdot g_1^{\mathsf{H}_\kappa(g^t)} \cdot \prod_{j \in S} g_{n+1-j})^t \rangle \text{ or } \langle g^t, (v \cdot g_1^{\mathsf{H}_\kappa(g^t)} \cdot \pi_0 / \prod_{j \in R} g_{n+1-j})^t \rangle$$

- $\mathsf{Decaps}(EK, sk_i, S/R, H)$ parses $H = (C_0, C_1)$, checks if the either of the following equation accordingly holds:

$$e(C_1, g) = e(v \cdot g_1^{\mathsf{H}_\kappa(C_0)} \cdot \prod_{j \in S} g_{n+1-j}, C_0) \text{ or } e(v \cdot g_1^{\mathsf{H}_\kappa(C_0)} \cdot \pi_0 / \prod_{j \in R} g_{n+1-j}, C_0),$$

and if it does, then calculates the session key accordingly as follows:

$$K = \frac{e(C_1, g_i)}{e(d_i \cdot g_{1+i}^{\mathsf{H}_\kappa(C_0)} \cdot \prod_{j \in S \setminus \{i\}} g_{n+1-j+i},\ C_0)}, \quad \text{or}$$

$$K = \frac{e(C_1, g_i)}{e(d_i \cdot g_{1+i}^{\mathsf{H}_\kappa(C_0)} \cdot \pi_i / \prod_{j \in R} g_{n+1-j+i},\ C_0)}.$$

Correctness. If $i \notin R$, the session key the user i calculates in the exclusive mode is effectively the same as the session key it would have calculated if it were decrypting a ciphertext encrypted to $S = N \setminus R$ in the inclusive mode, and therefore the scheme is correct.

Note that the parameters are set in a way that the scheme properly excludes users that join after the time of encryption from inclusive-mode ciphertexts, and includes such users in the exclusive-mode ciphertexts.

Efficiency. The proposed scheme enjoys similar desirable efficiency properties as the inclusive-only scheme; that is, the ciphertext and the user secret key sizes are both constant and the public key size is linear in the number of users.

Security. A similar security definition to that of broadcast encryption can be defined for such schemes, with the difference that the adversary is now allowed to ask decryption oracle queries for both modes. Naturally exclusive-mode decryption oracle queries $\mathcal{Dec}(i^*, N \setminus S^*, H^*)$ for $i^* \in S^*$ are also not allowed. It is not hard to see that the security of OurBE translates into the above scheme satisfying this security definition.

5 Achieving Adaptive CCA Security

Since we have a very efficient scheme with asymptotically optimal size secret keys and ciphertexts which is already proved selective CCA secure based on standard assumptions, in this section we try to see how further we can achieve in terms of security by considering reasonable generalizations of some standard assumptions, while retaining the same optimally efficient secret key and ciphertext sizes. We first propose reasonable generalizations of the GBDHE and prove that they hold in the generic group model; then we prove that OurBE can be proved ACCA secure under these assumptions; and finally we compare our scheme to existing adaptive or CCA secure broadcast encryptions.

5.1 The OBDHE Assumption

We consider extending the GBDHE problem assuming that an extra resource is also given: the *Diffie-Hellman computation oracle* $\mathcal{O}_{g,e}^{\mathrm{DH}}$, that takes two inputs $u, v \in \mathbb{G}$ and outputs $w \in \mathbb{G}$ such that $e(u, v) = e(g, w)$. Formally, we define:

The OBDHE Problem: Given the input $g^{P(x_1,\ldots,x_n)}$ and $g_{\mathrm{T}}^{Q(x_1,\ldots,x_n)}$ for random choices of $x_1,\ldots,x_n \in \mathbb{F}_p$, and access to the $\mathcal{O}_{g,e}^{\mathrm{DH}}$ oracle, decide between $g_{\mathrm{T}}^{f(x_1,\ldots,x_n)}$ and a random $T \in \mathbb{G}_{\mathrm{T}}$.

Note that the GBDHE assumption implies that the only elements (dependent on $x_1,\ldots,x_n$ and) in $\mathbb{G}$ that can be computed are those in the form $g^{\sum a_i p_i}$. Also note that if we assume $u = g^{\sigma_u}$ and $v = g^{\sigma_v}$, we will have $w = \mathcal{O}_{g,e}^{\mathrm{DH}}(u,v) = g^{\sigma_u \sigma_v}$. Hence, by providing access to $\mathcal{O}_{g,e}^{\mathrm{DH}}$, basically a number of "free multiplications" in the exponent are given. Let us define $p' = \sigma_u \sigma_v$. If we consider q' queries to $\mathcal{O}_{g,e}^{\mathrm{DH}}$, and the output to the i-th query represented as $w_i = g^{p'_i}$, we can define $P' = (p'_1,\ldots,p'_{q'})$. Our extension of the GBDHE assumption says that it is still hard to solve the GBDHE problem if these "free multiplications" in the exponent do not help breaking the independence property. Formally, letting $\|$ denote concatenation, we define:

Assumption 1 (OBDHE). *It is hard to solve the decision (P,Q,f)-OBDHE problem if f is independent of $(P \parallel P', Q)$.*

In the full version of this paper [23] we prove that the assumption holds in the generic group model [28,3]. We prove an upper bound on the success of any generic algorithm trying to solve the OBDHE problem which is negligible if p, the order of $\mathbb{F}_p$ is super-polynomial. In fact, our proof is very similar to that of [3], suggesting that our assumption is a natural extension of GBDHE. Note that OBDHE is falsifiable by simply solving the corresponding $(P \parallel P', Q, f)$-GBDHE problem efficiently.

5.2 The GKEA Assumption

We propose the generalized knowledge of exponent assumption (GKEA) as follows and prove that it holds in the generic group model.

In the following we use p to denote a polynomial (suppressing the random variables) and $p(x_1,\ldots,x_n)$ to denote the evaluation of p on the input $(x_1,\ldots,x_n)$. Let $P = (p_1,\ldots,p_s) \in \mathbb{F}_p[X_1,\ldots,X_n]^s$. Let the linear span of P, denoted by $\mathrm{Span}(P)$, be defined as the vector space containing all the polynomials in the form $\sum_{k=1}^{s} a_k p_k$.

Assumption 2 (GKEA). *Let $P = (p_1,\ldots,p_s) \in \mathbb{F}_p[X_1,\ldots,X_n]^s$, where $p_1 = 1$. Let A be an algorithm that given $g^{P(x_1,\ldots,x_n)}$ for a random $(x_1,\ldots,x_n)$, outputs*

$$(\ (a_k)_{k=1}^{s},\ h,\ h^{q(x_1,\ldots,x_n)}\ ), \qquad \textit{such that} \quad q(x_1,\ldots,x_n) = \sum_{k=1}^{s} a_k p_k(x_1,\ldots,x_n).$$

Consider the subspace of $\mathrm{Span}(P)$ defined as $V_q = \{r \mid r, rq \in \mathrm{Span}(P)\}$ and let $\{r_i\}_{i=1}^{t}$ be a basis for V_q. Then, there exists an extractor E that given the same input as A outputs

$$(b_i)_{i=1}^{t}, \qquad \textit{such that} \quad \mathrm{dlog}_g(h) = \sum_{i=1}^{t} b_i r_i(x_1,\ldots,x_n).$$

This assumption basically says that the only way an adversary can produce pairs of the form (h, h^q) is to pick given pairs of the form (h_i, h_i^q) and output $(\prod h_i^{b_i}, \prod (h_i^q)^{b_i})$ for some known values of b_i.

The original KEA of [9] and the KEA3 of [2] are both instances of GKEA [23]. These two instances have already been proved to hold in the generic group model [12,1,13]. We propose a theorem stating the generic assumption and prove it in the full version of this paper [23].

Theorem 2. *The GKEA Assumption holds in the generic group model.*

5.3 Adaptive CCA Security

In this section we prove OurBE adaptive CCA secure under our generalized versions of the BDHE and knowledge of exponent assumptions. To prove adaptive CCA security, we basically show that a decryption query by the adversary that contains a valid ciphertext does not increase the (cryptographic) 'knowledge' of the adversary. Also note that since ciphertext validity is publicly verifiable, a decryption query that contains an invalid ciphertext does not increase the adversary's knowledge either. Hence we basically show that a CCA attack against the system is equivalent to a CPA attack, under the GKEA assumption and the hash function being a UOWHF. Furthermore, the access to $\mathcal{O}_{g,e}^{\mathrm{DH}}$ enables answering adaptive corruption queries.

Formally, we prove adaptive CCA security assuming that the OBDHE and the GKEA assumptions hold and H is a UOWHF. Intuitively, selective CPA security stems from the BDHE assumption underlying the OBDHE assumption along with the hash function being a UOWHF; the Diffie-Hellman oracle enables adaptive security; and the CCA security is achieved from the GKEA assumption along with the hash function being a UOWHF. The following theorem is proved in the full version of this paper [23].

Theorem 3. OurBE *is adaptive CCA secure if the OBDHE and the GKEA assumptions hold and* H *is a universal one-way hash function.*

We note that we prove CCA security based on the GKEA assumption, an assumption which is much weaker than the generic model itself (and instances of it are shown to be falsifiable [2]), and in fact, proving the equivalence of CPA and CCA security is trivial if the generic group model is used directly, since on a decryption query with a first element g^t, we may assume that t is known.

5.4 Comparison

Since our scheme is the first to achieve adaptive CCA security with constant-size ciphertexts, we compare our scheme with those from the literature that are adaptive CPA or selective CCA secure. We do not consider schemes that are not fully collusion resistant. The schemes in the literature with constant-size ciphertexts include a selective CCA secure scheme from [4], and three adaptive CPA secure schemes from [17] and [30]. The schemes in the literature that do not have constant-size ciphertexts include adaptive CPA secure schemes from [14], [17] (identity-based) and [19] (revocation scheme), and recent adaptive CCA secure schemes from [24]. Table 1 summarizes our comparison. We list plain and identity-based (IB) broadcast encryption (BE) and revocation (R) schemes. Among these, schemes from [14] and [24] are generic schemes based on (hierarchical) identity-based encryption ((H)IBE), and encryption schemes (implemented under DDH), respectively. Since (H)IBE can be based on various assumption, we simply use it in parentheses in the table. All other schemes are explicit proposals based on various bilinear Diffie-Hellman assumptions, sometimes plus extra assumptions such as strong unforgeability (SUF), pseudo-random functions (PRF), and the random oracle model (ROM). To accommodate more information, we omit the O notation and write $O(f(n, s, r))$ as $f(n, s, r)$.

Table 1. Comparison of adaptive or CCA secure broadcast encryption schemes

Scheme		$O(\|sk_i\|)$	$O(\|H\|)$	Security	Assumption
[14]	BE	$\log n$	$r \log \frac{n}{r}$	ACCA1	(IBE)
	BE	$\log^{1+\epsilon} n$	$\frac{r}{\epsilon}$	ACCA1	(HIBE)
[4]	BE	**1**	**1**	SCCA	n-BDHE, SUF
[17]	BE	**1**	**1**	ACPA	n-BDHES, PRF, ROM
	BE	**1**	s	ACPA	n-BDHES, PRF
	IBBE	**1**	**1**	ACPA	n-BDHES, PRF, ROM
	IBBE	**1**	$\sqrt{s}$	ACPA	n-BDHES, PRF
[30]	BE	n	**1**	ACPA	dBDH, dLin
[19]	R	**1**	r	ACPA	dBDH, dLin
[24]	BE	**1**	$r \log \frac{n}{r}$	**ACCA**	DDH
	BE	**1**	r	**ACCA**	DDH
OurBE	BE	**1**	**1**	SCCA	n-BDHE, UOWHF
				ACCA	n-OBDHE, GKEA, UOWHF

$O(|\cdot|)$: order of size, n, s, r: number of total, targeted, revoked users.

6 Concluding Remarks

We proposed a very efficient broadcast encryption scheme. The sizes of the secret keys and ciphertexts in the scheme are asymptotically optimal, i.e., constant. We showed that the scheme can be proved selective CCA secure assuming BDHE and a universal one-way hash function. Furthermore, we showed that proving adaptive CCA security is possible if we consider extended versions of the GB-DHE and knowledge of exponent assumptions. Considering only the standard assumptions, our scheme provides shorter ciphertexts than the only other known CCA secure scheme. Considering the extended assumptions, our scheme is the first scheme to achieve constant size secret keys and ciphertexts and adaptive CCA security at the same time. The problem of designing schemes that achieve such properties under standard assumptions remains open.

Acknowledgments. This work was supported by the French ANR-09-VERS-016 BEST Project. The authors would like to thank the anonymous reviewers of ACISP 2012.

References

1. Abe, M., Fehr, S.: Perfect NIZK with Adaptive Soundness. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 118–136. Springer, Heidelberg (2007)
2. Bellare, M., Palacio, A.: The Knowledge-of-Exponent Assumptions and 3-Round Zero-Knowledge Protocols. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 273–289. Springer, Heidelberg (2004)
3. Boneh, D., Boyen, X., Goh, E.-J.: Hierarchical Identity Based Encryption with Constant Size Ciphertext. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Heidelberg (2005)
4. Boneh, D., Gentry, C., Waters, B.: Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 258–275. Springer, Heidelberg (2005)
5. Boyen, X., Mei, Q., Waters, B.: Direct chosen ciphertext security from identity-based techniques. In: Atluri, V., Meadows, C., Juels, A. (eds.) ACM CCS 2005 Conference on Computer and Communications Security, pp. 320–329. ACM Press (November 2005)
6. Canetti, R., Halevi, S., Katz, J.: Chosen-Ciphertext Security from Identity-Based Encryption. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 207–222. Springer, Heidelberg (2004)
7. Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM Journal on Computing 33(1), 167–226 (2003)
8. Damgård, I.: Collision Free Hash Functions and Public Key Signature Schemes. In: Price, W.L., Chaum, D. (eds.) EUROCRYPT 1987. LNCS, vol. 304, pp. 203–216. Springer, Heidelberg (1988)
9. Damgård, I.: Towards Practical Public Key Systems Secure against Chosen Ciphertext Attacks. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 445–456. Springer, Heidelberg (1992)
10. Delerablée, C.: Identity-Based Broadcast Encryption with Constant Size Ciphertexts and Private Keys. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 200–215. Springer, Heidelberg (2007)
11. Delerablée, C., Paillier, P., Pointcheval, D.: Fully Collusion Secure Dynamic Broadcast Encryption with Constant-Size Ciphertexts or Decryption Keys. In: Takagi, T., Okamoto, T., Okamoto, E., Okamoto, T. (eds.) Pairing 2007. LNCS, vol. 4575, pp. 39–59. Springer, Heidelberg (2007)
12. Dent, A.W.: The hardness of the dhk problem in the generic group model. Cryptology ePrint Archive, Report 2006/156 (2006), http://eprint.iacr.org/2006/156
13. Desmedt, Y.G., Phan, D.H.: A CCA Secure Hybrid Damgård's ElGamal Encryption. In: Baek, J., Bao, F., Chen, K., Lai, X. (eds.) ProvSec 2008. LNCS, vol. 5324, pp. 68–82. Springer, Heidelberg (2008)

14. Dodis, Y., Fazio, N.: Public Key Broadcast Encryption for Stateless Receivers. In: Feigenbaum, J. (ed.) DRM 2002. LNCS, vol. 2696, pp. 61–80. Springer, Heidelberg (2003)
15. Dodis, Y., Fazio, N.: Public Key Trace and Revoke Scheme Secure against Adaptive Chosen Ciphertext Attack. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 100–115. Springer, Heidelberg (2002)
16. Fiat, A., Naor, M.: Broadcast Encryption. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 480–491. Springer, Heidelberg (1994)
17. Gentry, C., Waters, B.: Adaptive Security in Broadcast Encryption Systems (with Short Ciphertexts). In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 171–188. Springer, Heidelberg (2009)
18. Goldreich, O., Goldwasser, S., Halevi, S.: Collision-free hashing from lattice problems. Cryptology ePrint Archive, Report 1996/009 (1996), http://eprint.iacr.org/1996/009
19. Lewko, A.B., Sahai, A., Waters, B.: Revocation systems with very small private keys. In: 2010 IEEE Symposium on Security and Privacy, pp. 273–285. IEEE Computer Society Press (May 2010)
20. Naor, D., Naor, M., Lotspiech, J.: Revocation and Tracing Schemes for Stateless Receivers. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 41–62. Springer, Heidelberg (2001)
21. Naor, M., Yung, M.: Universal one-way hash functions and their cryptographic applications. In: 21st ACM STOC, pp. 33–43. ACM Press (May 1989)
22. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: 22nd ACM STOC. ACM Press (May 1990)
23. Phan, D.-H., Pointcheval, D., Shahandashti, S.F., Strefler, M.: Adaptive cca broadcast encryption with constant-size secret keys and ciphertexts. Cryptology ePrint Archive, Report 2012/216 (2012), http://eprint.iacr.org/2012/216
24. Phan, D.-H., Pointcheval, D., Strefler, M.: Adaptively secure broadcast encryption with forward secrecy. Cryptology ePrint Archive, Report 2011/463 (2011), http://eprint.iacr.org/2011/463
25. Phan, D.H., Pointcheval, D., Strefler, M.: Security Notions for Broadcast Encryption. In: Lopez, J., Tsudik, G. (eds.) ACNS 2011. LNCS, vol. 6715, pp. 377–394. Springer, Heidelberg (2011)
26. Rogaway, P., Shrimpton, T.: Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. In: Roy, B., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 371–388. Springer, Heidelberg (2004)
27. Rompel, J.: One-way functions are necessary and sufficient for secure signatures. In: 22nd ACM STOC, pp. 387–394. ACM Press (May 1990)
28. Shoup, V.: Lower Bounds for Discrete Logarithms and Related Problems. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 256–266. Springer, Heidelberg (1997)
29. Simon, D.R.: Findings Collisions on a One-Way Street: Can Secure Hash Functions Be Based on General Assumptions? In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 334–345. Springer, Heidelberg (1998)
30. Waters, B.: Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009)

A Generic Construction of Accountable Decryption and Its Applications

Xuhua Zhou[1], Xuhua Ding[2], and Kefei Chen[1,3]

[1] Shanghai Jiao Tong University, China
xuhuazhou2010@gmail.com, kfchen@sjtu.edu.cn
[2] Singapore Management University, Singapore
xhding@smu.edu.sg
[3] Shanghai Key Laboratory of Scalable Computing and Systems, Shanghai, China

Abstract. We propose a new cryptographic notion called *accountable decryption* by which, given a ciphertext, a decryptor proves both the correctness of his decryption and the plaintext authenticity to a public verifier. We define its security from three aspects: message confidentiality, soundness of verifiability and plaintext authenticity. Given any asymmetric or symmetric key encryption scheme, we propose a method to construct the corresponding accountable decryption scheme with provable security. To demonstrate its applications, we also present the constructions for predicate encryption and for public-key encryption with keyword search.

Keywords: Accountable Decryption, Verifiable Decryption, Plaintext Authenticity.

1 Introduction

Recent advances in encryption schemes with fine-grained controls, such as PEKS [1], attribute-based encryption [2] and predicate encryption [3], offer a great flexibility for attribute-based access control, where the decryptor can access a protected object, e.g., an encrypted file, only if she satisfies certain access control rules. Such an access control paradigm can be further extended to decryption capability based authorization, in the sense that the decryption capability is bound with access control policies. For instance, an authorization server issues a ciphertext token such that only authorized users can decipher it successfully. A resource server (e.g., a database server) grants a user's accesses based on whether a user can decipher the token correctly. Scalability and flexibility are the main advantages of such an authorization system as compared with identity-based authorization such as Kerberos [4].

Nonetheless, it is not a trivial problem for the resource server to determine whether a user can decrypt an authorization ciphertext token correctly when the server is not equipped with the decryption key in many applications. Therefore, the user has to convince the server on two aspects: *the correctness of decryption* and *the authenticity of the plaintext*, by constructing publicly verifiable proofs. The former issue can be addressed by the verifiable decryption scheme defined by Camenisch and Shoup in [5], where a verifier requests a decryption key holder to prove that the decryption of ψ is m or satisfies other properties. Unfortunately, their construction only works with a specific public key encryption scheme in [5]. It is unclear whether their technique is applicable to other encryption schemes, especially to the family of attributed encryption schemes. The second issue can be easily addressed by using signatures. Nonetheless, no study has been made on integrating the signatures with verifiable encryption in an *efficient* manner.

In this paper, we propose *accountable decryption* as a new cryptographic notion to enable the decryption capability based authorization by addressing both issues securely and efficiently. In a high level view, we consider the problem that a ciphertext receiver (a user requesting resources) intends to convince a public verifier (the resource server) that the ciphertext ψ she receives from a sender (the authorization server) decrypts to a message m. As a result, the public verifier ascertains both the authenticity of m and the decryption of ψ. Note that signcryption schemes [6,7,8,9,10,11] do *not* solve the problem because the decryptor in those schemes cannot convince a public verifier that his decryption is correct.

Accountable decryption can be regarded as a combination of verifiable decryption of matching plaintext [5] and message authenticity. Different from [5], we are not interested in zero-knowledge proofs that the decryption of ψ has a particular structure because it is not the concern in the target setting. We are interested in using accountable decryption as a stand-alone scheme to prevent a dishonest decryptor from making fraudulent claims about his decryption capability. Besides the aforementioned decryption-capability based authorization, accountable decryption can be applied in other applications, such as a web cache server's proof that an HTML file is the plaintext of an encrypted update from the web server and a proxy server's proof to a user that an encrypted file from a data publisher matches her interests. Moreover, the proof the relationship between a plaintext and a ciphertext is the main point differing accountable encryption from signcryption.

The main contributions of our work include the formulation of the notion of accountable decryption and its security properties. We propose a generic method to construct an accountable decryption scheme from any encryption scheme, including public key encryption and symmetric key ciphers. We formally prove the security of the generic construction and then instantiate it for predicate encryption and PEKS.

Organization. The rest of this paper is organized as follows. We discuss related work in Section 2. Section 3 defines the concept of accountable decryption and its three security requirements. Our proposed generic construction and its extension to functional encryption are described in Section 4. Section 5 shows two specific accountable decryption schemes for predicate encryption and public-key encryption with keyword search. We conclude this paper in Section 6.

2 Related Work

Camenisch and Shoup in [5] proposed the concept of verifiable decryption with an emphasis on the binary relation between the plaintext and a witness, e.g. the discrete logarithm relation. Its main motivation is for cryptographic protocols such as key escrow and optimistic fair exchange, where the decryptor, usually a trusted party, proves to a verifier that a ciphertext is decrypted correctly.

Verifiable decryption is related to verifiable encryption, which is also proposed in [5]. In a verifiable encryption scheme, the encryptor, instead of the decryptor, proves that the ciphertext is an encryption satisfying certain property. This type of schemes are widely used in many schemes such as group signatures and confirmer signatures. In a recent work [12], Barbosa and Farshim defined the concept of verifiable function encryption, which is built on a standard functional encryption and a special MAC scheme, called *n-key-chameleon MAC*.

Another related concept is verifiably encrypted signatures (VES), which was introduced by Boneh et al. in [13]. In their scenario, a signer encrypts its signature under a trusted third party (PPT, called the adjudicator), and then appends a proof about its content. This proof is used to confirm that the signer has truly signed a certain object. A popular application of VES is online contract signing, which is a type of optimistic fair exchange protocol. Rückert and Schröder [14] suggested extractability and abuse-freeness as two fundamental properties for VES. In [15], Rückert constructed a VES only for RSA signatures without pairings and non-interactive zero-knowledge proofs, while [13,14] are based on paring and [16] are based on NIZKs. Rückert, Schneider and Schröder [17] sketched a more generic construction without pairing and NIZKs.

The notion of signcryption was introduced by Zheng [18] in order to address message integrity and confidentiality simultaneously. In a signcryption scheme, encryption and signature are done simultaneously with a much lower computational cost and communication overhead than the Sign-then-Encrypt approach. Nonetheless, confidentiality offered by signcryption schemes does not necessarily lead to verifiability required in our target applications. We refer readers to [19] which gives a good account of some previous work on signcryption.

3 Accountable Decryption

3.1 Scheme Definition

We consider a system setting of three types of participants: a data sender denoted by $\mathcal{S}$, a ciphertext receiver denoted by $\mathcal{P}$, and a verifier denoted by $\mathcal{V}$. $\mathcal{S}$ sends out encrypted data to $\mathcal{P}$, who can decipher it to get a plaintext message. In addition, $\mathcal{P}$ can produce a proof to $\mathcal{V}$ that the plaintext message is the result of decrypting the ciphertext sent by $\mathcal{S}$. Different from the verifiable decryption in [5], we do not require $\mathcal{P}$ to present a zero knowledge proof about the relationship between the plaintext and a witness. In addition, we do not restrict the underlying encryption scheme to be a public key encryption. $\mathcal{S}$ and $\mathcal{P}$ can share a key and use a secure symmetric key cipher. A scheme of accountable decryption is formally defined below.

Definition 1 (Accountable Decryption). *An accountable decryption scheme is a special encryption scheme consisting of the following four probabilistic polynomial time (PPT) algorithms.*

Setup(1^λ) *is an initialization algorithm which takes as input a security parameter* 1^λ*, and outputs two key-pairs: an encryption/decryption key-pair* $(EK_\mathcal{P}, DK_\mathcal{P})$ *for the receiver* $\mathcal{P}$*, and a signing/verification key-pair* $(SK_\mathcal{S}, VK_\mathcal{S})$ *for the sender* $\mathcal{S}$*. If the encryption scheme is a public key encryption,* $\mathcal{P}$*'s encryption key* $EK_\mathcal{P}$ *is public; otherwise* $EK_\mathcal{P}$*, which is the same as* $DK_\mathcal{P}$*, is assigned to* $\mathcal{S}$*.*

Encrypt$(EK_\mathcal{P}, SK_\mathcal{S}, M)$ *takes as input* $\mathcal{P}$*'s encryption key* $EK_\mathcal{P}$*,* $\mathcal{S}$*'s signing key* $SK_\mathcal{S}$ *as well as a plaintext message* M *and outputs a ciphertext* CT*.*

Decrypt$(DK_\mathcal{P}, VK_\mathcal{S}, CT)$ *takes as input* $\mathcal{P}$*'s decryption key* $DK_\mathcal{P}$*,* $\mathcal{S}$*'s verification key* $VK_\mathcal{S}$*, as well as a ciphertext* CT *and outputs a tuple* (M, σ) *where* M *is the plaintext message and* σ *is a tag if the decryption succeeds, or outputs* $\perp$ *otherwise.*

Verify$(VK_\mathcal{S}, CT, M, \sigma)$ *takes as input* $\mathcal{S}$*'s verification key* $VK_\mathcal{S}$*, a ciphertext* CT*, a plaintext* M *and a tag* σ*. It outputs* 1 *if* M *and* σ *are the outputs of* Decrypt$(DK_\mathcal{P}, VK_\mathcal{S}, CT)$*; otherwise, outputs 0.*

Correctness. These algorithms must satisfy the following consistency constraint. For any M *in the plaintext space, the below two equations hold,*

$$(M, \sigma) = \mathit{Decrypt}(DK_\mathcal{P}, VK_\mathcal{S}, CT), \quad 1 = \mathit{Verify}(VK_\mathcal{S}, CT, M, \sigma)$$

where $CT \leftarrow$ *Encrypt*$(EK_\mathcal{P}, SK_\mathcal{S}, M)$ *and* $(EK_\mathcal{P}, DK_\mathcal{P}), (SK_\mathcal{S}, VK_\mathcal{S}) \leftarrow$ *Setup*(1^λ)*.* □

The definition does not require $\mathcal{S}$'s key pair and $\mathcal{P}$'s key pair to be related. In fact, they are generated independently by $\mathcal{S}$ and $\mathcal{P}$ respectively, as in a stardard public key signature/encryption set up. Nonetheless, the encryption function Encrypt$(EK_\mathcal{P}, SK_\mathcal{S}, M)$ is different from the standard public key encryption algorithm as it takes a private key as input.

Remark. As compared with verifiable decryption [5], accountable decryption has an additional assurance on the genuineness of the origin of the plaintext embedded in a ciphertext. To some extent, accountable decryption has similarity with the signcryption scheme [18] in terms of usage of keys. However, these two notions are remarkably different. The signcryption scheme assures the decryptor on the received plaintext integrity and confidentiality. In the definition of acccountable decryption, the decryptor has to assure a *public verifier* about the plaintext authenticity and its relation to the given ciphertext. Although verifiability is also mentioned in several signcryption schemes [6,7,8,9,10,11], they are only concerned about the plaintext's authenticity, instead of the relationship between a plaintext and a ciphertext.

3.2 Security Definition

The security properties of an accountable decryption scheme consist of three aspects. First of all, it should maintain the security of the underlying encryption scheme. Namely, the privacy of the encrypted plaintext should not be compromised. This property is not discussed in the verifiable decryption definition in [5], because that construction makes no changes on the underlying encryption scheme. However, in our setting, $\mathsf{Encrypt}()$ function is not the standard one, as it involves an extra secret input. Therefore, we must preserve the semantic security. Secondly, as mentioned in [5] as well, the proof from $\mathcal{P}$ should be sound. In other words, a dishonest $\mathcal{P}$ can not derive a proof passing the verification function. $\mathcal{P}$ must perform the expected decryption to obtain a valid decryption proof. Last but not the least, we require the authenticity of the plaintext. Namely, the plaintext originates from $\mathcal{S}$. The authenticity property allows that an accountable decryption scheme can be used to detect message forgery in applications where $\mathcal{P}$ plays the role of an untrusted proxy. $\mathcal{P}$ will be held "accountable" in case of forgery. In the following, we formulate the three security properties by three game models respectively, all of which are played by a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

We refer to the first security property by *message confidentiality*, whose game model is the same as the indistinguishability game [20] used for symmetric or asymmetric key encryption schemes with the only difference on encryption queries and decryption queries. For different types of attack $atk \in \{CPA, CCA, CCA2\}$, we denote the game by $Game_{atk}$ and explain the two queries below without repeating the entire indistinguishability game. $\mathcal{A}$ in the game models is an outsider adversary who attempts to obtain semantic information of the encrypted message. The indistinguishability game is described below.

Setup: $\mathcal{C}$ runs $\mathsf{Setup}(1^\lambda)$ to generate an encryption key pair $EK_\mathcal{P}, DK_\mathcal{P}$ of an encryption scheme Σ, and a signature key pair $SK_\mathcal{S}, VK_\mathcal{S}$ of a signature scheme Γ. $\mathcal{C}$ sends $VK_\mathcal{S}$ to $\mathcal{A}$. If Σ is a public key cipher, $EK_\mathcal{P}$ is sent to $\mathcal{A}$ as well.

Query Phase I. $\mathcal{A}$ issues to $\mathcal{C}$ a polynomial number of queries, where each query is of one of two types:

- On the i-th encryption query, $\mathcal{A}$ sends to $\mathcal{C}$ a message $M^{(i)}$. In response, $\mathcal{C}$ replies with $\mathsf{Encrypt}(EK_\mathcal{P}, SK_\mathcal{S}, M^{(i)})$.
- (In CCA and CCA2 games) On the j-th decryption query, $\mathcal{A}$ sends to $\mathcal{C}$ a ciphertext $CT^{(j)}$. In response, $\mathcal{C}$ replies with $\mathsf{Decrypt}(DK_\mathcal{P}, VK_\mathcal{S}, CT^{(j)})$.

Challenge: $\mathcal{A}$ outputs a pair of plaintexts, M_0^*, M_1^*. Then $\mathcal{C}$ returns $CT^* \leftarrow \mathsf{Encrypt}(EK_\mathcal{P}, SK_\mathcal{S}, M_\beta^*)$, where β is a random bit chosen by $\mathcal{C}$.

Query Phase II. $\mathcal{A}$ continues to issue queries as in the Query Phase I, with the restriction that CT^* should not be submitted as a decryption query. (Here, only in CCA2 game, decryption queries are allowed.)

Guess: At last, $\mathcal{A}$ outputs a guess $\beta' \in \{0, 1\}$ and wins if $\beta = \beta'$.

The advantage of $\mathcal{A}$ is defined as $|\Pr[\beta = \beta'] - \frac{1}{2}|$. We define the notion of message confidentiality based on $\mathcal{A}$'s advantage.

Definition 2 (Message Confidentiality). *An accountable decryption scheme provides message confidentiality under atk attacks if and only if, for all PPT* *adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in winning $Game_{atk}$ is negligible in λ, where $atk \in \{CPA, CCA, CCA2\}$.* □

We refer to the second security property explained above as the *soundness of verifiability*, and formally define it using a game $Game_{\mathcal{S}}$ described below. Note that $\mathcal{A}$ in $Game_{\mathcal{S}}$ models both outsiders and a malicious $\mathcal{P}$, whose goal is to fake a proof passing the verifier's verification function.

Setup: $\mathcal{C}$ runs $\mathsf{Setup}(1^\lambda)$ to generate an encryption key pair $EK_{\mathcal{P}}, DK_{\mathcal{P}}$ of an encryption scheme Σ, and a signature key pair $SK_{\mathcal{S}}, VK_{\mathcal{S}}$ of a signature scheme Γ. It sends $EK_{\mathcal{P}}, DK_{\mathcal{P}}, VK_{\mathcal{S}}$ to $\mathcal{A}$.

Query: $\mathcal{A}$ adaptively issues encryption queries. On the i-th encryption query, $\mathcal{A}$ sends a plaintext $M^{(i)}$ to $\mathcal{C}$ who responds with $\mathsf{Encrypt}(EK_{\mathcal{P}}, SK_{\mathcal{S}}, M^{(i)})$.

Output: After a polynomial number of queries, $\mathcal{A}$ obtains a set of ciphertext denoted by Ω. It outputs a tuple (CT^*, M^*, σ^*) where $CT^* \in \Omega$. The adversary $\mathcal{A}$ wins the game if M^* is not the plaintext encrypted in CT^* and $\mathsf{Verify}(VK_{\mathcal{S}}, CT^*, M^*, \sigma^*) = 1$.

The advantage of $\mathcal{A}$ is defined as $\Pr[(M^*, \sigma^*) \neq \mathsf{Decrypt}(DK_{\mathcal{P}}, VK_{\mathcal{S}}, CT^*) \wedge \mathsf{Verify}(VK_{\mathcal{S}}, CT^*, M^*, \sigma^*) = 1]$.

Definition 3 (Soundness of Verifiability). *An accountable decryption scheme is sound if and only if, for all PPT adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in winning $Game_{\mathcal{S}}$ is negligible in λ.* □

The third security property is referred to as *plaintext authenticity.* We define it using the following game $Game_A$ between $\mathcal{C}$ and $\mathcal{A}$. Similar to the way in $Game_{\mathcal{S}}$, $\mathcal{A}$ models the malicious $\mathcal{P}$ and outsiders.

Setup: the same as the Setup phase in $Game_{\mathcal{S}}$.

Query: the same as the Query phase in $Game_{\mathcal{S}}$.

Output: After issuing a polynomial number of queries, $\mathcal{A}$ outputs CT^*. The adversary $\mathcal{A}$ wins the game if $(M^*, \sigma^*) \neq \perp \wedge M^* \notin \Upsilon$ holds, where $(M^*, \sigma^*) = \mathsf{Decrypt}(DK_{\mathcal{P}}, VK_{\mathcal{S}}, CT^*)$ and Υ is the set of queried plaintexts.

The advantage of $\mathcal{A}$ is defined as $\Pr[(M^*, \sigma^*) = \mathsf{Decrypt}(DK_{\mathcal{P}}, VK_{\mathcal{S}}, CT^*) : M^* \neq \perp \wedge M^* \notin \Upsilon]$.

Definition 4 (Plaintext Authenticity). *An accountable decryption scheme is said to be plaintext authentic if, for all PPT adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in winning $Game_A$ is negligible in λ.* □

The differences between the definitions of soundness and plaintext authenticity are subtle. However, their security implications are different, yet closely related. Soundness emphasizes on the trustworthiness of the matching relationship between a message and a ciphertext. The plaintext authenticity emphasizes on the genuineness of the embedded plaintext's origin, which is a desirable property for end users in our targeted applications. With both soundness and plaintext authenticity, the verifiers can trust $\mathcal{P}$'s decryption results.

4 A Generic Construction

Verifiable decryption in the public key setting could be trivially solved if the decryptor is able to derive and publish all the randomness used in encryption. A verifier can re-encrypt the plaintext using the same randomness which should result in the same ciphertext. Unfortunately, this method is not feasible for all encryptions schemes including the symmetric key ciphers. We are inspired to design a generic accountable decryption construction for all encryption schemes.

The basic idea is that $\mathcal{S}$ encrypts her signature related to the pay-load message M using $\mathcal{P}$'s encryption key $EK_{\mathcal{P}}$, and securely masks M using the signature. If $\mathcal{P}$ performs decryption correctly, she obtains the correct signature which allows her to recover M. Taking the signature as a proof, the verifier $\mathcal{V}$ checks its validity against the ciphertext and M which ensures the correctness of decryption.

Remark. An accountable decryption scheme can be trivially obtained by retrofitting the verifiable decryption [5] with a sender signature. However, it is *not* a generic construction and the technique is not applicable to other encryption schemes, such as symmetric key ciphers and attribute-based encryption.

4.1 Notation

A signature scheme is denoted by Γ, which consists of three algorithms $\Gamma = (KeyGen_{\Gamma}, Sign, Ver)$; an encryption scheme is denoted by Σ, which consists of three algorithms $\Sigma = (KeyGen_{\Sigma}, Enc, Dec)$; and the proposed scheme of accountable decryption is denoted by Π. We suppose that the messages in $\mathcal{MS}_{\Pi}$ have the same length k, i.e. $\mathcal{MS}_{\Pi} \subseteq \{0,1\}^k$, and that the signatures of Γ have the same length m as the plaintexts of Σ, i.e. $\mathcal{SS}_{\Gamma} \subseteq \mathcal{MS}_{\Sigma} \subseteq \{0,1\}^m$. Also, we restrict $m \geq k$ because of Theorem 2.5.6 in [21].

4.2 Proposed Construction

Before detailing our construction, we introduce the Goldreich-Levin (GL) theorem [21] and the hardcore function briefly. The GL theorem (Theorem 2.5.2 in [21]) says that if g be defined by $g(x,r) = (f(x), r)$, where f is an arbitrary strong one-way function and $|x| = |r|$, the inner product mod 2 of the binary vectors x and r, denoted by $b(x,r)$, is a hard-core of the function g; and Theorem 2.5.6 in [21] extends the notion of hard-core from a bit to a bit string (whose length is $|x|$ at most). According their definition, the bit string produced by the hardcore function is indistinguishable from a random bit string chosen from the proper domain uniformly. In our construction below, we will construct a hardcore function and use its bit string to mask the message to encrypt.

Given a secure signature scheme Γ and a secure encryption scheme Σ satisfying the requirements described in Section 4.1, we construct an accountable decryption scheme Π consisting of four algorithms Setup, Encrypt, Decrypt and Verify as follows:

Setup(1^{λ})**:** This algorithm runs $KeyGen_{\Sigma}(1^{\lambda})$ to generate an en/decryption key-pair $EK_{\mathcal{P}}$ and $DK_{\mathcal{P}}$ of the encryption scheme Σ for $\mathcal{P}$and also runs

$KeyGen_\Gamma(1^\lambda)$ to generate the signing/verification key-pair $(SK_\mathcal{S}, VK_\mathcal{S})$ of the signature scheme Γ. As stated in Definition 1, $\mathcal{S}$ holds its private signature key $SK_\mathcal{S}$ and publishes $VK_\mathcal{S}$. $\mathcal{P}$ holds a secret decryption key $DK_\mathcal{P}$. If Σ is a public key encryption, $EK_\mathcal{P}$ is public too; otherwise $EK_\mathcal{P}$ is sent to $\mathcal{S}$ securely as a shared encryption key.

Finally, using the GL theorem, the algorithm outputs a hardcore function $B()$ of k bits outputs, w.r.t. the one-way function $Enc_{EK_\mathcal{P}}(\sigma)$, where $\sigma \in \mathcal{MS}_\Sigma$.[1] Specifically, $B()$ is defined as $B : \mathcal{MS}_\Sigma \times \{0,1\}^{m+k-1} \to \{0,1\}^k$.

Encrypt$(EK_\mathcal{P}, SK_\mathcal{S}, M)$**:** To encrypt a message $M \in \mathcal{MS}_\Pi$, $\mathcal{S}$ firstly chooses a random $R \in_R \{0,1\}^{m+k-1}$, and generates a signature as $\sigma \leftarrow Sign_{SK_\mathcal{S}}(M||R)$. Then she encrypts σ as $E \leftarrow Enc_{EK_\mathcal{P}}(\sigma)$ and computes the hardcore bits $K = B(\sigma, R)$. Finally, she sets $C = M \oplus K$. The ciphertext of M is a three-tuple $CT = (R, E, C)$.

Decrypt$(DK_\mathcal{P}, VK_\mathcal{S}, CT)$**:** To decrypt $CT = (R, E, C)$, $\mathcal{P}$ firstly computes $\sigma \leftarrow Dec_{DK_\mathcal{P}}(E)$, then computes $K = B(\sigma, R)$, and $M = C \oplus K$. If the output of $Ver_{VK_\mathcal{S}}(M||R, \sigma)$ is 1, return (M, σ); otherwise, return $\perp$.

Verify$(VK_\mathcal{S}, CT, M, \sigma)$**:** Let $CT = (R, E, C)$. $\mathcal{V}$ computes $K = B(\sigma, R)$ and $M' = C \oplus K$. If $Ver_{VK_\mathcal{S}}(M||R, \sigma) = 1 \wedge M = M'$, return 1; otherwise, return 0.

Our construction does not require Γ and Σ to be related. They can be computationally independent of each other. We first prove that if Σ is IND-CCA2, then the generic construction above is also IND-CCA2.

Theorem 1 (IND-CCA2 Security). *If the signature scheme Γ is secure in the sense of UF-CMA, and the encryption scheme Σ is secure in the sense of IND-CCA2, then the proposed scheme Π of accountable decryption has message confidentiality under IND-CCA2 attacks.*

Due to the page limitation, we only give out the sketch of the proof and some key points. Supposing an adversary $\mathcal{A}$ wins $Game^{\Pi}_{CCA2}$ with non-negligible probability, i.e. breaks the proposed scheme Π, we could construct a simulator $\mathcal{B}$ to win $Game^{\Sigma}_{CCA2}$ with non-negligible probability. With the help of the challenger $\mathcal{C}$ in $Game^{\Sigma}_{CCA2}$, $\mathcal{B}$ can answer $\mathcal{A}$'s en/decryption queries.

In the Challenge phase, the simulator $\mathcal{B}$ constructs the challenge ciphertext $CT^* = (R^*, E^*, C^*)$. R^* is a random chosen from the proper domain by $\mathcal{B}$; E^* is a ciphertext of σ^*_b, which is a sigmature of either $M^*_0\|R^*$ or $M^*_1\|R^*$; $C^* = M^*_\beta \oplus B(\sigma^*_\beta, R^*)$. There are two random bits b and β chosen by $\mathcal{C}$ and $\mathcal{B}$ respectively.

We denote $\mathcal{A}$'s guess in $Game^{\Pi}_{CCA2}$ by β', $\mathcal{B}$'s guess in $Game^{\Sigma}_{CCA2}$ by b'. If $b = b'$, i.e. $\mathcal{B}$ wins $Game^{\Sigma}_{CCA2}$, there are two cases for this event: 1) $b = \beta$; 2) $b \neq \beta$. The former indicates $\mathcal{A}$ is challenged with a valid ciphertext. If $\mathcal{A}$ can win $Game^{\Pi}_{CCA2}$ with a non-negligible probability ϵ, $\mathcal{B}$ also can win $Game^{\Sigma}_{CCA2}$ with

[1] Let $B()$'s inputs be σ, R. As described in [21], the output binary string is $b_1(\sigma, R) \cdots b_k(\sigma, R)$, where $b_i(\sigma, R)$ denotes the inner product mod 2 of the binary strings σ and $(R_i, \ldots, R_{m+i-1})$, where $R = (R_1, \ldots, R_{m+k-1})$.

ϵ. That is, $\Pr[b = b'|b = \beta] = 1/2 + \epsilon$. The latter indicates $\mathcal{A}$ is challenged with an invalid ciphertext. The second part E^* has no information about M^*_β and the third part C^* can be treated as a random string. Thus, the probability that $\mathcal{A}$ wins $Game^{\Pi}_{CCA2}$ in this case is $1/2$, so does $\mathcal{B}$. That is, $\Pr[b = b'|b \neq \beta] = 1/2$.

Both the above cases have the same probability, i.e. $1/2$, to occur. The advantage of $\mathcal{B}$ is $\epsilon/2$. Since Σ is IND-CCA2 secure, $\epsilon/2$ is negligible, so is ϵ.

It's worth to mention that in Query Phase II the adversary $\mathcal{A}$ may issue a decryption query $CT = (R, E, C)$, s.t. $CT \neq CT^* \wedge E = E^*$, e.g just replacing R^* with R in CT^*. In this situation, the simulator $\mathcal{B}$ just outputs $\perp$ as response. The reason is that the probability for CT to be a valid ciphertext of Π is negligible. Let $\sigma \in \{\sigma^*_0, \sigma^*_1\}$ be the plaintext enclosed in E^*. Since Σ is IND-CCA2 secure, E^* would not leak any information about σ. Thus, $\mathcal{A}$ could not distinguish $K = B(\sigma, R)$ from a random string chosen from the proper domain, then $M = C \oplus K$ also seems like a random string for $\mathcal{A}$. For a random message M, the probability of $Ver_{PK_S}(M\|R, \sigma) = 1$ is negligible. So is the probability that CT is a valid ciphertext.

Theorem 2 (IND-CCA Security). *If the signature scheme Γ is secure in the sense of UF-CMA and the encryption scheme Σ is secure in the sense of IND-CCA, then the proposed scheme Π of accountable decryption is secure in the sense of IND-CCA.*

Theorem 3 (IND-CPA Security). *If the signature scheme Γ is secure in the sense of UF-CMA and the encryption scheme Σ is secure in the sense of IND-CPA, then the proposed scheme Π of accountable decryption is secure in the sense of IND-CPA.*

The proofs of Theorem 2 and 3 are similar with Theorem 1. The differences are that the adversary $\mathcal{A}$ could not issue decryption queries and the simulator $\mathcal{B}$ has no access to the decryption oracle $\mathcal{O}_d$ in Query Phase II. We omit the details to avoid verbosity. Next, we show that our construction of Π has sound verifiability.

Theorem 4 (Soundness of Verifiability). *Suppose the signature scheme Γ is strong UF-CMA secure and the encryption scheme Σ is IND-CPA. Then the scheme Π described above has soundness of verifiability.*

Theorem 5 (Plaintext Authenticity). *If the digital signature scheme Γ is UF-CMA secure and the encryption scheme Σ is IND-CPA secure, the generic construction Π described above has plaintext authenticity.*

Also due to the page limitation, we omit the proofs of Theorem 4 and 5. Roughly, if $\mathcal{A}$ wins $Game^{\Pi}_{S}$ or $Game^{\Pi}_{A}$, the output of $\mathcal{A}$ will lead to a forgery signature of the underlying signature scheme, which is (strong) UF-CMA secure.

5 Applications

Our construction of accountable decryption is applicable to all encryption schemes. In the following, we only show how it can be applied for predicate encryption and public-key encryption with keyword search because of their unique applications. For the former, we apply our method to the full-fledged predicate encryption scheme (PE-KSW) described in [3]. In the resulting scheme, the decryptor gives a proof by decrypting correctly, which assures the verifier that the decryptor has the proper attributes (or capability) to decrypt. We envisage it as a realization of attribute based access control. For the latter, the construction (based on pairing) in [1] does not encrypt a payload message, so it needs a slight modification. The resulting scheme allows the search query issuer (i.e. the verifier) to detect whether a proxy (i.e. the decryptor) manipulates the testing results. We envisage it as a security enhancement in publish-subscribe systems whereby the untrusted proxy filters events to subscribers making payments on received events.

5.1 A Full-Fledged Predicate Encryption Scheme with Accountable Decryption

By instantiating Σ in Section 4 with PE-KSW, we obtain the construction for the full-fledged PE-KSW scheme with accountable decryption. The used signature scheme is also denoted by $\Gamma = (KeyGen_\Gamma, Sign, Ver)$.

Setup(1^n)**:** The setup algorithm first runs $\mathcal{G}(1^n)$ to obtain $(p, q, r, \mathbb{G}, \mathbb{G}_T, e)$ with $\mathbb{G} = \mathbb{G}_p \times \mathbb{G}_q \times \mathbb{G}_r$. Next, it computes g_p, g_q and g_r as generators of $\mathbb{G}_p, \mathbb{G}_q$ and $\mathbb{G}_r$, respectively. It then chooses $R_{1,i}, R_{2,i} \in \mathbb{G}_r$ and $h_{1,i}, h_{2,i} \in \mathbb{G}_p$ uniformly and randomly for $i = 1$ to n and $R_0 \in \mathbb{G}_p$. The public parameters include $(N = pqr, \mathbb{G}, \mathbb{G}_T, e)$ along with:

$$PK = \begin{pmatrix} g_p, \;\; g_r, \;\; Q = g_q \cdot R_0, \;\; P = e(g_p, h)^\gamma, \\ \{H_{1,i} = h_{1,i} \cdot R_{1,i}, \;\; H_{2,i} = h_{2,i} \cdot R_{2,i}\}_{i=1}^n \end{pmatrix}.$$

The master secret key MSK is $(p, q, r, g_q, h^{-\gamma}, \{h_{1,i}, h_{2,i}\}_{i=1}^n)$.
In addition, sender $\mathcal{S}$ runs $KeyGen_\Gamma(1^n)$ to obtain a signature key-pair $(SK_\mathcal{S}, VK_\mathcal{S})$ with message space $\mathcal{MS}_\Gamma$ and signature space $\mathcal{SS}_\Gamma$. Besides, the algorithm chooses an injective function f_{map}, which are defined as $f_{map} : \mathcal{SS}_\Gamma \to \mathbb{G}_T$, where k is the length of messages and is polynomial in n as described in Section 4, and the inverse function of $f_{map}()$ is denoted by $f_{map}^{-1}()$. Finally, using the Goldreich-Levin theorem, the algorithm outputs a hardcore function $B()$ w.r.t the encryption algorithm of Σ. B is defined as $B : \mathcal{MS}_\Sigma \times \{0,1\}^{m+k-1}$, where m is the length of elements in $\mathbb{G}_T$.

GenKey$(MSK, \boldsymbol{v})$**:** Let $\boldsymbol{v} = (v_1, \ldots, v_n)$. It first chooses random $r_{1,i}, r_{2,i} \in \mathbb{Z}_p$ for $i = 1$ to n, random $f_1, f_2 \in \mathbb{Z}_q$, random $R_5 \in \mathbb{G}_r$ and random $Q_6 \in \mathbb{G}_q$, and then outputs

$$TK_v = \begin{pmatrix} K_0 = R_5 \cdot Q_6 \cdot h^{-\gamma} \cdot \prod_{i=1}^n h_{1,i}^{-r_{1,i}} h_{2,i}^{-r_{2,i}}, \\ \{K_{1,i} = g_p^{r_{1,i}} \cdot g_q^{f_1 v_i}, \;\; K_{2,i} = g_p^{r_{2,i}} \cdot g_q^{f_2 v_i}\}_{i=1}^n \end{pmatrix}.$$

Enc$(PK, SK_\mathcal{S}, \boldsymbol{x}, M)$**:** To encrypt a message M, sender $\mathcal{S}$ firstly chooses a random $R \in_R \{0,1\}^{m+k-1}$ and signs it as $\sigma \leftarrow Sign_{SK_\mathcal{S}}(M||R)$.
Let $\boldsymbol{x} = (x_1, \ldots, x_n)$ with $x_i \in \mathbb{Z}_N$. It then encrypts the signature as follows.

– Choose random $s, a, b \in \mathbb{Z}_N$ and $R_{3,i}, R_{4,i} \in \mathbb{G}_r$ for $i = 1$ to n.
– Set

$$E = \begin{pmatrix} C' = f_{map}(\sigma) \cdot P^s, \ \ C_0 = g_p^s, \\ \{C_{1,i} = H_{1,i}^s \cdot Q^{ax_i} \cdot R_{3,i}, \ \ C_{2,i} = H_{2,i}^s \cdot Q^{bx_i} \cdot R_{4,i}\}_{i=1}^n \end{pmatrix}.$$

It outputs a three-tuple (R, E, C) as the ciphertext CT, where $C = M \oplus B(\sigma, R)$.

Dec$(TK_v, VK_{\mathcal{S}}, CT)$**:** Let TK_v, CT be as above. Receiver $\mathcal{P}$ computes

$$\sigma' = C' \cdot e(C_0, K_0) \cdot \prod_{i=1}^{n} e(C_{1,i}, K_{1,i}) \cdot e(C_{2,i}, K_{2,i})$$

$$\sigma = f_{map}^{-1}(\sigma') \qquad M = C \oplus B(\sigma, R).$$

Finally, it outputs (M, σ), if $Ver_{VK_{\mathcal{S}}}(M||R, \sigma) = 1$; otherwise, outputs $\perp$.

Verify$(VK_{\mathcal{S}}, CT, M, \sigma)$**:** Let $CT = (R, E, C)$. A verifier $\mathcal{V}$ computes $M' = C \oplus B(\sigma, R)$. It outputs 1, if $Ver_{VK_{\mathcal{S}}}(M||R, \sigma) = 1 \wedge M = M'$; otherwise, outputs 0.

It is clear that if $\mathcal{P}$ cannot decrypt properly, it is infeasible for her to construct an account decryption proof. Our modification slightly increases the communication and computation costs. The increased communication cost mainly lies in longer ciphertext, due to R, C of a ciphertext. However, the increased communication cost is still slight compared with the length of E, i.e. the ciphertext of PE-KSW. The computation cost is on signature generation and verification which are rather small as compared to the bilinear mappings used in PE-KSW. The computation of the hardcore function is lightweight as well, as it involves a sequence of additions and multiplication modulo 2.

As proved in [3], the underlying encryption scheme Σ, i.e. PE-KSW, is attribute-hiding. It is straightforward to show the above construction preserves this property. Due to the length limit, we informally explain the proof as follows. On inputting $(\boldsymbol{x}_0^*, M_0^*)$ and $(\boldsymbol{x}_1^*, M_1^*)$ as two challenge attribute-plaintext pairs, the challenger chooses a bit β and runs the above encryption algorithm to get the challenge ciphertext, i.e. $CT^* \leftarrow \mathsf{Enc}(PK, SK_{\mathcal{S}}, \boldsymbol{x}_\beta^*, M_\beta^*)$. Let $CT^* = (R^*, E^*, C^*)$. We have the following facts:

1. Because R^* is chosen randomly and independent from β, this component of the challenge ciphertext would not leak any information about β.
2. Since the underlying encryption Σ is attribute-hiding, E^* keeps both β and corresponding signature σ private.
3. Due to the privacy of σ and the randomness of the output of the hardcore function $B()$, C^* appears as a random string for the adversary and leaks no information about β too.

Consequently, the scheme proposed above is also attribute-hiding.

5.2 PEKS with Verifiable Test

For PEKS schemes, we are concerned with the correctness of keyword test, rather than the actual decryption of the payload. In our generic construction, the underlying encryption scheme Σ is used to encrypt a signature. However, the PEKS function in [1] takes as input an attribute associated with the encrypted message. A ciphertext of PEKS in [1] is in the form of $(g^r, H(w))$ where r is a random and w is related to r and a keyword W. We replace $H(w)$ in the ciphertext with $H(w) \oplus \sigma$, where σ is an encrypted message. Note that the modified version is just the IBE construction proposed by Boneh and Franklin in [22]. We call this IBE construction as IBE-BF for short, and denote the modified version by Σ. The used signature scheme is denoted by $\Gamma = (KeyGen_\Gamma, Sign, Ver)$. The details of PEKS with verifiable test (denoted by Π) are shown as below.

KeyGen(1^λ)**:** This algorithm first runs $\mathcal{G}(1^\lambda)$ to obtain $(p, \mathbb{G}, \mathbb{G}_T, e)$. Next, it picks a random $\alpha \in \mathbb{Z}_p^*$ and a generator g of $\mathbb{G}$, and outputs $A_{pub} = (g, h = g^\alpha)$ and $A_{priv} = \alpha$.

In addition, sender $\mathcal{S}$ runs $KeyGen_\Gamma(1^\lambda)$ to obtain a signature key-pair $(SK_\mathcal{S}, VK_\mathcal{S})$ with message space $\mathcal{MS}_\Gamma$ and signature space $\mathcal{SS}_\Gamma$. She also chooses two hash functions: $H_1 : \{0,1\}^* \to \mathbb{G}$, $H_2 : \mathbb{G}_T \to \{0,1\}^m$, where m is the length of signatures of Γ, k is the length of plaintexts of Π and both k, m are polynomial in λ. Finally, this algorithm outputs a hardcore function $B()$ w.r.t the encryption algorithm of the modified PEKS Σ. B is defined as $B : \mathcal{MS}_\Sigma \times \{0,1\}^{m+k-1}$.

Trapdoor(A_{priv}, W)**:** It outputs $T_W = H_1(W)^\alpha \in \mathbb{G}$.

PEKS$(A_{pub}, SK_\mathcal{S}, W)$**:** $\mathcal{S}$ firstly chooses a random message $M \in \{0,1\}^k$ and a random $R \in_R \{0,1\}^{m+k-1}$. Then it signs $M||R$ as $\sigma \leftarrow Sign_{SK_\mathcal{S}}(M||R)$. Next, it computes $w = e(H_1(W), h^r)$ for a random $r \in \mathbb{Z}_p^*$ and $E = (g^r, H_2(w) \oplus \sigma)$. The output ciphertext is $CT = (R, E, C)$ where $C = M \oplus B(\sigma, R)$.

Test$(T_W, VK_\mathcal{S}, CT)$**:** Let $CT = (R, E, C)$ and $E = (E_1, E_2)$. $\mathcal{P}$ uses the trapdoor T_W to decrypt the embedded signature as $\sigma = E_2 \oplus H_2(e(T_W, E_1))$. Then, the message is computed as $M = C \oplus B(\sigma, R)$. Eventually, $\mathcal{P}$ outputs $true$ along with (M, σ), if $Ver_{VK_\mathcal{S}}(M||R, \sigma) = 1$ holds; otherwise, outputs $false$.

Verify$(VK_\mathcal{S}, CT, M, \sigma)$**:** Let $CT = (R, E, C)$. A verifier $\mathcal{V}$ computes $M' = C \oplus B(\sigma, R)$. It outputs 1, iff $Ver_{VK_\mathcal{S}}(M||R, \sigma) = 1 \wedge M = M'$; otherwise, outputs 0.

When $\mathcal{P}$ indeed has a matching trapdoor for a keyword W, it can remove $H_2(w)$ from E_2 to obtain the signature σ, and further recovers message M. If M is exactly the random message chosen during encryption algorithm PEKS(), the verification algorithm of Γ outputs 1. The transformation mainly introduces one extra signature verification to Test() and a signature generation to PEKS(), while other increased computation cost is negligible. As compared to PEKS, the ciphertext length in the above construction is increased by $m+2k-1$ bits. For a positive response, the proxy (i.e. the prover) has to append a tuple (M, σ), and it adds $k + m$ bits to the communication cost between the proxy and the user.

6 Conclusion

To summarize, we have proposed a new cryptographic notion called *accountable decryption* which combines verifiable decryption and message authenticity. With the formulation of accountable decryption and its security, we design a generic construction, which is applicable for any semantically secure encryption scheme. The security properties of the generic construction are proved rigorously. Taking predicate encryption and public key encryption with keyword search as two concrete examples, we transform them to the ones with accountable decryption, while keeping their notable properties and efficiency.

Acknowledgements. We are grateful to the anonymous reviewers of ACISP 2012 for their helpful comments. This work is partially funded by Natural Science Foundation of China (No. 60970111, 60903189, 61133014) and Doctoral Fund of Ministry of Education of China (New Teachers) No. 20090073120024, and also supported by the Office of Research, Singapore Management University.

References

1. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public Key Encryption with Keyword Search. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004)
2. Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010)
3. Katz, J., Sahai, A., Waters, B.: Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)
4. Neuman, B., Ts'o, T.: Kerberos: an authentication service for computer networks. IEEE Communications Magazine 32, 33–38 (1994)
5. Camenisch, J.L., Shoup, V.: Practical Verifiable Encryption and Decryption of Discrete Logarithms. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 126–144. Springer, Heidelberg (2003)
6. Selvi, S.S.D., Sree Vivek, S., Pandu Rangan, C.: Identity Based Public Verifiable Signcryption Scheme. In: Heng, S.-H., Kurosawa, K. (eds.) ProvSec 2010. LNCS, vol. 6402, pp. 244–260. Springer, Heidelberg (2010)
7. Chow, S.S.M.: Verifiable Pairing and Its Applications. In: Lim, C.H., Yung, M. (eds.) WISA 2004. LNCS, vol. 3325, pp. 170–187. Springer, Heidelberg (2005)
8. Chow, S.S.M., Yiu, S., Hui, L., Chow, K.: Efficient Forward and Provably Secure ID-Based Signcryption Scheme with Public Verifiability and Public Ciphertext Authenticity. In: Lim, J., Lee, D. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 352–369. Springer, Heidelberg (2004)

9. Boyen, X.: Multipurpose Identity-Based Signcryption. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 383–399. Springer, Heidelberg (2003)
10. Shin, J., Lee, K., Shim, K.: New DSA-Verifiable Signcryption Schemes. In: Lee, P., Lim, C. (eds.) ICISC 2002. LNCS, vol. 2587, pp. 35–47. Springer, Heidelberg (2003)
11. Bao, F., Deng, R.H.: A Signcryption Scheme with Signature Directly Verifiable by Public Key. In: Imai, H., Zheng, Y. (eds.) PKC 1998. LNCS, vol. 1431, pp. 55–59. Springer, Heidelberg (1998)
12. Barbosa, M., Farshim, P.: Delegatable homomorphic encryption with applications to fully secure outsourcing of computation. Cryptology ePrint Archive, Report 2011/215 (2011)
13. Boneh, D., Gentry, C., Lynn, B., Shacham, H.: Aggregate and Verifiably Encrypted Signatures from Bilinear Maps. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 416–432. Springer, Heidelberg (2003)
14. Rückert, M., Schröder, D.: Security of Verifiably Encrypted Signatures and a Construction without Random Oracles. In: Shacham, H., Waters, B. (eds.) Pairing 2009. LNCS, vol. 5671, pp. 17–34. Springer, Heidelberg (2009)
15. Rückert, M.: Verifiably Encrypted Signatures from RSA without NIZKs. In: Roy, B., Sendrier, N. (eds.) INDOCRYPT 2009. LNCS, vol. 5922, pp. 363–377. Springer, Heidelberg (2009)
16. Lu, S., Ostrovsky, R., Sahai, A., Shacham, H., Waters, B.: Sequential Aggregate Signatures and Multisignatures Without Random Oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 465–485. Springer, Heidelberg (2006)
17. Rückert, M., Schneider, M., Schröoder, D.: Generic Constructions for Verifiably Encrypted Signatures without Random Oracles or NIZKs. In: Zhou, J., Yung, M. (eds.) ACNS 2010. LNCS, vol. 6123, pp. 69–86. Springer, Heidelberg (2010)
18. Zheng, Y.: Digital Signcryption or How to Achieve Cost (Signature & Encryption) $<<$ Cost(Signature) + Cost(Encryption). In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 165–179. Springer, Heidelberg (1997)
19. Matsuda, T., Matsuura, K., Schuldt, J.C.N.: Efficient Constructions of Signcryption Schemes and Signcryption Composability. In: Roy, B., Sendrier, N. (eds.) INDOCRYPT 2009. LNCS, vol. 5922, pp. 321–342. Springer, Heidelberg (2009)
20. Goldwasser, S., Micali, S.: Probabilistic encryption. J. Comput. Syst. Sci. 28, 270–299 (1984)
21. Goldreich, O.: Foundations of Cryptography: Basic Tools. Cambridge University Press, New York (2000)
22. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. SIAM J. Comput. 32(3), 586–615 (2003)

Threshold Ciphertext Policy Attribute-Based Encryption with Constant Size Ciphertexts

Aijun Ge[1], Rui Zhang[2], Cheng Chen[2], Chuangui Ma[1], and Zhenfeng Zhang[2]

[1] Department of Applied Mathematics, Zhengzhou Information Science and Technology Institute, Zhengzhou, Henan 450002, China
geaijun2011@gmail.com, cgm20090505@163.com

[2] State Key Laboratory of Information Security (SKLOIS), Institute of Information Engineering (IIE), Chinese Academy of Sciences (CAS), Beijing, 100190, China
r-zhang@iie.ac.cn, {chencheng,zfzhang}@is.iscas.ac.cn

Abstract. In PKC 2010, Herranz *et al.* proposed the first ciphertext policy attribute-based encryption (CP-ABE) scheme with constant size ciphertexts for threshold predicates. However, their scheme was only secure against chosen plaintext attacks (CPA), which was impossible to obtain security against chosen ciphertext attacks (CCA) in the standard model, and they left open the following three problems for CP-ABE schemes with constant size ciphertexts, i.e., how to achieve full security (i.e., not only the selective security), CCA security in the standard model, and security reduction to a more standard mathematical problem. In this paper, we answer the last two of these three problems affirmatively. Towards our goal, we first design a CPA secure threshold CP-ABE scheme, which can be further upgraded to the CCA security. The security of our schemes can be proved under the decisional q-Bilinear Diffie-Hellman Exponent (q-BDHE) assumption in the selective model. To the best of our knowledge, this is the first construction of CCA secure CP-ABE scheme with constant size ciphertexts that can support flexible threshold access structure in the standard model.

Keywords: attribute-based encryption, constant size ciphertext, chosen ciphertext security, threshold access structure.

1 Introduction

Attribute-based encryption (ABE) was introduced by Sahai and Waters [18] (first under the name fuzzy identity-based encryption), as an extension of identity-based encryption (IBE) [19]. In an ABE system, a user's private keys and ciphertexts are associated with sets of descriptive attributes, and decryption is possible only if there is a match between the attributes of the ciphertext and the user's private keys.

Goyal *et al.* [10] further extended this idea and introduced two variants: key policy attribute-based encryption (KP-ABE) and ciphertext policy attribute-based encryption (CP-ABE). In a CP-ABE system, a user's private key is associated with a set of attributes and encrypted ciphertext will specify an access policy over attributes. A user can decrypt if and only if his attributes satisfy the ciphertext's policy. While in a KP-ABE system, the situation is reversed: the private key is associated with an access policy, and the ciphertext is associated with a set of attributes. A user can decrypt if and only if the attributes associated with the ciphertext satisfy the user's private key policy. A remarkable feature of ABE schemes is collusion resistance: even if multiple users collude, they should only be able to decrypt a ciphertext if at least one of the users could decrypt it on their own.

Since ABE can simultaneously provide access control and data confidentiality functionalities, it has become a promising technique for building secure databases. ABE has received a lot of attention in recent years, and a number of schemes have been proposed [18,10,5,8,11,14,24,2,21,16,6]. However, in most of the previous ABE schemes (except [8,11,24,2,6]), the ciphertext size is at least linear to the number of attributes involved in the access policy. So it is with the number of pairing computations in the ABE schemes. These limit the usage of ABE in real life applications very much, especially for the scenarios where bandwidth issues are of great importance. Therefore, CP-ABE schemes whose ciphertexts are of constant size, i.e., regardless of the number of underlying attributes, are more preferable.

The first CP-ABE scheme with constant ciphertext size was proposed by Emura *et al.* in [8], but the policies in this scheme were restricted to AND-gates (or (t,t)-threshold). Motivated by the dynamic threshold IBE scheme of [7], Herranz *et al.* [11] constructed another constant size ciphertexts CP-ABE scheme, and this scheme could work for the (t,n)-threshold policy. Their scheme is not very efficient, as the encryption operation requires $n + t + 1$ exponentiations and the decryption operation requires 3 pairings and $O(t^2)$ exponentiations. In addition, the security of Herranz *et al.*'s scheme [11] is reduced to an augment assumption: Multi-Sequence of Exponents Decisional Diffie-Hellman (aMSE-DDH) problem, which is not a standard one.

Security against chosen ciphertext attacks (i.e., CCA security) is considered as an important notion for ABE schemes. While it is easy to obtain CCA security in the random oracle model by applying Fujisaki-Okamoto conversion [9], in the standard model, it is not so easy. Some ABE schemes [10,5] used the CHK technique [4] to achieve CCA security if the ABE scheme satisfied delegatability. Recently, Yamada *et al.* [22] generalized this idea, and showed a generic construction to CCA security if the ABE scheme provided either delegatability or verifiability. Using a method different from [22], Chen *et al.* [6] proposed another CCA secure CP-ABE construction with constant size ciphertexts, but their scheme only admitted AND-gates. Note that Herranz *et al.*'s scheme [11] does not satisfy the framework of [22], so their scheme does not seems to be able to achieve CCA security in the standard model directly.

Up to now, most of the known ABE schemes in the literature focus on the security under selective attacks, i.e., the attacker has to commit to a target access structure before the setup phase. Lewko *et al.* [14] first obtained full security by adapting the dual system encryption technique [20] to the ABE case. Two other schemes [17,16] also achieved full security.

Our Contributions. As noted by Herranz *et al.* in [11], there are three open problems for CP-ABE with constant size ciphertexts: how to achieve full security (i.e., not the selective security merely), CCA security in the standard model, and security reduction to a more standard mathematical problem. In this paper, we have solved the last two of these three open problems. Using the "default attributes" approach, which was indicated by Sahai and Waters in [18], we first propose a threshold CP-ABE scheme with constant size ciphertexts, and the design strategy is completely different from Attrapadung *et al.*'s method [2]. The security against chosen plaintext attacks of our CP-ABE scheme can be proven in the standard model by reduction to the decisional q-Bilinear Diffie-Hellman Exponent (q-BDHE) problem. q-BDHE problem is introduced and shown to be hard by Boneh *et al.* in [3], which seems more natural than the aMSE-DDH problem used in [11].

As our threshold CP-ABE scheme satisfies the property of delegatability, it is trivial to achieve CCA security using the CHK technique [4] or the generic technique from [22]. However, these conversions are not efficient, especially for the small attribute universe case. Instead, we use a technique similar to the one used in [6] to extend our scheme to be CCA secure without losing its efficiency: the CCA secure threshold CP-ABE scheme still maintains both constant size ciphertexts and computation cost in the encryption and decryption algorithms.

2 Preliminaries

In this section, we review some useful notations and notions.

Notations. We denote parameters by Greek letters (e.g., λ, ε, etc.), with λ always denoting the security parameter. Real numbers and integers are denoted by lowercase English letters (p, q, x, y, etc.). Sets are denoted by capital English letters (S, W, etc.). We denote Υ as the access structure used in the encryption algorithm. If x is a string, let $|x|$ denotes its length, while if S is a set then $|S|$ denotes its size. If $k \in \mathbb{N}$, a function $f(k)$ is negligible if $\exists\ k_0 \in \mathbb{N}$, $\forall\ k > k_0$, $f(k) < 1/k^c$, where $c > 0$ is a constant.

2.1 Bilinear Groups

We briefly review bilinear maps and bilinear groups. Let $\mathbb{G}, \mathbb{G}_T$ be cyclic multiplicative groups of prime order p, and g is a generator of $\mathbb{G}$. A bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ has the following properties:

- **Bilinearity:** For all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p^*$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
- **Non-degeneracy:** $e(g, g) \neq 1$.
- **Computability:** It is efficient to compute $e(u, v)$ for all $u, v \in \mathbb{G}$.

2.2 Hardness Assumption

The security of our schemes is based on the decisional q-BDHE problem, which is defined as follows: Denote $\overrightarrow{y}_{g,\alpha,q} = (g_1, g_2, ..., g_q, g_{q+2}, ...g_{2q})$ where $g_i = g^{\alpha^i}$ for some unknown $\alpha \in \mathbb{Z}_p^*$. An algorithm $\mathcal{A}$ that outputs $b \in \{0,1\}$ has advantage ϵ in solving the decisional q-BDHE problem if

$$\left|\Pr\left[\mathcal{A}(g, h, \overrightarrow{y}_{g,\alpha,q}, e(g,h)^{(\alpha^{q+1})}) = 0\right] - \Pr\left[\mathcal{A}(g, h, \overrightarrow{y}_{g,\alpha,q}, T) = 0\right]\right| \geqslant \epsilon$$

where the probability is over the random choice of generators g, h in $\mathbb{G}$, the random choice of α in $\mathbb{Z}_p^*$, the random choice of $T \in \mathbb{G}_T$, and the random bits used by $\mathcal{B}$.

Definition 1. *We say that the decisional (t, ϵ, q)-BDHE assumption holds if no t-time algorithm has advantage at least ϵ in solving the decisional q-BDHE problem in $(\mathbb{G}, \mathbb{G}_T)$.*

2.3 Syntax of CP-ABE Scheme

There are two entities in the CP-ABE system: a central attribute authority and users. The authority is in charge of the attribute private key, and each user receives secret keys corresponding to the attributes that he satisfies from the central attribute authority. Denote the universe of attributes as $\mathbb{U}$. Intuitively, an access structure Υ is a boolean function over $\mathbb{U}$ that returns either 0 or 1 given an attribute set $A \subseteq \mathbb{U}$. We say that A satisfies Υ if and only if $\Upsilon(A) = 1$. The access structure used in our scheme is threshold policy, the sender chooses an attribute set $S \subseteq \mathbb{U}$ and a threshold t such that $1 \leqslant t \leqslant |S|$. Only the user with attributes A who holds t or more attributes in S can satisfy this policy.

A CP-ABE system consists of four fundamental algorithms: **Setup**, **KeyGen**, **Encrypt** and **Decrypt**. In addition, we allow for the option of a fifth algorithm **Delegate**.

- **Setup**$(\lambda, \mathbb{U})$: This algorithm takes as input a security parameter λ and an attribute universe description $\mathbb{U}$, and it returns public parameters *params* and a master secret key *msk* as output. The master secret key *msk* is kept secret by the central authority, and is used to generate users' secret keys, while the public parameters *params* containing the universe of attributes $\mathbb{U}$ are made public.
- **KeyGen**$(msk, params, A)$: This algorithm takes a master secret key *msk*, public parameters *params* and a user's attribute set $A \subseteq \mathbb{U}$ as input, and it returns the attribute private key SK_A.
- **Delegate**$(SK_A, A' \subseteq A)$: This algorithm takes a private key SK_A for some attribute set A and a subset $A' \subseteq A$ as input, and it returns an attribute private key $SK_{A'}$ for the set of attributes A'.
- **Encrypt**$(M, params, \Upsilon)$: This algorithm takes a message M, public parameters *params* and an access structure Υ as input, and it returns a ciphertext CT such that only the user with attribute set A satisfies Υ can decrypt CT.

- **Decrypt**$(params, \Upsilon, SK_A, CT)$: This algorithm takes public parameters *params*, an access structure Υ, a ciphertext CT and a private key SK_A for a set of attributes A, and it returns "$\perp$" indicating invalid ciphertext, or a message M if the set of attributes A satisfies the access structure Υ.

Correctness. For any correctly generated ciphertexts on the message M and the access structure Υ, with the private key SK_A satisfying $\Upsilon(A) = 1$, it is required that: $\mathbf{Decrypt}(params, \Upsilon, SK_A, \mathbf{Encrypt}(M, params, \Upsilon)) = M$.

2.4 Security Model for CP-ABE

We now describe the security model against chosen plaintext attacks (CPA) for CP-ABE. This is defined by a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ as follows:

- **Setup**: The challenger $\mathcal{C}$ chooses a security parameter λ and runs **Setup** algorithm to generate a master secret key *msk* and public parameters *params*. $\mathcal{C}$ gives *params* to the adversary $\mathcal{A}$, while keeping the *msk* secret.
- **Phase 1**: The adversary $\mathcal{A}$ queries the challenger $\mathcal{C}$ for private keys corresponding to sets of attributes $A_1, ..., A_{q_{K_1}}$.
- **Challenge**: The adversary $\mathcal{A}$ declares two equal length messages M_0 and M_1 and an access structure Υ^*. This access structure Υ^* cannot be satisfied by any of the queried attribute sets $A_1, ..., , A_{q_{K_1}}$. The challenger $\mathcal{C}$ flips a random coin $\beta \in \{0, 1\}$ and encrypts M_β under Υ^*. The result ciphertext CT^* is given to the adversary $\mathcal{A}$.
- **Phase 2**: The adversary $\mathcal{A}$ queries the challenger $\mathcal{C}$ for private keys corresponding to sets of attributes $A_{q_{K_1}+1}, ..., A_{q_K}$, with the restriction that none of these attributes satisfies Υ^*.
- **Guess**: Finally, the adversary $\mathcal{A}$ outputs a guess β' for β.

We say that the adversary $\mathcal{A}$ succeeds if $\beta' = \beta$, and the advantage of $\mathcal{A}$ is defined as $Adv^{CPA}_{\mathcal{A},CP-ABE} = |\Pr[\beta' = \beta] - 1/2|$.

Definition 2. *A CP-ABE scheme is (t, ϵ, q_K)-CPA secure if the advantage of $\mathcal{A}$ $Adv^{CPA}_{\mathcal{A},CP-ABE}$ is negligible for each t-time adversaries $\mathcal{A}$ who makes a total of q_K private key queries, we have that $Adv^{CPA}_{\mathcal{A},CP-ABE} \leqslant \epsilon$. We say that a CP-ABE scheme is CPA secure if for all polynomially bounded t, q_K, the advantage ϵ is negligible.*

Note that the security model against chosen ciphertext attacks (CCA) for CP-ABE can be easily extended by allowing for decryption queries in **Phase 1** and **Phase 2**, where no decryption query is allowed on CT^* in Phase 2. We say that a system is selectively secure if we add an **Init** stage before **Setup** where the adversary commits to the challenge access structure Υ^*.

3 The Proposed CP-ABE Schemes

In this section, we present two constructions that will be proved secure without random oracles under the decisional q-BDHE assumption. The first scheme is

CPA secure and the second one is CCA secure. Both of our CP-ABE schemes have constant size ciphertexts and pairing computation costs. The security analysis of these schemes will be postponed to Section 4.

3.1 Construction I: CPA Secure CP-ABE

Let $\mathbb{G}$ be a bilinear group of prime order p with a generator g, and $e : \mathbb{G}\times\mathbb{G} \rightarrow \mathbb{G}_T$ is an efficient bilinear map. Our CPA secure threshold CP-ABE scheme with constant size ciphertexts works as follows:

- **Setup**$(\lambda, \mathbb{U})$:Takes as input a security parameter λ and the universal set of attributes $\mathbb{U} = \{att_1, att_2, ..., att_\ell\}$, the central attribute authority chooses a suitable encoding function ρ sending each of the ℓ attributes att_i onto a different element $\rho(att_i) = x_i \in \mathbb{Z}_p$. For simplicity, we define $\rho(att_i) = i$ for each $att_i \in \mathbb{U}$. Besides these ℓ normal attributes $\mathbb{U}$, the attribute authority also chooses $\ell-1$ default attributes $\mathbb{U}' = \{att'_1, att'_2, ..., att'_{\ell-1}\}= \{\ell+1, ..., 2\ell-1\}$. Note that the default attributes $\mathbb{U}'(|\mathbb{U}'| = \ell-1)$ are possessed by every user in this system.

  After that, the central attribute authority chooses $g_2, h_0, h_1, ..., , h_{2\ell-1}$ uniformly at random from $\mathbb{G}$, a random $x \in \mathbb{Z}_p^*$, and sets $g_1 = g^x$, $Z = e(g_1, g_2)$. The master secret key is then $msk = x$ and the public parameters are $params = (g, g_2, Z, h_0, h_1, ..., h_{2\ell-1})$.
- **KeyGen**$(msk, params, A)$: The attribute authority takes the following steps to generate a private key for a user with his attribute set $A \subseteq \mathbb{U}$:
  1. Randomly choose an $\ell-1$ degree polynomial $q(\cdot)$ with $q(0) = x$;
  2. For each attribute $i \in (A \cup \mathbb{U}')$, the attribute authority selects a random r_i in $\mathbb{Z}_p$, and computes:

$$\begin{aligned} sk_i &= (g_2^{q(i)}(h_0h_i)^{r_i}, g^{r_i}, h_1^{r_i}, ..., h_{i-1}^{r_i}, h_{i+1}^{r_i}, ..., h_{2\ell-1}^{r_i}) \\ &= (a_i, b_i, c_{i,1}, ..., c_{i,i-1}, c_{i,i+1}, ..., c_{i,2\ell-1}); \end{aligned}$$

  3. Finally, the attribute authority outputs the private key: $\{sk_i\}_{i\in A\cup\mathbb{U}'}$.
- **Delegate**$(SK_A, A' \subseteq A)$: The delegate algorithm takes the attribute private key SK_A for some attribute set $A \subseteq \mathbb{U}$ and a subset $A' \subseteq A$ as input. The attribute private key SK_A for the set of attributes A is of the form $SK_A = \{sk_i\}_{i\in A\cup\mathbb{U}'}$. The algorithm chooses random $\widetilde{r}_i$ for each $i \in A' \cup \mathbb{U}'$. Then it creates a new secret key as $SK_{A'} = \{\widetilde{sk}_i\}_{i\in A'\cup\mathbb{U}'}$, where

$$\begin{aligned} \widetilde{sk}_i &= (\widetilde{a}_i, \widetilde{b}_i, \widetilde{c}_{i,1}, ..., \widetilde{c}_{i,i-1}, \widetilde{c}_{i,i+1} ..., \widetilde{c}_{i,2\ell-1}) \\ &= (a_i(h_0h_i)^{\widetilde{r}_i}, b_i g^{\widetilde{r}_i}, c_{i,1}h_1^{\widetilde{r}_i}, ..., c_{i,i-1}h_{i-1}^{\widetilde{r}_i}, c_{i,i+1}h_{i+1}^{\widetilde{r}_i}, ..., c_{i,2\ell-1}h_{2\ell-1}^{\widetilde{r}_i}) \end{aligned}$$

- **Encrypt**$(M, params, \Upsilon_{t,S})$: Given a threshold access structure $\Upsilon_{t,S}$, i.e., a subset $S \subseteq \mathbb{U}$ and a threshold $1 \leqslant t \leqslant |S|$, the sender does the following to encrypt a message $M \in \mathbb{G}_T$:
  1. The sender picks a default attribute subset $\Omega \subseteq \mathbb{U}'$ with the first $\ell-t$ elements, i.e., $\Omega = \{\ell+1, \ell+2, ..., 2\ell-t\}$;

2. The sender chooses a random value $s \in \mathbb{Z}_p$, and computes the ciphertext $CT = (C_0, C_1, C_2)$ as follows:

$$C_0 = M \cdot Z^s, C_1 = g^s, C_2 = (h_0 \prod_{j \in S \cup \Omega} h_j)^s.$$

– **Decrypt**$(params, \Upsilon_{t,S}, SK_A, CT)$: Any user with a set of attributes A satisfying the threshold access structure $\Upsilon_{t,S}$ (i.e., there exists a subset $A' \subseteq A \cap S$ with $|A'| = t$) can use the secret key $\{sk_i\}_{i \in A' \cup \mathbb{U}'}$ to decrypt the ciphertexts as follows:
  1. The user chooses the first $\ell - t$ elements in $\mathbb{U}'$ and sets $\Omega = \{\ell + 1, \ell + 2, ..., 2\ell - t\}$;
  2. After that, the user calculates:

$$D_1 = \prod_{i \in A' \cup \Omega} (a_i \prod_{j \in S \cup \Omega, j \neq i} (c_{i,j}))^{\Delta_{i,A' \cup \Omega}(0)}, D_2 = \prod_{i \in A' \cup \Omega} (b_i)^{\Delta_{i,A' \cup \Omega}(0)};$$

  3. Finally, the user recovers the message by computing $M = C_0 \cdot e(C_2, D_2)/e(C_1, D_1)$.

Correctness: Correctness is shown in Appendix A.

3.2 Construction II: CCA Secure CP-ABE

Motivated by the work of [23,13,6], we present an efficient CCA secure CP-ABE construction with the small attribute universe. Compared with the CHK technique [4] or the generic conversion in [22], our construction is very simple and compact: the ciphertext size and public key size do not increase much, and so it is with the computation costs increase in encryption and decryption operations. Our CCA secure construction works as follows:

– **Setup**$(\lambda, \mathbb{U})$: The attribute authority randomly picks $(\delta_1, \delta_2, \delta_3) \in \mathbb{G}$ as well as a collision-resistant hash function $H : \{0, 1\}^* \to \mathbb{Z}_p^*$. The other part of the public parameters $params = (g, g_2, Z, h_0, h_1, ..., h_{2\ell-1}, \delta_1, \delta_2, \delta_3, H)$ and the master secret key $msk = x$ are chosen samt to the Setup algorithm in Construction I given in Section 3.1.
– **KeyGen**$(msk, params, A)$: The key generation algorithm is same to the KeyGen algorithm in Construction I given in Section 3.1.
– **Delegate**$(SK_A, A' \subseteq A)$: The delegate algorithm is same to the delegate algorithm in Construction I given in Section 3.1.
– **Encrypt**$(M, params, \Upsilon_{t,S})$: Given a threshold access structure $\Upsilon_{t,S}$, the sender does the following to encrypt a message $M \in \mathbb{G}_T$:
  1. The sender picks a default attribute subset $\Omega \subseteq \mathbb{U}'$ with the first $\ell - t$ elements, i.e., $\Omega = \{\ell + 1, \ell + 2, ..., 2\ell - t\}$;
  2. The sender chooses random $s, r \in \mathbb{Z}_p$, and computes the follows:

$$C_0 = M \cdot Z^s, C_1 = g^s, C_2 = (h_0 \prod_{j \in S \cup \Omega} h_j)^s, C_3 = (\delta_1^c \delta_2^r \delta_3)^s$$

  where $c = H(\Upsilon_{t,S}, C_0, C_1, C_2)$.
  3. The sender outputs the ciphertext $CT = (C_0, C_1, C_2, C_3, r)$.

– **Decrypt**$(params, \Upsilon_{t,S}, SK_A, CT)$: Suppose that a ciphertext $CT = (C_0, C_1, C_2, C_3, r)$ is encrypted using a threshold access structure $\Upsilon_{t,S}$, the user should first check the following two equations:

$$e(g, C_2) = e(C_1, h_0 \prod_{j \in S \cup \Omega} h_j), e(g, C_3) = e(C_1, \delta_1^c \delta_2^r \delta_3)$$

where $c = H(\Upsilon_{t,S}, C_0, C_1, C_2)$, and $\Omega = \{\ell+1, \ell+2, ..., 2\ell-t\}$ is the first $\ell - t$ elements in $\mathbb{U}'$. If one of the two equations does not hold, return $\perp$. Otherwise, the user can decrypt the ciphertext by using the secret key $\{sk_i\}_{i \in A' \cup \mathbb{U}'}$ which is same to the decrypt algorithm in Construction I given in Section 3.1.

Correctness: Correctness can be verified similarly with the above CPA secure CP-ABE in Section 3.1, so we omit it here too.

4 Security Analysis and Performance

4.1 Security Analysis

In this section, we prove that the scheme in Section 3.2 is secure against chosen ciphertext attacks, assuming that the decisional q-BDHE assumption is hard to solve in the standard model. Due to page limit, here we leave the security analysis of the scheme in Section 3.1 in the full version.

Theorem 1. *Our CP-ABE scheme in Section 3.2 is* (t, ϵ, q_K, q_D)*-CCA secure in the selective model, suppose the decisional* (t', ϵ', q)*-BDHE assumption holds in* $(\mathbb{G}, \mathbb{G}_T)$ *and collision resistant hash function exists. Here* $\epsilon'' = (\varepsilon - q_D/p)/2$*, where* q_K *and* q_D *are the number of private key queries and decryption queries an adversary can make at most*

Proof. Suppose there exists a (t, ϵ, q_K, q_D)-adversary $\mathcal{A}$ against our scheme, then we can construct a probabilistic polynomial time algorithm that can solve the decisional q-BDHE problem with probability at least ϵ' and in time at most t'. Suppose that the challenger $\mathcal{C}$ is given the the decisional q-BDHE challenge $(h, g, g^{\alpha}, ..., g^{\alpha^q}, g^{\alpha^{q+2}}, ..., g^{\alpha^{2q}}, T)$, where T is either $e(g^{\alpha^{q+1}}, h)$ or a random element of $\mathbb{G}_T$. Consider the game between the challenger $\mathcal{C}$ and the adversary $\mathcal{A}$ as follows:

Init: During the initial phase, $\mathcal{C}$ receives a challenge access structure $\Upsilon^*_{t^*,S^*}$, namely, a threshold value t^* out of the attributes S^*.

Setup: The challenger $\mathcal{C}$ first defines the universe of attributes used in this system as $\mathbb{U} = \{1, 2, ..., \ell\}$, and the $\ell - 1$ default attribute set $\mathbb{U}' = \{\ell + 1, \ell + 2, ..., 2\ell - 1\}$. For simplicity, here we let $2\ell - 1 = q$, and the $\ell - t^*$ default attributes chosen in the challenge $\Upsilon^*_{t^*,S^*}$ be $\Omega^* = \{\ell+1, \ell+2, ..., 2\ell-t^*\}$. $\mathcal{C}$ chooses $\gamma_j (0 \leqslant j \leqslant q)$ randomly from $\mathbb{Z}_p$ and sets $h_0 = g^{\gamma_0} \prod_{i \in S^* \cup \Omega^*} h_i^{-1}$, $h_j = g^{\gamma_j} g^{\alpha^{q-j+1}}$. Furthermore, $\mathcal{C}$ chooses random $\alpha' \in \mathbb{Z}_p^*$ and implicitly sets $x = \alpha' + \alpha^q$ by letting

$g_1 = g^x = g^{\alpha'} g^{\alpha^q}$, $g_2 = g^{\alpha}$. In addition, $\mathcal{C}$ randomly selects $d_2, d_3, e_1, e_2, e_3 \in \mathbb{Z}_p^*$ to compute $\delta_1 = g_2 g^{e_1}$, $\delta_2 = g_2^{d_2} g^{e_2}$,$\delta_3 = g_2^{d_3} g^{e_3}$. The challenger $\mathcal{C}$ then provides $\mathcal{A}$ the public parameters $params = (g, g_2, Z, h_0, h_1, ..., h_q, \delta_1, \delta_2, \delta_3, H)$ with $Z = e(g_1, g_2) = e(g^{\alpha'}, g^{\alpha}) e(g^{\alpha^q}, g^{\alpha})$, and H is a public collision-resistant hash function.

Phase 1: In this phase, the challenger $\mathcal{C}$ answers private key queries and decryption queries from the adversary $\mathcal{A}$.

Private Key Queries: Suppose the adversary $\mathcal{A}$ makes at most q_{K_1} extraction queries for the private key of attributes S with the restriction that $|S \cap S^*| < t^*$ (i.e., $\Upsilon^*_{t^*,S^*}(S) = 0$). Define three sets T, T', T'' in the following manner: $T = (S \cap S^*) \cup \Omega^*$, $T \subseteq T' \subseteq (S^* \cup \Omega^*)$ and $|T'| = \ell - 1$. Set $T'' = T' \cup \{0\}$. For each attribute $i \in S \cup \mathbb{U}'$, $\mathcal{C}$ random chooses an $\ell - 1$ degree polynomial $q(\cdot)$ such that $q(0) = x = \alpha' + \alpha^q$ (In fact $\mathcal{C}$ does not know the value of x.), and the private key sk_i is computed as follows for each attribute $i \in S \cup \mathbb{U}'$:

1. For $i \in T'$, i.e., $i \in (S^* \cup \Omega^*)$. $\mathcal{C}$ randomly picks $t_i, r_i' \in \mathbb{Z}_p$ and sets $q(i) = t_i$, $r_i = \alpha^i + r_i'$, then computes:

$$\begin{aligned} sk_i &= (g_2^{q(i)} (h_0 h_i)^{r_i}, g^{r_i}, h_1^{r_i}, ..., h_{i-1}^{r_i}, h_{i+1}^{r_i}, ..., h_q^{r_i}) \\ &= (g_2^{t_i} (h_0 h_i)^{r_i' + \alpha^i}, g^{r_i' + \alpha^i}, h_1^{r_i' + \alpha^i}, ..., h_{i-1}^{r_i' + \alpha^i}, h_{i+1}^{r_i' + \alpha^i}, ..., h_q^{r_i' + \alpha^i}) \\ &= (g_2^{t_i} (h_0 h_i)^{r_i'} (g^{\gamma_0} \prod_{j \in S^* \cup \Omega^*, j \neq i} h_j^{-1})^{\alpha^i}, g^{r_i'} g^{\alpha^i}, h_1^{r_i' + \alpha^i}, ..., h_{i-1}^{r_i' + \alpha^i}, h_{i+1}^{r_i' + \alpha^i}, ..., h_q^{r_i' + \alpha^i}). \end{aligned}$$

2. For $i \notin T'$, i.e., $i \notin (S^* \cup \Omega^*)$. $\mathcal{C}$ randomly selects $r_i' \in \mathbb{Z}_p$ and assigns $r_i = r_i' - \Delta_{0,T''}(i) \alpha^i$. By using the Lagrange interpolation: $q(i) = \Delta_{0,T''}(i) q(0) + \sum_{j \in T'} \Delta_{j,T''}(i) q(j)$, $\mathcal{C}$ can compute the private key:

$$\begin{aligned} sk_i &= (g_2^{q(i)} (h_0 h_i)^{r_i}, g^{r_i}, h_1^{r_i}, ..., h_{i-1}^{r_i}, h_{i+1}^{r_i}, ..., h_q^{r_i}) \\ &= (g_2^{\Delta_{0,T''}(i) q(0) + \sum_{j \in T'} \Delta_{j,T''}(i) q(j)} (h_0 h_i)^{r_i' - \Delta_{0,T''}(i) \alpha^i}), g^{r_i' - \Delta_{0,T''}(i) \alpha^i}, \\ &\qquad h_1^{r_i' - \Delta_{0,T''}(i) \alpha^i}, ..., h_{i-1}^{r_i' - \Delta_{0,T''}(i) \alpha^i}, h_{i+1}^{r_i' - \Delta_{0,T''}(i) \alpha^i}, ..., h_q^{r_i' - \Delta_{0,T''}(i) \alpha^i}) \\ &= (g_2^{\Delta_{0,T''}(i) \alpha' + \sum_{j \in T'} \Delta_{j,T''}(i) t_j} (h_0 h_i)^{r_i'} (h_0)^{-\Delta_{0,T''}(i) \alpha^i} (g^{\alpha^i})^{-\Delta_{0,T''}(i) \gamma_i}, g^{r_i' - \Delta_{0,T''}(i) \alpha^i}, \\ &\qquad h_1^{r_i' - \Delta_{0,T''}(i) \alpha^i}, ..., h_{i-1}^{r_i' - \Delta_{0,T''}(i) \alpha^i}, h_{i+1}^{r_i' - \Delta_{0,T''}(i) \alpha^i}, ..., h_q^{r_i' - \Delta_{0,T''}(i) \alpha^i}). \end{aligned}$$

Remarks: The tricky part is to simulate the sk_i values since this contains terms of the form $g^{\alpha^{q+1}}$ which is unknown to $\mathcal{C}$. The reason is that, by dividing three sets: T, T' and T'', all of these terms of $g^{\alpha^{q+1}}$ can be canceled out. Intuitively, for any attribute $i \in T'$, the term of $g^{\alpha^{q+1}}$ can be canceled out by $(h_0 h_i)^{\alpha^i}$; for any attribute $i \notin T'$, the term of $g^{\alpha^{q+1}}$ can be canceled out by $(g_2)^{q(0)} (h_i)^{\alpha^i}$. The consistence of this simulation can be assured by the Lagrange interpolation:$q(i) = \Delta_{0,T''}(i) q(0) + \sum_{j \in T'} \Delta_{j,T''}(i) q(j)$. As $(g^{\alpha}, g^{\alpha^2}, ..., g^{\alpha^q}, g^{\alpha^{q+2}}, ...,, g^{\alpha^{2q}})$ is known to $\mathcal{C}$, and $\mathcal{C}$ does not need to know $g^{\alpha^{q+1}}$ for above calculation, so the simulator can construct the private key for each attribute $i \in S \cup \mathbb{U}'$. Furthermore, the distribution of the private key is identical to that of the original scheme.

Decryption Queries: The adversary $\mathcal{A}$ submits a ciphertext $CT = (C_0, C_1, C_2, C_3, r)$ with a threshold access structure $\Upsilon_{t,S}$. The challenger $\mathcal{C}$ first computes $c = H(\Upsilon_{t,S}, C_0, C_1, C_2)$ and checks whether the ciphertext is consistent:

$$e(g, C_2) = e(C_1, h_0 \prod_{j \in S \cup \Omega} h_j), e(g, C_3) = e(C_1, \delta_1^c \delta_2^r \delta_3)$$

If one of the two equations does not hold, return $\perp$. $\mathcal{C}$ then checks whether $c + rd_2 + d_3 = 0$. Note that the probability that $c + rd_2 + d_3 = 0$ occurs is at most $1/p$. If so, the challenger $\mathcal{C}$ aborts and randomly outputs a bit; otherwise $\mathcal{C}$ outputs

$$M = C_0 \Big/ e((C_3/C_1^{ce_1 + re_2 + e_3}), g_1^{(c+rd_2+d_3)^{-1}})$$

Challenge: The adversary $\mathcal{A}$ submits two challenge messages M_0 and M_1 of equal length to the challenger $\mathcal{C}$. $\mathcal{C}$ flips a fair binary coin β, and returns an encryption of M_β. The ciphertext $CT^* = (C_0^*, C_1^*, C_2^*, C_3^*, r^*)$ is output as follows:

$$C_0^* = M_\beta T \cdot e(h, g_2^{\alpha'}), C_1^* = h, C_2^* = h^{\gamma_0}, C_3^* = h^{e_1 c^* + r^* e_2 + e_3}, r^* = -(c^* + d_3)/d_2,$$

where $c^* = H(\Upsilon^*_{t^*, S^*}, C_0^*, C_1^*, C_2^*)$.

If $\mu = 0$, then $T = e(g^{\alpha^{q+1}}, h)$, and the challenge ciphertext is a valid encryption of M_β. On the other hand, when $\mu = 1$, T is uniform and independent in $\mathbb{G}_T$, the challenger ciphertext CT is independent of β in the adversary's view.

Phase 2: $\mathcal{A}$ continues to make private key queries and decryption queries, and $\mathcal{C}$ responds similar as in **Phase 1**.

Guess: The adversary $\mathcal{A}$ will eventually output a guess β' of β. If $\beta' = \beta$, the challenger $\mathcal{C}$ then outputs $\mu' = 0$ to guess that $T = e(g^{\alpha^{q+1}}, h)$; otherwise, it outputs $\mu' = 1$ to indicate that it believes T is a random group element in $\mathbb{G}_T$.

Probability Analysis: As shown in the above, the distributions of public parameters and the private keys are identical to the real world.

If $\mu = 1$, i.e., T is a random group element, the adversary $\mathcal{A}$ gains no information about β. Therefore, we have $\Pr[\beta' \neq \beta | \mu = 1] = \Pr[\beta' = \beta | \mu = 1] = 1/2$.

If $\mu = 0$, the adversary sees an encryption of M_β, The adversary's advantage in this situation is at least ϵ. Note that the probability that $\mathcal{C}$ aborts during the simulation is at most q_D/p, where q_D is the number of decryption queries $\mathcal{A}$ can make. Therefore, we have $\Pr[\beta' = \beta | \mu = 0] = \epsilon + 1/2 - q_D/p$.

The overall advantage of the challenger $\mathcal{C}$ in the decisional q-BDHE game is

$$(1/2) \cdot \Pr[\beta' = \beta | \mu = 0] + (1/2) \cdot \Pr[\beta' = \beta | \mu = 1] - (1/2) = (\varepsilon - q_D)\ /2.$$

This completes the proof of Theorem 1. □

4.2 Performance Comparison

In this section, we compare efficiency and security among available CP-ABE schemes that have constant size ciphertexts in the literature. Comparisons are made in terms of the computation costs of encryption and decryption operations, the private key size, the ciphertext length, the flexibility of access policy, the security assumption as well as the security model.

In Table 1, let Enc., Dec. be the computation of encryption and decryption, respectively. For simplicity, the computation of multiplication over group is ignored. Let **p**, **e** be the number of pairing and exponentiation computation, respectively. We denote the number of attributes involved in the access policy as n, the number of attributes the user holds as s (used in the private key size of Table 1). Let t be the number of attributes the user has to hold in order to satisfy the access policy. Let ℓ be the number of normal attributes used in the attribute university $\mathbb{U}$. The notation $|\mathbb{G}|$ is the bit length of the element which belongs to $\mathbb{G}$.

Table 1. Comparison with other CP-ABE schemes with constant size ciphertexts

Schemes	Enc.	Dec.	Private key	Ciphertext	Expressiveness	Security
EMN+[8]	$(t+2)\mathbf{e}$	$2\mathbf{p}+2\mathbf{e}$	$2\,\|\mathbb{G}\|$	$\|\mathbb{G}_T\| + 2\,\|\mathbb{G}\|$	**AND**-gate	CPA
ZH[24]	$2\mathbf{e}$	$(2t+1)\mathbf{p}$	$(2\ell+1)\mathbb{G}$	$\|\mathbb{G}_T\| + 2\,\|\mathbb{G}\|$	**AND**-gate	CPA
CZF1[6]	$3\mathbf{e}$	$2\mathbf{p}$	$\ell\mathbb{G}$	$\|\mathbb{G}_T\| + 2\,\|\mathbb{G}\|$	**AND**-gate	CPA
CZF2[6]	$6\mathbf{e}$	$6\mathbf{p}+2\mathbf{e}$	$\ell\mathbb{G}$	$\|\mathbb{G}_T\| + 3\,\|\mathbb{G}\| + \|\mathbb{Z}_p\|$	**AND**-gate	CCA
AL[1]	$4\mathbf{e}$	$3\mathbf{p}+(n-1)\mathbf{e}$	$(2\ell+5)\mathbb{G}$	$\|\mathbb{G}_T\| + 2\,\|\mathbb{G}\|$	inner product	CPA
HLR[11]	$(n+t+1)\mathbf{e}$	$3\mathbf{p}+O(t^2)\mathbf{e}$	$(\ell+s-1)\mathbb{G}$	$\|\mathbb{G}_T\| + 2\,\|\mathbb{G}\|$	(t,n)-threshold gate	CPA
This Work1	$3\mathbf{e}$	$2\mathbf{p}+(2\ell)\mathbf{e}$	$2\ell(\ell+s)\mathbb{G}$	$\|\mathbb{G}_T\| + 2\,\|\mathbb{G}\|$	(t,n)-threshold gate	CPA
This Work2	$6\mathbf{e}$	$6\mathbf{p}+(2\ell+2)\mathbf{e}$	$2\ell(\ell+s)\mathbb{G}$	$\|\mathbb{G}_T\| + 3\,\|\mathbb{G}\| + \|\mathbb{Z}_p\|$	(t,n)-threshold gate	CCA

Note that [1] has a unique feature of being adaptively secure. However, since using the inner product as the basic tool, as is shown by Katz *et al.* in [12], these schemes in [1] can be only extended to "exact" threshold predicate with constant size ciphertexts (i.e., predicates which are true if and only if *exact* k input out of a set of d inputs are set to true), but not the "flexible" threshold predicate (i.e., predicates which are true if and only if *at least* k input out of a set of d inputs are set to true).

Compared with the scheme in [11], our scheme is efficient in the overhead of encryption operation (as encryption algorithm requires $O(n)$ exponentiations in [11] whereas 3 exponentiations are suffice in our scheme). However, for the decryption algorithm our scheme seems less efficient. Beyond that, the security proof is under a well-established decisional q-BDHE assumption in the standard model. In particular, Yamada *et al.*'s framework [22] does not apply to [11]. To the best of our knowledge, this is the first construction for CCA secure CP-ABE scheme with constant size ciphertexts that can support flexible threshold access structure in the standard model.

We admit that by embedding the whole universe of attributes in the private keys, the ciphertext size in our scheme can achieve constant, somewhat sacrificing the size of private keys to optimize the bandwidth requirements (i.e., the ciphertext size). It seems that bandwidth needs are the most important feature in the Pay-TV world or the anonymous access control, as the computational capacities of modern receivers tend to follow Moore's law in a quite natural way, while increasing the bandwidth capacities for the communication is extremely costly.

5 Conclusions

In this paper, we have presented two new CP-ABE schemes which have constant size ciphertexts and constant pairing costs for a flexible threshold access structure. Compared with Herranz *et al.*'s threshold CP-ABE scheme, our first scheme, which is CPA secure, has a better efficiency in the encryption operation. Our second scheme is the first threshold CP-ABE scheme with constant size ciphertexts that can achieve CCA security. The security of both schemes can be reduced to a well-established assumption: decisional q-BDHE assumption. Our schemes can be easily extended to admit weighted threshold decryption policies, which is a fair trade-off between expressiveness and efficiency.

Acknowledgment. This work was partially done while Aijun Ge was visiting State Key Laboratory of Information Security, Chinese Academy of Sciences. The authors would like to thank Javier Herranz and Xinyi Huang for advice and encouragement, and the anonymous reviewers of the ACISP 2012 for their helpful comments.

The work is partially supported by Strategic Priority Research Program of the Chinese Academy of Sciences, Grant No. XDA06010701, National Natural Science Foundation of China (No. F020701, No. 61100225, No. 60873261 and No.61170278) and Key Scientific and Technological Project of Henan Province (No.092101210502).

References

1. Attrapadung, N., Libert, B.: Functional Encryption for Inner Product: Achieving Constant-Size Ciphertexts with Adaptive Security or Support for Negation. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 384–402. Springer, Heidelberg (2010)
2. Attrapadung, N., Libert, B., de Panafieu, E.: Expressive Key-Policy Attribute-Based Encryption with Constant-Size Ciphertexts. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 90–108. Springer, Heidelberg (2011)
3. Boneh, D., Boyen, X., Goh, E.-J.: Hierarchical Identity Based Encryption with Constant Size Ciphertext. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Heidelberg (2005)

4. Canetti, R., Halevi, S., Katz, J.: Chosen-Ciphertext Security from Identity-Based Encryption. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 207–222. Springer, Heidelberg (2004)
5. Cheung, L., Newport, C.: Provably secure ciphertext policy ABE. In: ACM CCS 2007, pp. 456–465. ACM, New York (2007)
6. Chen, C., Zhang, Z., Feng, D.: Efficient Ciphertext Policy Attribute-Based Encryption with Constant-Size Ciphertext and Constant Computation-Cost. In: Boyen, X., Chen, X. (eds.) ProvSec 2011. LNCS, vol. 6980, pp. 84–101. Springer, Heidelberg (2011)
7. Delerablée, C., Pointcheval, D.: Dynamic Threshold Public-Key Encryption. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 317–334. Springer, Heidelberg (2008)
8. Emura, K., Miyaji, A., Nomura, A., Omote, K., Soshi, M.: A Ciphertext-Policy Attribute-Based Encryption Scheme with Constant Ciphertext Length. In: Bao, F., Li, H., Wang, G. (eds.) ISPEC 2009. LNCS, vol. 5451, pp. 13–23. Springer, Heidelberg (2009)
9. Fujisaki, E., Okamoto, T.: How to Enhance the Security of Public-Key Encryption at Minimum Cost. In: Imai, H., Zheng, Y. (eds.) PKC 1999. LNCS, vol. 1560, pp. 53–68. Springer, Heidelberg (1999)
10. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM CCS 2006, pp. 89–98. ACM, New York (2006)
11. Herranz, J., Laguillaumie, F., Ràfols, C.: Constant Size Ciphertexts in Threshold Attribute-Based Encryption. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 19–34. Springer, Heidelberg (2010)
12. Katz, J., Sahai, A., Waters, B.: Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)
13. Lai, J., Deng, R.H., Liu, S., Kou, W.: Efficient CCA-Secure PKE from Identity-Based Techniques. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 132–147. Springer, Heidelberg (2010)
14. Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010)
15. Lewko, A., Sahai, A., Waters, B.: Revocation systems with very small private keys. In: IEEE Symposium on Security and Privacy, pp. 273–285 (2010)
16. Lewko, A., Waters, B.: Decentralizing Attribute-Based Encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 568–588. Springer, Heidelberg (2011)
17. Okamoto, T., Takashima, K.: Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 191–208. Springer, Heidelberg (2010)
18. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
19. Shamir, A.: Identity-Based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
20. Waters, B.: Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009)

21. Waters, B.: Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 53–70. Springer, Heidelberg (2011)
22. Yamada, S., Attrapadung, N., Hanaoka, G., Kunihiro, N.: Generic Constructions for Chosen-Ciphertext Secure Attribute Based Encryption. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 71–89. Springer, Heidelberg (2011)
23. Zhang, R.: Tweaking TBE/IBE to PKE Transforms with Chameleon Hash Functions. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 323–339. Springer, Heidelberg (2007)
24. Zhou, Z., Huang, D.: On efficient ciphertext-policy attribute based encryption and broadcast encryption. In: ACM CCS 2010, pp. 753–755. ACM, New York (2010), the full version, http://eprint.iacr.org/2010/395

A Correctness of CP-ABE

Proof. Assuming the private key for a user with his attribute set A satisfying $\Upsilon_{t,S}$ is well-formed:

$$\begin{aligned} D_1 &= \prod_{i \in A' \cup \Omega} (a_i \prod_{j \in S \cup \Omega, j \neq i} (c_{i,j}))^{\Delta_{i,A' \cup \Omega}(0)} \\ &= \prod_{i \in A' \cup \Omega} (g_2^{q(i)} (h_0 h_i)^{r_i} \prod_{j \in S \cup \Omega, j \neq i} (h_j)^{r_i})^{\Delta_{i,A' \cup \Omega}(0)} \\ &= \prod_{i \in A' \cup \Omega} (g_2^{q(i) \Delta_{i,A' \cup \Omega}(0)} (h_0 \prod_{j \in S \cup \Omega} h_j)^{r_i \Delta_{i,A' \cup \Omega}(0)}) \\ &= g_2^{\sum_{i \in A' \cup \Omega} q(i) \Delta_{i,A' \cup \Omega}(0)} (h_0 \prod_{j \in S \cup \Omega} h_j)^{\sum_{i \in A' \cup \Omega} r_i \Delta_{i,A' \cup \Omega}(0)} \\ &= g_2^x (h_0 \prod_{j \in S \cup \Omega} h_j)^{\sum_{i \in A' \cup \Omega} r_i \Delta_{i,A' \cup \Omega}(0)} \end{aligned}$$

For the ciphertext that is correctly generated, then

$$\begin{aligned} e(C_1, D_1) &= e(g^s, g_2^x (h_0 \prod_{j \in S \cup \Omega} h_j)^{\sum_{i \in A' \cup \Omega} r_i \Delta_{i,A' \cup \Omega}(0)}) \\ &= e(g_1, g_2)^s \cdot e((h_0 \prod_{j \in S \cup \Omega} h_j)^s, g^{\sum_{i \in A' \cup \Omega} r_i \Delta_{i,A' \cup \Omega}(0)}) \\ &= Z^s \cdot e(C_2, D_2). \end{aligned}$$

It is easily verified that $M = C_0 \cdot e(C_2, D_2) / e(C_1, D_1)$.

Fully Private Revocable Predicate Encryption

Juan Manuel González Nieto[1], Mark Manulis[2], and Dongdong Sun[1]

[1] Queensland University of Technology, Brisbane QLD 4001, Australia
j.gonzaleznieto@qut.edu.au, dd.sun@student.qut.edu.au
[2] University of Surrey, Guildford, United Kingdom
mark@manulis.eu

Abstract. We introduce the concept of *Revocable Predicate Encryption (RPE)*, which extends current predicate encryption setting with revocation support: private keys can be used to decrypt an RPE ciphertext only if they match the decryption policy (defined via attributes encoded into the ciphertext and predicates associated with private keys) and were not revoked by the time the ciphertext was created.

We formalize the notion of attribute hiding in the presence of revocation and propose an RPE scheme, called AH-RPE, which achieves attribute-hiding under the Decision Linear assumption in the standard model.

We then present a stronger privacy notion, termed *full hiding*, which further cares about privacy of revoked users. We propose another RPE scheme, called FH-RPE, that adopts the Subset Cover Framework and offers full hiding under the Decision Linear assumption in the standard model. The scheme offers very flexible privacy-preserving access control to encrypted data and can be used in *sender-local revocation* scenarios.

Keywords: Predicate Encryption, Revocation, Attribute Hiding, Full Hiding, Sender-Local Revocation.

1 Introduction

Functional Encryption. In recent years, asymmetric encryption has experienced a paradigm shift from encryption of secret messages for particular recipients (Public Key Encryption or Identity-Based Encryption) towards more flexible encryption mechanisms, which offer manifold forms of access control to encrypted data. These mechanisms rely on arbitrary *functional* relationships between policies and attributes encoded in ciphertexts and recipients' decryption keys. Functional Encryption has emerged from Identity-Based Encryption techniques [6,19], and encompasses novel concepts such as Attribute-Based Encryption [9,18], Hidden-Vector Encryption [7], and Predicate-Based Encryption [10,11,14,15,20–22]. At a high level these schemes implement the idea of creating ciphertexts without prior knowledge of potential recipients. The success of message recovery depends usually on some relation, which is implicitly evaluated through the decryption procedure, on input the information encoded in the ciphertext and information contributed by the private key.

Predicate Encryption. In this work we focus on the notion of Predicate Encryption (PE), formalized by Katz, Sahai, and Waters [10], and further studied in [11,14,15,20,22]. In PE schemes private keys of users are associated with *predicates* f and ciphertexts are bound to *attributes* a. The decryption procedure is successful if and only if $f(a) = 1$. If this relation is not satisfied then no information about the plaintext is leaked. In contrast to Attribute-Based Encryption, which also states this requirement on the security of the decryption procedure, PE schemes offer privacy of attributes that legitimate recipients of PE ciphertexts must possess, that is PE ciphertexts ensure *attribute hiding* in that they do not leak any information about a for which the condition $f(a) = 1$ would be satisfied. Concrete constructions of PE schemes typically focus on the realization of certain types of predicates f. In their seminal work, Katz, Sahai, and Waters [10] introduced PE schemes supporting Inner-Product Encryption (IPE), i.e. vector $\overrightarrow{y}$ represents attributes and vector $\overrightarrow{x}$ determines the predicate $f_{\overrightarrow{x}}$ such that $f_{\overrightarrow{x}}(\overrightarrow{y}) = 1$ iff $\overrightarrow{x} \cdot \overrightarrow{y} = 0$ ($\overrightarrow{x} \cdot \overrightarrow{y}$ denotes the inner product of vectors $\overrightarrow{x}$ and $\overrightarrow{y}$ over a field or ring). It has been shown that IPE can be leveraged to evaluate a wide class of predicates such as conjunctions or disjunctions of equality tests, conjunctions of comparison or subset tests, and more generally, arbitrary CNF or DNF formulae.

Revocation in Functional Encryption. The revocation problem in FE turns out to be more subtle than in previous encryption paradigms, e.g. in comparison to CRL-based revocation mechanisms used in traditional PKE schemes (within public key infrastructures) [1,8,13] or to the time-based revocation approach suggested by Boneh and Franklin [6] for IBE schemes, where the identities of receivers are linked to time periods and unrevoked users must be in possession of up-to-date private keys, obtained from the Private Key Generator (PKG). The revocation problem in FE is apparent in that FE ciphertexts are encrypted for predicates f that can possibly be satisfied by multiple recipients, all in possession of suitable attributes a. Using time-based revocation for users' attributes is inappropriate here for several reasons: First, a user may be in possession of several attributes and if time periods for all attributes in the system are not synchronized then unrevoked users would have to update their private keys whenever any of their attributes expires. Second, even if time periods are synchronized then the problem with scalability still remains. Indeed, the PKG would have to be regularly contacted by all unrevoked users in the system to obtain updates for their private keys. This would require online presence of the PKG, establishment of secure channels between the PKG and each user for the transmission of updated private keys, and authentication of users towards the PKG to prove eligibility with regard to the update procedure. The amount of work performed by the PKG is then linear in the number of (unrevoked) users and attributes available in the system.

A more efficient approach for handling revocation in IBE systems was suggested by Boldyreva, Goyal, and Kumar (BGK) [4], where the PKG on each time period publishes some update information that is then used by unrevoked users to update their private keys locally. The amount of work performed by

PKG is logarithmic and, more importantly, no online communication between the PKG and unrevoked users is required. The approach from [4] could also be applied to ABE systems, in which case, however, it would result in a significant limitation — while in IBE systems revoking user identities is sufficient, revoking attributes in ABE systems would implicitly revoke private keys of all users with those attributes. That is revocation of users (which is possible with the time-based approach of Boneh and Franklin [6] when applied to ABE systems) would no longer be possible with the BGK approach. Another limitation of the BGK approach is that unrevoked users still have to update their private keys for each time period.

To alleviate this limitation, Attrapadung and Imai [3] suggested another way for revocation in ABE schemes: Instead of enforcing revocation via an authority, the revocation is carried out by the senders directly, i.e., the senders encrypt a message under a normal attribute set, as well as a revocation list. Each user's private key has an associated policy and some unique identifier. A private key can be used to decrypt the ciphertext if the attributes in the ciphertext satisfy the policy associated with the key *and* the identifier of the key is not contained in the revocation list encoded into the ciphertext. This method solves the mentioned problem behind the BGK approach, namely each user's private key can now be issued by PKG once and need not be updated thereafter. Later on, Attrapadung and Imai [2] proposed another system by combining techniques from [4] with their previous work from [3], which inherits the advantages of both approaches.

Revocation in PE Schemes and Privacy. The different ABE revocation techniques mentioned above, aside from their scalability issues, are only partially applicable to PE schemes due to the distinguished attribute-hiding property of the latter. In particular, care should be taken to ensure that by introducing revocation to a PE system this privacy property is preserved. To the best of our knowledge, revocation in PE schemes has not been investigated so far and it is not clear whether revocation introduces further privacy challenges, in addition to the challenge of preserving their basic attribute-hiding property. We observe that additional privacy problems may arise in scenarios, where revocation is performed for individual private keys. For example, in the revocable ABE scheme of Attrapadung and Imai [3], each sender builds a revocation list on-the-fly, using unique identifiers of users' private keys, and encodes this list into the ciphertext. However, a close inspection of the scheme shows that ciphertexts reveal information about the encoded key identifiers and by this leak information about the revoked users. In this work we explore the concept of privacy-preserving revocation in PE schemes. Our contributions are detailed in the following.

1.1 Our Contributions

We formalize the concept of **Revocable Predicate Encryption (RPE)** and propose two RPE schemes allowing for efficient revocation of private keys.

Attribute-Hiding RPE Scheme. Our first scheme, termed AH-RPE, offers attribute-hiding, which is the standard PE property (and further implies payload-hiding used in the context of ABE). The revocation concept behind AH-RPE uses revocation lists (RL) and is mostly suitable for applications where revocation management is handled centrally by the PKG. It is assumed that senders obtain up-to-date RL published by the PKG prior to encryption. The attribute-hiding property of our AH-RPE scheme is proven against adaptive adversaries in the standard model under the established DLIN assumption. The AH-RPE scheme has *constant-size* private and public keys while the length of its ciphertexts remains linear in the number of *revoked* keys.

Full-Hiding RPE Scheme. Our second scheme, termed FH-RPE, offers even stronger privacy guarantees that ensure no information about revoked users is leaked from a given ciphertext and is a natural extension in the context of PE that cares about privacy. Our FH-RPE scheme can be used in applications where senders may freely decide to exclude certain key holders from running a successful decryption operation, even if private keys of those holder match the ciphertext policy. Such **sender-local revocation (SLR)**, as previously addressed in [2] for ABE schemes, allows for more flexible forms of access control to PE plaintexts and the requirement of full-hiding keeps revoked recipients undisclosed. The full-hiding property of FH-RPE also relies on the DLIN assumption and the length of keys and ciphertexts becomes *logarithmic* in the number of decryption keys.

Techniques. Our RPE schemes are based on the Dual System Encryption of Waters [23] and the Dual Pairing Vector Spaces (DPVS) of Okamoto and Takashima [14]. Our AH-RPE scheme deploys the revocation system of Lewko, Sahai and Waters [12], introduced originally for public-key broadcast encryption, and modified here for an integration with the (payload-hiding) FE scheme of Okamoto and Takashima [15] in a way that achieves attribute-hiding by further using some techniques underlying the PE scheme by Lewko et al. [11]. Our FH-RPE scheme is obtained from Okamoto and Takashima [15] and Lewko et al. [11] in a more direct way. The logarithmic complexity of our FH-RPE scheme is due to the use of the **complete-subtree** technique by Naor et al. [13], whose integration preserving the full-hiding property was a challenge. In order to prove security of our RPE schemes we utilize the modular approach from Okamoto and Takashima [15] that breaks the proof down into several higher-level (artificially looking) assumptions and proves them to be secure under the DLIN assumption.

2 Dual Pairing Vector Spaces and Assumptions

Let $\mathcal{G}_{\mathsf{bpg}}$ be an algorithm that takes as input a security parameter 1^λ and outputs a description of the symmetric bilinear group setting $(q, \mathbb{G}, \mathbb{G}_T, G, e)$ where q is a prime, $\mathbb{G}$ and $\mathbb{G}_T$ are two cyclic groups of order q, G is the generator of $\mathbb{G}$, e is a non-degenerate bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, i.e., $e(sG, tG) = e(G, G)^{st}$ and $e(G, G) \neq 1$.

Let $\mathbb{V} = \overbrace{\mathbb{G} \times \cdots \times \mathbb{G}}^{N}$ be a *vector space* and each element in $\mathbb{V}$ be expressed by an *N-dimensional vector* $\boldsymbol{x} = (x_1 G, \ldots, x_N G)$ $(x_i \in \mathbb{F}_q$ for $i = 1, \ldots, N)$.

The *canonical base* $\mathbb{A}$ of $\mathbb{V}$ is $\mathbb{A} = (\boldsymbol{a}_1, \dots, \boldsymbol{a}_N)$, where $\boldsymbol{a}_1 = (G, 0, \dots, 0)$, $\boldsymbol{a}_2 = (0, G, 0, \dots, 0), \dots, \boldsymbol{a}_N = (0, \dots, 0, G)$. Given two vectors $\boldsymbol{x} = (x_1G, \dots, x_NG) = x_1\boldsymbol{a}_1 + \cdots + x_N\boldsymbol{a}_N \in \mathbb{V}$ and $\boldsymbol{y} = (y_1G, \dots, y_NG) = y_1\boldsymbol{a}_1 + \cdots + y_N\boldsymbol{a}_N \in \mathbb{V}$, where $\overrightarrow{x} = (x_1, \dots, x_N)$ and $\overrightarrow{y} = (y_1, \dots, y_N)$, the pairing operation is defined as $e(\boldsymbol{x}, \boldsymbol{y}) = \prod_{i=1}^{N} e(x_iG, y_iG) = e(G, G)^{\sum_{i=1}^{N} x_i y_i} = g_T^{\overrightarrow{x}\,\overrightarrow{y}} \in \mathbb{G}_T$.

Let $\mathbb{B} = (\boldsymbol{b}_1, \dots, \boldsymbol{b}_N)$ be the basis of $\mathbb{V}$ which is obtained from the canonical basis $\mathbb{A}$ using a uniformly chosen linear transformation, $\Lambda = (\lambda_{i,j}) \overset{U}{\leftarrow} GL(N, \mathbb{F}_q)$ ($GL(N, \mathbb{F}_q)$ creates a matrix of size $N \times N$ in which each element is uniformly selected from $\mathbb{F}_q$), such that $\boldsymbol{b}_i = \sum_{j=1}^{N} \lambda_{i,j}\boldsymbol{a}_j$, for $i = 1, \dots, N$. Similarly, $\mathbb{B}^* = (\boldsymbol{b}_1^*, \dots, \boldsymbol{b}_N^*)$ of $\mathbb{V}$ is also obtained from $\mathbb{A}$, such that $\mu_{i,j} = (\Lambda^T)^{-1}$, $\boldsymbol{b}_i^* = \sum_{j=1}^{N} \mu_{i,j}\boldsymbol{a}_j$, for $i = 1, \dots, N$. It can be shown that $e(\boldsymbol{b}_i, \boldsymbol{b}_j^*) = g_T^{\delta_{i,j}}$, where $\delta_{i,j} = 1$ if $i = j$, and $\delta_{i,j} = 0$ if $i \neq j$. $\mathbb{B}$ and $\mathbb{B}^*$ are thus dual orthonormal bases of $\mathbb{V}$.

Definition 1 (Dual Pairing Vector Space (DPVS) [14]). *Let $(q, \mathbb{G}, \mathbb{G}_T, G, e)$ be a symmetric pairing group. A* Dual Pairing Vector Space $(q, \mathbb{V}, \mathbb{G}_T, \mathbb{A}, e)$, *generated by an algorithm denoted $\mathcal{G}_{\mathsf{dpvs}}$, is a tuple of a prime q, N-dimensional vector space $\mathbb{V}$ over $\mathbb{F}_q$, a cyclic group $\mathbb{G}_T$ of order q, canonical base $\mathbb{A} = (\boldsymbol{a}_1, \dots, \boldsymbol{a}_N)$ of $\mathbb{V}$, and pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ that satisfy the following conditions:*

1. Non-degenerate bilinear pairing: *There exists a polynomial-time computable non-degenerate bilinear pairing $e(\boldsymbol{x}, \boldsymbol{y}) = \prod_{i=1}^{N} e(G_i, H_i)$ where $\boldsymbol{x} = (G_1, \dots, G_N) \in \mathbb{V}$ and $\boldsymbol{y} = (H_1, \dots, H_N) \in \mathbb{V}$. This is non-degenerate bilinear i.e., $e(s\boldsymbol{x}, t\boldsymbol{y}) = e(\boldsymbol{x}, \boldsymbol{y})^{st}$ and if $e(\boldsymbol{x}, \boldsymbol{y}) = 1$ for all $\boldsymbol{y} \in \mathbb{V}$, then $\boldsymbol{x} = \boldsymbol{0}$.*
2. Dual orthonormal bases: *$\mathbb{A}$ and e satisfy that $e(\boldsymbol{a_i}, \boldsymbol{a_j}) = g_T^{\delta_{i,j}}$ for all i and j, where $\delta_{i,j} = 1$ if $i = j$, and 0 otherwise, and $g_T \neq 1 \in \mathbb{G}_T$.*
3. Distortion maps: *Linear transformations $\phi_{i,j}$ on $\mathbb{V}$ s.t. $\phi_{i,j}(\boldsymbol{a}_j) = \boldsymbol{a}_i$ and $\phi_{i,j}(\boldsymbol{a}_k) = \boldsymbol{0}$ if $k \neq j$ are polynomial-time computable. We call $\phi_{i,j}$ "distortion maps".*

In our schemes, we will use the following probabilistic generator $\mathcal{G}_{\mathsf{ob}}$ for dual orthonormal bases:

```
$\mathcal{G}_{\mathsf{ob}}(1^\lambda, \overrightarrow{n} = (d; n_1, \dots, n_d))$:
  $\mathsf{param}_{\mathbb{G}} = (q, \mathbb{G}, \mathbb{G}_T, G, e) \overset{\mathsf{R}}{\leftarrow} \mathcal{G}_{\mathsf{bpg}}(1^\lambda)$, $\psi \overset{\mathsf{U}}{\leftarrow} \mathbb{F}_q^\times, N_0 = 5, N_l = 3n_l + 1$ for $l = 1, \dots, d$;
  For $l = 0, \dots, d$:
    $\mathsf{param}_{\mathbb{V}_l} = (q, \mathbb{V}_l, \mathbb{G}_T, \mathbb{A}_l, e) \overset{\mathsf{R}}{\leftarrow} \mathcal{G}_{\mathsf{dpvs}}(1^\lambda, N_l, \mathsf{param}_{\mathbb{G}})$,
    $\Lambda^{(l)} = (\lambda_{i,j}^{(l)}) \overset{\mathsf{U}}{\leftarrow} GL(N_l, \mathbb{F}_q)$, $(\mu_{i,j}^{(l)}) = \psi \cdot ({\Lambda^{(l)}}^T)^{-1}$,
    ${\boldsymbol{b}_i}^{(l)} = \sum_{j=1}^{N_l} \lambda_{i,j}^{(l)}\boldsymbol{a}_j^{(l)}$ for $i = 1, \dots, N_l, \mathbb{B}^{(l)} = (\boldsymbol{b}_1^{(l)}, \dots, \boldsymbol{b}_{N_l}^{(l)})$,
    $\boldsymbol{b}_i^{*(l)} = \sum_{j=1}^{N_l} \mu_{i,j}^{(l)}\boldsymbol{a}_j^{(l)}$ for $i = 1, \dots, N_l, \mathbb{B}^{*(l)} = (\boldsymbol{b}_1^{*(l)}, \dots, \boldsymbol{b}_{N_l}^{*(l)})$,
  $g_T = e(G, G)^\psi$, $\mathsf{param}_{\overrightarrow{n}} = (\{\mathsf{param}_{\mathbb{V}_l}\}_{l=0,\dots,d}, g_T)$,
Output $(\mathsf{param}_{\overrightarrow{n}}, \{\mathbb{B}^{(l)}, \mathbb{B}^{*(l)}\}_{l=0,\dots,d})$. (Note that $g_T = e({\boldsymbol{b}_i}^{(l)}, \boldsymbol{b}_i^{*(l)})$ for $l = 0, \dots, d$; $i = 1, \dots, N_l$.)
```

Definition 2 (Decisional Linear Assumption (DLIN) [5]). *The DLIN problem is to decide on bit* $\beta \in \{0,1\}$, *given* $(\mathsf{param}_{\mathbb{G}}, G, aG, bG, acG, bdG, Y_\beta) \xleftarrow{\mathsf{R}} \mathcal{G}_\beta^{\mathrm{DLIN}}(1^\lambda)$, *where the algorithm* $\mathcal{G}_\beta^{\mathrm{DLIN}}(1^\lambda)$:

$$\begin{aligned}&\mathsf{param}_{\mathbb{G}} = (q, \mathbb{G}, \mathbb{G}_T, G, e) \xleftarrow{\mathsf{R}} \mathcal{G}_{\mathsf{bpg}}(1^\lambda), a, b, c, d \xleftarrow{\mathsf{U}} \mathbb{F}_q,\\ &Y_0 = (c+d)G,\ Y_1 \xleftarrow{\mathsf{U}} \mathbb{G},\ \beta \xleftarrow{\mathsf{U}} \{0,1\}.\\ &\text{Output } (\mathsf{param}_{\mathbb{G}}, G, aG, bG, acG, bdG, Y_\beta).\end{aligned}$$

The advantage $\mathsf{Adv}_{\mathcal{D}}^{\mathrm{DLIN}}(\lambda)$ *of a probabilistic polynomial-time DLIN solver* $\mathcal{D}$ *is defined as*

$$\left| \mathsf{Pr}\left[\mathcal{D}(1^\lambda, \varpi) \to 1 \;\middle|\; \varpi \xleftarrow{\mathsf{R}} \mathcal{G}_0^{\mathrm{DLIN}}(1^\lambda)\right] - \mathsf{Pr}\left[\mathcal{D}(1^\lambda, \varpi) \to 1 \;\middle|\; \varpi \xleftarrow{\mathsf{R}} \mathcal{G}_1^{\mathrm{DLIN}}(1^\lambda)\right] \right|.$$

The DLIN assumption states that for any probabilistic polynomial-time solver $\mathcal{D}$, *the advantage* $\mathsf{Adv}_{\mathcal{D}}^{\mathrm{DLIN}}(\lambda)$ *is negligible in* λ.

3 Revocable Predicate Encryption: Model and Definitions

In predicate encryption for the inner-product relation, an attribute is expressed as a vector $\vec{y} \in \mathbb{F}_q^n \setminus \{\vec{0}\}$ and a predicate $f_{\vec{x}}$ is associated with a vector $\vec{x} \in \mathbb{F}_q^n \setminus \{\vec{0}\}$, where $f_{\vec{x}}(\vec{y}) = 1$, iff $\vec{y} \cdot \vec{x} = 0$. Let $\mathbb{A} = \mathbb{F}_q^n \setminus \{\vec{0}\}$ be the attribute space, and $\mathbb{P} = \{f_{\vec{x}} | \vec{x} \in \mathbb{F}_q^n \setminus \{\vec{0}\}\}$ be the predicate space. We assume that indexes are in the set $\Gamma = \{1, \ldots, N\}$, where N is the number of keys in the system. In our definitions and schemes, we assume that attribute vector, $\vec{y} = (y_1, \ldots, y_{n_1})$, is normalized such that $y_1 = 1$ (If $\vec{y}$ is not normalized, change it to a normalized one by $(1/y_1) \cdot \vec{y}$, assuming that y_1 is non-zero). $\vec{e}_i^{(k)}$ is the canonical basis vector $(\overbrace{0, \ldots, 0}^{i-1}, 1, \overbrace{0, \ldots, 0}^{n_k - i}) \in \mathbb{F}_q^{n_k}$ for $k = 1, 2$ and $i = 1, \ldots, n_k$.

3.1 Syntax

Definition 3. *A* Revocable Predicate Encryption (RPE) *comprises of four algorithms* (Setup, GenKey, Encrypt, Decrypt) *and has associated attribute space* $\mathbb{A}$, *predicate space* $\mathbb{P}$ *and index space* Γ.

Setup$(1^\lambda, \Delta)$ *The* Setup *algorithm takes as input a security parameter* 1^λ *and format* Δ *of attribute and index. It outputs a public key* PK, *a master secret key* MSK, *and a state information* S.

GenKey$(MSK, S, \vec{x})$ *The* GenKey *algorithm takes as input a master secret key* MSK, *a state information* S, *and a predicate vector* $\vec{x}$. *It outputs an updated state* S *and a secret key* $\boldsymbol{k}^*_{\vec{x},I}$, *where* $I \in \Gamma$ *denotes the associated index of the key and is included in the key.*

Encrypt$(PK, L, \vec{y}, M)$ *The* Encrypt *algorithm takes as input a public key* PK, *a revocation list* $L \subseteq \Gamma$, *an attribute vector* $\vec{y}$, *and a message* M *in some associated message space. It outputs a ciphertext* C.

$\mathsf{Decrypt}(C, \boldsymbol{k}^*_{\overrightarrow{x},I})$ *The* Decrypt *algorithm takes as input a ciphertext C and a secret key $\boldsymbol{k}^*_{\overrightarrow{x},I}$. It outputs either a message M or the distinguished symbol $\perp$.*

Correctness. *The correctness property of the schemes says that for all PK and MSK output by* Setup *algorithm, all predicate $f_{\overrightarrow{x}} \in \mathbb{P}$, all message M, all attribute $\overrightarrow{y} \in \mathbb{A}$, and all possible valid state information S output by* Setup *or* GenKey *algorithm, if the key $\boldsymbol{k}^*_{\overrightarrow{x},I}$ was not revoked, i.e., $I \notin L$, then for correctly generated $\boldsymbol{k}^*_{\overrightarrow{x},I} \overset{R}{\leftarrow} \mathsf{GenKey}(MSK, S, \overrightarrow{x})$ and $C \overset{R}{\leftarrow} \mathsf{Encrypt}(PK, L, \overrightarrow{y}, M)$:*

- *If $f_{\overrightarrow{x}}(\overrightarrow{y}) = 1$ then $\mathsf{Decrypt}(C, \boldsymbol{k}^*_{\overrightarrow{x},I}) = M$.*
- *If $f_{\overrightarrow{x}}(\overrightarrow{y}) = 0$ then $\mathsf{Decrypt}(C, \boldsymbol{k}^*_{\overrightarrow{x},I}) = \perp$ with all but negligible probability.*

3.2 Definitions of Attribute-Hiding and Full-Hiding in RPE

In the definition of attribute hiding for RPE schemes we additionally allow the adversary to specify the revocation list used to create the challenge ciphertext but we do not require ciphertexts to hide information about revoked key indices. This definition suits applications where revocation lists are managed and published by the master authority.

Definition 4 (Attribute-Hiding RPE). *An* RPE *scheme is adaptively attribute hiding against chosen plaintext attacks if for all PPT adversaries $\mathcal{A}$, the advantage $\mathsf{Adv}^{\mathsf{AH}}_{\mathcal{A},\mathsf{RPE}}(\lambda)$ in the following game is negligible in the security parameter λ:*

Setup. *A challenger $\mathcal{C}$ runs the* Setup *algorithm to generate a public key PK, a master secret key MSK, and S. PK is given to $\mathcal{A}$.*

Query phase 1. *$\mathcal{A}$ adaptively makes a polynomial number of* GenKey *queries: $\mathcal{A}$ produces a predicate $\overrightarrow{x}$, $\mathcal{C}$ computes the key $\boldsymbol{k}^*_{\overrightarrow{x},I} \overset{R}{\leftarrow} \mathsf{GenKey}(MSK, S, \overrightarrow{x})$ associated with an index I, and gives it to $\mathcal{A}$.*

Challenge. *$\mathcal{A}$ outputs challenge attribute vectors $(\overrightarrow{y}^{(0)}, \overrightarrow{y}^{(1)})$, challenge plaintexts $(M^{(0)}, M^{(1)})$, and a revocation list L, subject to one of the following restrictions for each queried key $\boldsymbol{k}^*_{\overrightarrow{x},I}$:*
1. *$I \in L$, or*
2. *$I \notin L$ and $f_{\overrightarrow{x}}(\overrightarrow{y}^{(0)}) = f_{\overrightarrow{x}}(\overrightarrow{y}^{(1)}) = 0$.*

$\mathcal{C}$ flips a random bit b. If $b = 0$ then $\mathcal{A}$ is given $C = \mathsf{Encrypt}(PK, L, \overrightarrow{y}^{(0)}, M^{(0)})$. If $b = 1$ then $\mathcal{A}$ is given $C = \mathsf{Encrypt}(PK, L, \overrightarrow{y}^{(1)}, M^{(1)})$.

Query phase 2. *Repeat the* **Query phase 1** *subject to the restrictions as in the challenge phase.*

Guess. *$\mathcal{A}$ outputs a guess b' of b, and succeeds if $b' = b$.*

The advantage of $\mathcal{A}$ is defined to be $\mathsf{Adv}^{\mathsf{AH}}_{\mathcal{A},\mathsf{RPE}}(\lambda) = |\Pr[b = b'] - 1/2|$.

In addition to attribute hiding, our notion of *full hiding* ensures that ciphertexts do not leak any information about the revoked indexes. This strong privacy goal is essential when key indexes can be linked to users and whenever senders wish to exclude some users from decrypting PE ciphertexts — this concept of sender-local revocation (SLR) [2] allows senders to compose revocation lists (per ciphertext) during the encryption process and by this flexibly refine access control to encrypted data.

Definition 5 (Full-Hiding RPE). *An* RPE *scheme is adaptively full hiding against chosen plaintext attacks if for all PPT adversaries $\mathcal{A}$, the advantage* $\mathsf{Adv}^{\mathsf{FH}}_{\mathcal{A},\mathsf{RPE}}(\lambda)$ *in the following game is negligible in the security parameter λ:*

Setup. *A challenger $\mathcal{C}$ runs the* Setup *algorithm to generate a public key PK, a master secret key MSK, and S. PK is given to $\mathcal{A}$.*

Query phase 1. *$\mathcal{A}$ adaptively makes a polynomial number of* GenKey *queries: $\mathcal{A}$ produces a predicate $\overrightarrow{x}$, $\mathcal{C}$ computes the key $\boldsymbol{k}^*_{\overrightarrow{x},I} \xleftarrow{R} \mathsf{GenKey}(MSK, S, \overrightarrow{x})$ associated with an index I, and gives it to $\mathcal{A}$.*

Challenge. *$\mathcal{A}$ outputs challenge attribute vectors $(\overrightarrow{y}^{(0)}, \overrightarrow{y}^{(1)})$, challenge revocation lists $(L^{(0)}, L^{(1)})$, and challenge plaintexts $(M^{(0)}, M^{(1)})$, subject to one of the following restrictions for each queried key $\boldsymbol{k}^*_{\overrightarrow{x},I}$:*

1. $f_{\overrightarrow{x}}(\overrightarrow{y}^{(0)}) = f_{\overrightarrow{x}}(\overrightarrow{y}^{(1)}) = 0$
2. $f_{\overrightarrow{x}}(\overrightarrow{y}^{(0)}) = f_{\overrightarrow{x}}(\overrightarrow{y}^{(1)}) = 1$ *and* $(I \in L^{(0)} \wedge I \in L^{(1)})$
3. $(f_{\overrightarrow{x}}(\overrightarrow{y}^{(0)}) = 1 \wedge f_{\overrightarrow{x}}(\overrightarrow{y}^{(1)}) = 0)$ *and* $I \in L^{(0)}$
4. $(f_{\overrightarrow{x}}(\overrightarrow{y}^{(0)}) = 0 \wedge f_{\overrightarrow{x}}(\overrightarrow{y}^{(1)}) = 1)$ *and* $I \in L^{(1)}$.

$\mathcal{C}$ flips a random coin b. If $b = 0$ then $\mathcal{A}$ is given $C = \mathsf{Encrypt}(PK, L^{(0)}, \overrightarrow{y}^{(0)}, M^{(0)})$. If $b = 1$ then $\mathcal{A}$ is given $C = \mathsf{Encrypt}(PK, L^{(1)}, \overrightarrow{y}^{(1)}, M^{(1)})$.

Query phase 2. *Repeat the* **Query phase 1** *subject to the restrictions as in the challenge phase.*

Guess. *$\mathcal{A}$ outputs a guess b' of b, and succeeds if $b' = b$.*

The advantage of $\mathcal{A}$ is defined to be $\mathsf{Adv}^{\mathsf{FH}}_{\mathcal{A},\mathsf{RPE}}(\lambda) = |\mathsf{Pr}[b = b'] - 1/2|$.

Remark 1. In Definition 5, adversary $\mathcal{A}$ is not allowed to ask a key query for an index I and a predicate $\overrightarrow{x}$ such that $I \notin L^{(b)}$ and $f_{\overrightarrow{x}}(\overrightarrow{y}^{(b)}) = 1$ for some $b \in \{0, 1\}$, i.e., the queried key is not allowed to decrypt the challenge ciphertext. Recently, Okamoto and Takashima [17] proposed a PE (HPE) which allow such key query, provided that $M^{(0)} = M^{(1)}$. The technique of Okamoto and Takashima [17] can be applied in our scheme to achieve strong security.

Remark 2. Definitions 4 and 5 can be extended to capture chosen-ciphertext attacks (CCA) by allowing decryption queries (for all but the challenge ciphertext). The advantage of $\mathcal{A}$ in such CCA game is defined to be $\mathsf{Adv}^{\mathsf{X\text{-}CCA}}_{\mathcal{A},\mathsf{RPE}}(\lambda) = |\mathsf{Pr}[b = b'] - 1/2|$, where $X \in \{\mathsf{AH}, \mathsf{FH}\}$. One could also define relaxed selective security, where the adversary is required to specify the challenge attributes and the revocation list in advance (before obtaining public key PK).

4 An RPE Scheme with Attribute Hiding (AH-RPE)

Our first RPE scheme is attribute-hiding. It offers short, constant-size public and private keys. The size of its ciphertexts is linear in the number of revoked keys (which is small relative to the total number of users).

In its construction we adopt the "two equation" technique from the public broadcast encryption scheme by Lewko, Sahai and Waters (LSW) [12], where a secret value required for the decryption is broken into secret shares (one share for each revoked index) such that any unrevoked private key can be used to reconstruct the secret. We show how to use this approach with the FE scheme by Okamoto and Takashima [15]: at a high level, every private key is associated with some predicate $\overrightarrow{x}$ and an index I, the ciphertext on some message M is produced using attributes $\overrightarrow{y}$ and the set of indexes $\{I_1, \ldots, I_r\}$ corresponding to r revoked keys. A private key can decrypt the ciphertext if its predicate evaluates to true, provided that its index $I \neq I_i$ for all $i \in [1, r]$. We achieve this functionality using non-zero inner products amongst the attribute and predicate vectors.

We now give detailed specification of our AH-RPE scheme:

$\mathsf{Setup}\big(1^\lambda, \Delta = (\overrightarrow{n} = (2; n_1, n_2 = 2), N)\big)$: Perform the following computations: $(\mathsf{param}_{\overrightarrow{n}}, \mathbb{B}^{(0)}, \mathbb{B}^{*(0)}, \mathbb{B}^{(1)}, \mathbb{B}^{*(1)}, \mathbb{B}^{(2)}, \mathbb{B}^{*(2)}) \xleftarrow{R} \mathcal{G}_{\mathsf{ob}}(1^\lambda, \overrightarrow{n})$,

$$\widetilde{\mathbb{B}}^{(0)} = (\boldsymbol{b}_1^{(0)}, \boldsymbol{b}_3^{(0)}, \boldsymbol{b}_5^{(0)}),\ \widetilde{\mathbb{B}}^{(1)} = (\boldsymbol{b}_1^{(1)}, \ldots, \boldsymbol{b}_{n_1}^{(1)}, \boldsymbol{b}_{3n_1+1}^{(1)}),\ \widetilde{\mathbb{B}}^{(2)} = (\boldsymbol{b}_1^{(2)}, \boldsymbol{b}_2^{(2)}, \boldsymbol{b}_7^{(2)}),$$

$$\widetilde{\mathbb{B}}^{*(0)} = (\boldsymbol{b}_1^{*(0)}, \boldsymbol{b}_3^{*(0)}, \boldsymbol{b}_4^{*(0)}),\ \widetilde{\mathbb{B}}^{*(1)} = (\boldsymbol{b}_1^{*(1)}, \ldots, \boldsymbol{b}_{n_1}^{*(1)}, \boldsymbol{b}_{2n_1+1}^{*(1)}, \ldots, \boldsymbol{b}_{3n_1}^{*(1)}),$$
$$\widetilde{\mathbb{B}}^{*(2)} = (\boldsymbol{b}_1^{*(2)}, \boldsymbol{b}_2^{*(2)}, \boldsymbol{b}_5^{*(2)}, \boldsymbol{b}_6^{*(2)}).$$

Let S denote the (initially empty) state information on the so far assigned indices I. The algorithm outputs the public key $PK = \big(1^\lambda, N, \mathsf{param}_{\overrightarrow{n}}, \{\widetilde{\mathbb{B}}^{(k)}\}_{k\in\{0,1,2\}}\big)$, the master secret key $MSK = \big(\{\widetilde{\mathbb{B}}^{*(k)}\}_{k\in\{0,1,2\}}\big)$, and the state information S.

$\mathsf{GenKey}(MSK, S, \overrightarrow{x} = (x_1, \ldots, x_{n_1}) \in \mathbb{F}_q^{n_1} \setminus \{\overrightarrow{0}\})$: Pick s, η, β, η_1, $\ldots$, η_{n_1}, ρ_1, and ρ_2 uniformly at random from $\mathbb{F}_q$ and $s_1, s_2 \xleftarrow{\mathsf{U}} \mathbb{F}_q$ such that $s = s_1 + s_2$. Choose index $I \xleftarrow{\mathsf{U}} \Gamma$ such that $I \notin S$; set $S = S \cup \{I\}$ and compute:

$$\boldsymbol{k}_0 = (-s, 0, 1, \eta, 0)_{\mathbb{B}^{*(0)}},$$
$$\boldsymbol{k}_1 = (\overbrace{s_1 \overrightarrow{e}_1^{(1)} + \beta\overrightarrow{x},}^{n_1}\ \overbrace{0^{n_1},}^{n_1}\ \overbrace{\eta_1, \ldots, \eta_{n_1},}^{n_1}\ \overbrace{0}^{1})_{\mathbb{B}^{*(1)}},$$
$$\boldsymbol{k}_2 = (\overbrace{s_2(1, I),}^{2}\ \overbrace{0^2,}^{2}\ \overbrace{\rho_1, \rho_2,}^{2}\ \overbrace{0}^{1})_{\mathbb{B}^{*(2)}}.$$

Output the updated state S and the secret key $\boldsymbol{k}^*_{\overrightarrow{x},I} = (I, \boldsymbol{k}_0, \boldsymbol{k}_1, \boldsymbol{k}_2)$.

$\mathsf{Encrypt}(PK, L, \overrightarrow{y} = (y_1, \ldots, y_{n_1}) \in \mathbb{F}_q^{n_1} \setminus \{\overrightarrow{0}\}, M \in \mathbb{G}_T)$: If L is empty, set $L = \{N+1\}$, where $N+1$ is a dummy index. Choose $\delta, \zeta, \varphi, \varphi' \xleftarrow{\mathsf{U}} \mathbb{F}_q$, also choose $\varphi_r, \delta_r \xleftarrow{\mathsf{U}} \mathbb{F}_q$ for all $r \in L$ such that $\delta = \sum_{r\in L} \delta_r$, and compute:

$$\boldsymbol{c}_0 = (\delta, 0, \zeta, 0, \varphi)_{\mathbb{B}^{(0)}},$$

$$\boldsymbol{c}_1 = (\overbrace{\delta\overrightarrow{y},}^{n_1} \quad \overbrace{0^{n_1},}^{n_1} \quad \overbrace{0^{n_1},}^{n_1} \quad \overbrace{\varphi'}^{1} \;)_{\mathbb{B}^{(1)}},$$

$$\forall r \in L: \quad \boldsymbol{c}_r = (\overbrace{\delta_r(-r,1),}^{2} \quad \overbrace{0^2,}^{2} \quad \overbrace{0^2,}^{2} \quad \overbrace{\varphi_r}^{1})_{\mathbb{B}^{(2)}},$$

$$\boldsymbol{c}_M = g_T^{\zeta} M.$$

Output the ciphertext $C = (L, \boldsymbol{c}_0, \boldsymbol{c}_1, \{\boldsymbol{c}_r\}_{r\in L}, \boldsymbol{c}_M)$.

$\mathsf{Decrypt}(C, \boldsymbol{k}^*_{\overrightarrow{x},I})$: Given a ciphertext $C = (L, \boldsymbol{c}_0, \boldsymbol{c}_1, \{\boldsymbol{c}_r\}_{r\in L}, \boldsymbol{c}_M)$ and a secret key $\boldsymbol{k}^*_{\overrightarrow{x},I} = (I, \boldsymbol{k}_0, \boldsymbol{k}_1, \boldsymbol{k}_2)$, if $I \in L$, output $\perp$; otherwise compute and output message M:

$$M = \frac{\boldsymbol{c}_M}{e(\boldsymbol{c}_0, \boldsymbol{k}_0)e(\boldsymbol{c}_1, \boldsymbol{k}_1)\prod_{r\in L} e(\boldsymbol{c}_r, \boldsymbol{k}_2)^{\frac{1}{I-r}}}.$$

The correctness of our scheme holds due to the following observation. Let C and $\boldsymbol{k}^*_{\overrightarrow{x},I}$ be as above. If $\overrightarrow{x} \cdot \overrightarrow{y} = 0$ and $I \notin L$ then M can be recovered by $\boldsymbol{c}_M / (e(\boldsymbol{c}_0, \boldsymbol{k}_0)e(\boldsymbol{c}_1, \boldsymbol{k}_1)\prod_{r\in L} e(\boldsymbol{c}_r, \boldsymbol{k}_2)^{\frac{1}{I-r}})$, since

$$e(\boldsymbol{c}_0, \boldsymbol{k}_0)e(\boldsymbol{c}_1, \boldsymbol{k}_1)\prod_{r\in L} e(\boldsymbol{c}_r, \boldsymbol{k}_2)^{\frac{1}{I-r}} = g_T^{-s\delta+\zeta} g_T^{s_1\delta+\beta\delta\overrightarrow{x}\cdot\overrightarrow{y}} g_T^{s_2\sum_{r\in L}\delta_r} = g_T^{-s\delta+\zeta} g_T^{s\delta}.$$

Remark 3. In the $\mathsf{Encrypt}$ algorithm, if the revocation list L is empty, i.e., no key is revoked, a dummy index $N+1$ is placed into the revocation list. Since $N+1$ is not in the index space Γ, the ciphertext computed from $L = \{N+1\}$ and an attribute $\overrightarrow{y}$ can be decrypted by any key $\boldsymbol{k}^*_{\overrightarrow{x},I}$ provided $\overrightarrow{x} \cdot \overrightarrow{y} = 0$.

Theorem 1. *Our AH-RPE is adaptively attribute hiding against chosen plaintext attacks under the DLIN assumption. For any adversary $\mathcal{A}$, there exists a probabilistic polynomial time machine $\mathcal{D}$ such that for any security parameter λ,*

$$\mathsf{Adv}^{\mathsf{AH}}_{\mathcal{A},\mathsf{AH\text{-}RPE}}(\lambda) \le (2\nu+1)\mathsf{Adv}^{\mathrm{DLIN}}_{\mathcal{D}}(\lambda) + \psi$$

where ν is the maximum number of $\mathcal{A}$'s key queries and $\psi = (2\nu|L| + 18\nu + 10)/q$ ($|L|$ denotes the number of revoked keys).

The proof of Theorem 1 is presented in the full version.

5 An RPE Scheme with Full Hiding (FH-RPE)

Our second RPE scheme is based on Okamoto and Takashima's FE [15] and the Subset-Cover Framework due to Naor *et al.* [13]. At a high level, in addition to attribute and predicate vectors we use index and revocation vectors. The scheme can be seen as a combination of two encryption steps, one using attribute and predicate vectors, and the other one using index and revocation vectors.

Our scheme takes advantage of the **complete-subtree** data structure from [13]. Informally, in a binary tree with N leaves, the index I of a key will be associated with a leaf node. Each node in the tree will be assigned a unique identity. To compute a key with an index, we compute on identities of all the nodes on the path from the leaf node associated with I to the root node. To encrypt, the sender first finds a minimal set of nodes which contains an ancestor (or, the node itself) of all the non-revoked indexes. It then computes ciphertext on the attribute and the identities of all the nodes in that set. To retain the full-hiding property we apply the binary structure in an anonymous setting. Decryption works if there exists one common node (identity) between the key and the ciphertext, which is given for unrevoked keys only.

We now give a detailed specification of our FH-RPE scheme:

$\mathsf{Setup}\big(1^\lambda, \Delta = (\overrightarrow{n} = (2; n_1, n_2 = 2), N)\big)$: Perform the following computations: $(\mathsf{param}_{\overrightarrow{n}}, \mathbb{B}^{(0)}, \mathbb{B}^{*(0)}, \mathbb{B}^{(1)}, \mathbb{B}^{*(1)}, \mathbb{B}^{(2)}, \mathbb{B}^{*(2)}) \overset{R}{\leftarrow} \mathcal{G}_{\mathsf{ob}}(1^\lambda, \overrightarrow{n})$,

$\widetilde{\mathbb{B}}^{(0)} = (\boldsymbol{b}_1^{(0)}, \boldsymbol{b}_3^{(0)}, \boldsymbol{b}_5^{(0)})$, $\widetilde{\mathbb{B}}^{(1)} = (\boldsymbol{b}_1^{(1)}, \ldots, \boldsymbol{b}_{n_1}^{(1)}, \boldsymbol{b}_{3n_1+1}^{(1)})$, $\widetilde{\mathbb{B}}^{(2)} = (\boldsymbol{b}_1^{(2)}, \boldsymbol{b}_2^{(2)}, \boldsymbol{b}_7^{(2)})$,

$\widetilde{\mathbb{B}}^{*(0)} = (\boldsymbol{b}_1^{*(0)}, \boldsymbol{b}_3^{*(0)}, \boldsymbol{b}_4^{*(0)})$, $\widetilde{\mathbb{B}}^{*(1)} = (\boldsymbol{b}_1^{*(1)}, \ldots, \boldsymbol{b}_{n_1}^{*(1)}, \boldsymbol{b}_{2n_1+1}^{*(1)}, \ldots, \boldsymbol{b}_{3n_1}^{*(1)})$,
$\widetilde{\mathbb{B}}^{*(2)} = (\boldsymbol{b}_1^{*(2)}, \boldsymbol{b}_2^{*(2)}, \boldsymbol{b}_5^{*(2)}, \boldsymbol{b}_6^{*(2)})$.

Let *Tree* be a complete binary tree structure with at least N leaf nodes, which corresponds to the number of keys in the system. Each node x in *Tree* has unique identity ID_x. Let state information S, which records the assigned indexes I so far, be an initially empty set.

The output of the algorithm is given by the public key $PK = \big(1^\lambda, \mathsf{param}_{\overrightarrow{n}}, \{\widetilde{\mathbb{B}}^{(k)}\}_{k=0,1,2}, \mathit{Tree}\big)$, the master secret key $MSK = \big(\{\widetilde{\mathbb{B}}^{*(k)}\}_{k=0,1,2}\big)$, and the state information S.

$\mathsf{GenKey}(MSK, S, \overrightarrow{x} = (x_1, \ldots, x_{n_1}) \in \mathbb{F}_q^{n_1} \setminus \{\overrightarrow{0}\})$: Pick α, η, $\eta_1^{(1)}$, …, $\eta_{n_1}^{(1)}$, and $\beta^{(1)}$ uniformly at random from $\mathbb{F}_q$, and $\alpha^{(1)}, \alpha^{(2)} \overset{\mathsf{U}}{\leftarrow} \mathbb{F}_q$ such that $\alpha = \alpha^{(1)} + \alpha^{(2)}$. Choose index $I \overset{\mathsf{U}}{\leftarrow} \Gamma$ such that $I \notin S$; set $S = S \cup \{I\}$ and compute:

$$\boldsymbol{k}_0 = (-\alpha, 0, 1, \eta, 0)_{\mathbb{B}^{*(0)}},$$

$$\boldsymbol{k}_1 = (\overbrace{\alpha^{(1)}\overrightarrow{e}_1^{(1)} + \beta^{(1)}\overrightarrow{x},}^{n_1} \overbrace{0^{n_1},}^{n_1} \overbrace{\eta_1^{(1)}, \ldots, \eta_{n_1}^{(1)},}^{n_1} \overbrace{0}^{1})_{\mathbb{B}^{*(1)}},$$

$$\forall x \in \mathbb{P}(I): \quad \boldsymbol{k}_x = (\overbrace{\alpha^{(2)} + \beta_x^{(2)} ID_x, \beta_x^{(2)},}^{2} \overbrace{0^2,}^{2} \overbrace{\eta_{1,x}^{(2)}, \eta_{2,x}^{(2)},}^{2} \overbrace{0}^{1})_{\mathbb{B}^{*(2)}},$$

$$\text{with } \beta_x^{(2)}, \eta_{1,x}^{(2)}, \eta_{2,x}^{(2)} \overset{\mathsf{U}}{\leftarrow} \mathbb{F}_q.$$

The output is given by S and the secret key $\boldsymbol{k}^*_{\overrightarrow{x},I} = (I, \boldsymbol{k}_0, \boldsymbol{k}_1, \{\boldsymbol{k}_x\}_{x \in \mathbb{P}(I)})$. (Note that I is associated with the I_{th} leaf node in the binary tree. $\mathbb{P}(I)$ denotes all the nodes on the path from the leaf node I up to the root node (leaf and root nodes inclusive). The secret key $\boldsymbol{k}^*_{\overrightarrow{x},I}$ thus contains secrets for all nodes ID_x on the mentioned path from I to the root.)

$\mathsf{Encrypt}(PK, L, \overrightarrow{y} = (y_1, \ldots, y_{n_1}) \in \mathbb{F}_q^{n_1} \backslash \{\overrightarrow{0}\}, M \in \mathbb{G}_T)$: Choose $\delta, \zeta, \varphi, \varphi^{(1)} \xleftarrow{\mathsf{U}} \mathbb{F}_q$ and compute:

$$\boldsymbol{c}_0 = (\delta, 0, \zeta, 0, \varphi)_{\mathbb{B}^{(0)}},$$

$$\boldsymbol{c}_1 = (\overbrace{\delta\overrightarrow{y}}^{n_1}, \overbrace{0^{n_1}}^{n_1}, \overbrace{0^{n_1}}^{n_1}, \overbrace{\varphi^{(1)}}^{1})_{\mathbb{B}^{(1)}},$$

$$\forall x \in \mathsf{RevokeNodes}(\mathit{Tree}, L): \quad \boldsymbol{c}_x = (\overbrace{\delta, \delta(-ID_x)}^{2}, \overbrace{0^2}^{2}, \overbrace{0^2}^{2}, \overbrace{\varphi_x^{(2)}}^{1})_{\mathbb{B}^{(2)}}, \quad \text{where } \varphi_x^{(2)} \xleftarrow{\mathsf{U}} \mathbb{F}_q,$$

$$\boldsymbol{c}_M = g_T^{\zeta} M.$$

Output the ciphertext $C = (\boldsymbol{c}_0, \boldsymbol{c}_1, \{\boldsymbol{c}_x\}_{x \in \mathsf{RevokeNodes}(\mathit{Tree},L)}, \boldsymbol{c}_M)$.

Remark 4. Note that $\mathsf{RevokeNodes}(\mathit{Tree}, L)$ outputs a minimal set of nodes which contains an ancestor (or, the node itself) of all the non-revoked indexes. $\boldsymbol{c}_x$ is then computed on all the identities of the nodes in the set.

$\mathsf{Decrypt}(C, \boldsymbol{k}^*_{\overrightarrow{x},I})$: Given a ciphertext $C = (\boldsymbol{c}_0, \boldsymbol{c}_1, \{\boldsymbol{c}_x\}_{x \in \mathsf{RevokeNodes}(\mathit{Tree},L)}, \boldsymbol{c}_M)$ and a secret key $\boldsymbol{k}^*_{\overrightarrow{x},I} = (I, \boldsymbol{k}_0, \boldsymbol{k}_1, \{\boldsymbol{k}_{x'}\}_{x' \in \mathbb{P}(I)})$ compute

$$\forall x, x': \quad M_{x,x'} = \frac{\boldsymbol{c}_M}{e(\boldsymbol{c}_0, \boldsymbol{k}_0)e(\boldsymbol{c}_1, \boldsymbol{k}_1)e(\boldsymbol{c}_x, \boldsymbol{k}_{x'})}.$$

If there exists a pair (x, x') corresponding to the same node in *Tree* and $\overrightarrow{x} \cdot \overrightarrow{y} = 0$, the decrypted message is $M = M_{x,x'}$. Otherwise, obtained messages are random with all but negligible probability.

Correctness. Let C and $\boldsymbol{k}^*_{\overrightarrow{x},I}$ be as above. If $\overrightarrow{x} \cdot \overrightarrow{y} = 0$ and $I \notin L$ then M can be recovered by computing $\boldsymbol{c}_M / (e(\boldsymbol{c}_0, \boldsymbol{k}_0)e(\boldsymbol{c}_1, \boldsymbol{k}_1)e(\boldsymbol{c}_x, \boldsymbol{k}_{x'}))$, since

$$e(\boldsymbol{c}_0, \boldsymbol{k}_0)e(\boldsymbol{c}_1, \boldsymbol{k}_1)e(\boldsymbol{c}_x, \boldsymbol{k}_{x'}) = g_T^{-\alpha\delta+\zeta} g_T^{\alpha^{(1)}\delta+\beta^{(1)}\delta\overrightarrow{x}\cdot\overrightarrow{y}} g_T^{\alpha^{(2)}\delta+\beta_{x'}^{(2)}\delta(ID_{x'}-ID_x)} = g_T^{\zeta}.$$

Theorem 2. *FH-RPE is adaptively full hiding against chosen plaintext attacks under the DLIN assumption (provided the restriction in Remark 5 holds). For any adversary $\mathcal{A}$, there exists a probabilistic polynomial time machine $\mathcal{D}$ such that for any security parameter λ,*

$$\mathsf{Adv}^{\mathsf{FH}}_{\mathcal{A},\mathsf{FH\text{-}RPE}}(\lambda) \leq (2\nu + 1)\mathsf{Adv}^{\mathrm{DLIN}}_{\mathcal{D}}(\lambda) + \psi$$

where ν is the maximum number of $\mathcal{A}$'s key queries and $\psi = (2\log N\nu + 18\nu + \log N + 10)/q$.

The proof of Theorem 2 is presented in the full version.

Remark 5. In the proof of Theorem 2 we assume that $|X^{(0)}| = |X^{(1)}|$, where $X^{(0)} = \{x | x \in \mathsf{RevokeNodes}(\mathit{Tree}, L^{(0)})\}$ and $X^{(1)} = \{x' | x' \in \mathsf{RevokeNodes}(\mathit{Tree}, L^{(1)})\}$. The revocation lists $L^{(0)}$ and $L^{(1)}$ are defined in the challenge phase of Definition 5. This restriction is necessary to prevent the adversary from trivially distinguishing based on the length of the challenge ciphertext.

6 Conclusion

We formalized Revocable Predicate Encryption (RPE) and proposed two RPE schemes. Our AH-RPE scheme is attribute hiding whereas our FH-RPE scheme offers stronger full hiding. Both schemes are proven secure in the standard model under the DLIN assumption. Recently, Okamoto and Takashima [16] proposed a PE scheme with short private keys. We observe that private keys in our RPE constructions can be further reduced in size by adopting their techniques.

Acknowledgements. This work is part of the project between Germany and Australia, funded jointly by the German Academic Exchange Service (DAAD, grant 53361649) and by Australia's Department of Innovation, Industry, Science and Research (DIISR).

References

1. Aiello, W., Lodha, S., Ostrovsky, R.: Fast Digital Identity Revocation. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 137–152. Springer, Heidelberg (1998)
2. Attrapadung, N., Imai, H.: Attribute-Based Encryption Supporting Direct/Indirect Revocation Modes. In: Parker, M.G. (ed.) Cryptography and Coding 2009. LNCS, vol. 5921, pp. 278–300. Springer, Heidelberg (2009)
3. Attrapadung, N., Imai, H.: Conjunctive Broadcast and Attribute-Based Encryption. In: Shacham, H., Waters, B. (eds.) Pairing 2009. LNCS, vol. 5671, pp. 248–265. Springer, Heidelberg (2009)
4. Boldyreva, A., Goyal, V., Kumar, V.: Identity-based encryption with efficient revocation. In: ACM CCS 2008, pp. 417–426. ACM (2008)
5. Boneh, D., Boyen, X., Shacham, H.: Short Group Signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
6. Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
7. Boneh, D., Waters, B.: Conjunctive, Subset, and Range Queries on Encrypted Data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 535–554. Springer, Heidelberg (2007)
8. Gentry, C.: Certificate-Based Encryption and the Certificate Revocation Problem. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 272–293. Springer, Heidelberg (2003)
9. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM CCS 2006, pp. 89–98. ACM (2006)

10. Katz, J., Sahai, A., Waters, B.: Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)
11. Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010)
12. Lewko, A.B., Sahai, A., Waters, B.: Revocation systems with very small private keys. In: IEEE S&P, pp. 273–285. IEEE (2010)
13. Naor, D., Naor, M., Lotspiech, J.: Revocation and Tracing Schemes for Stateless Receivers. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 41–62. Springer, Heidelberg (2001)
14. Okamoto, T., Takashima, K.: Hierarchical Predicate Encryption for Inner-Products. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 214–231. Springer, Heidelberg (2009)
15. Okamoto, T., Takashima, K.: Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 191–208. Springer, Heidelberg (2010)
16. Okamoto, T., Takashima, K.: Achieving Short Ciphertexts or Short Secret-Keys for Adaptively Secure General Inner-Product Encryption. In: Lin, D., Tsudik, G., Wang, X. (eds.) CANS 2011. LNCS, vol. 7092, pp. 138–159. Springer, Heidelberg (2011)
17. Okamoto, T., Takashima, K.: Adaptively Attribute-Hiding (Hierarchical) Inner Product Encryption. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 591–608. Springer, Heidelberg (2012)
18. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
19. Shamir, A.: Identity-Based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
20. Shen, E., Shi, E., Waters, B.: Predicate Privacy in Encryption Systems. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 457–473. Springer, Heidelberg (2009)
21. Shi, E., Bethencourt, J., Chan, H.T.-H., Song, D.X., Perrig, A.: Multi-Dimensional Range Query over Encrypted Data. In: IEEE S&P, pp. 350–364. IEEE (2007)
22. Shi, E., Waters, B.: Delegating Capabilities in Predicate Encryption Systems. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part II. LNCS, vol. 5126, pp. 560–578. Springer, Heidelberg (2008)
23. Waters, B.: Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009)

Anonymous ID-Based Proxy Re-Encryption*

Jun Shao

School of Computer and Information Engineering
Zhejiang Gongshang University
310018, Hangzhou, Zhejiang, P.R. China
chn.junshao@gmail.com

Abstract. ID-based proxy re-encryption (IBPRE) allows a proxy with some information (a.k.a. re-encryption key) to transform the ciphertext under one identity to another ciphertext under another identity. These two ciphertexts can yield the same plaintext, while the proxy cannot get any information of the plaintext. Due to its transformable functionality, IBPRE can be used in many applications. Some of these applications require that the underlying IBPRE scheme is CCA-secure and anonymous. However, to the best of our knowledge, none of the existing schemes satisfy the security requirement. In this paper, we first extend the concept of IBPRE to that of anonymous IBPRE (AIBPRE), including the definition and security model. After that, we propose the first AIBPRE scheme, which can be proven-secure in the random oracle model based on the decisional bilinear Diffie-Hellman assumption and modified decisional bilinear Diffie-Hellman assumption.

Keywords: identity-based, proxy re-encryption, CCA security, anonymity.

1 Introduction

ID-based proxy re-encryption (IBPRE) proposed by Green and Ateniese[11] aims to give an efficient solution to the following problem: how to distribute data protected under one identity $\mathtt{id}_1$ to a user with a different identity $\mathtt{id}_2$ without revealing the private key corresponding to $\mathtt{id}_1$? In particular, IBPRE allows a proxy given special information (a.k.a. re-encryption key) to efficiently transform a ciphertext for the delegator ($\mathtt{id}_1$) to a ciphertext for the delegatee ($\mathtt{id}_2$) of the same message. In Green-Ateniese definition, there is only a fully trusted party named private key generator (PKG) whose responsibility is to generate the user's private key, while a little trust is placed in the proxy. That is, an adversary cannot learn the contents of messages encrypted under either $\mathtt{id}_1$ or $\mathtt{id}_2$, even if the adversary corrupts the proxy.

However, some applications of IBPRE require that the underlying IBPRE scheme can protect not only the confidentiality of the message, but also the

* The author thanks the anonymous reviewers for their insightful comments and helpful suggestions. This work was supported by NSFC No. 61003308, ZJGSUSF No. 1130XJ2010045, and ECZJF No. Y201017312.

privacy of intended recipients. For example, in the IBPRE-based distributed file systems, user $\mathtt{id}_1$, user $\mathtt{id}_2$ and the proxy act the file owner, file sharer(s) and file server, respectively. It is easy to see that a good file system should guarantee that only the intended recipients can obtain the file content even if the file server cannot do. The anonymity requirement may be from the file owner or the file sharer, who asks for the unlinkability of their activities on the file server. The unlinkability can be obtained by using multiple identities, which however leads to the extension of secret storage. A simple solution of anonymity is highly desirable for many encrypted communication scenarios [1].

Nevertheless, to the best of our knowledge, there is no IBPRE scheme hiding the contents of messages and the identities of the intended recipients at the same time. In this paper, we will propose the first such IBPRE scheme.

1.1 Our Contributions

The contributions in this paper can be summarized as follows.

– We propose the concept of IBPRE hiding the contents of messages and the identities of the recipients simultaneously, called anonymous IBPRE (AIBPRE).
– We propose the security models of AIBPRE, including the CCA (chosen ciphertext attack) security model corresponding to the confidentiality of messages, and the anonymity security model corresponding to the anonymity of the intended recipients. It was not easy to extend the security models of IBE to the ones of AIBPRE, especially with the anonymity security model. It is mainly because that compared to IBE, there are two kinds of ciphertexts (original ciphertexts and re-encrypted ciphertexts) and the re-encryption keys in AIBPRE. The situation in AIBPRE is more complex than that in IBE.
– We propose the first AIBPRE scheme that can be proven secure in the random oracle model. Designing such a scheme was a surprisingly difficult task. We cannot obtain an AIBPRE scheme directly from an anonymous IBE, such as BF-IBE scheme [4]. See the details in Section 4.

In this paper, we only consider single-use unidirectional AIBPRE scheme, i.e. the re-encrypted ciphertext cannot be further re-encrypted, and the proxy can only do the transformation from $\mathtt{id}_1$ to $\mathtt{id}_2$ but not from $\mathtt{id}_2$ to $\mathtt{id}_1$.

2 Related Work

The concept of proxy re-encryption (PRE), proposed by Blaze et al. [3] at Eurocrypt 1998, was formalized by Ateniese et al. [2]. Ivan and Dodis [13] proposed a generic construction for PRE from public key encryption. However, the schemes proposed by them are only CPA (chosen plaintext attack)-secure. Canetti and Hohenberger [6] proposed the first CCA-secure bidirectional PRE scheme where the proxy can do bidirectional transformation. Libert and Vergnaud [16] proposed the first replayable CCA-secure unidirectional PRE scheme in the standard model. Shao and Cao [18] proposed the first CCA-secure unidirectional PRE scheme in the random oracle model. Later, Chow et. al. [7] proposed a more efficient CCA-secure unidirectional PRE scheme in the random oracle model. Shao et al. [19] proposed a generic construction for CCA-secure unidirectional PRE by using CCA-secure $(2, 2)$ threshold encryption. Recently, Hanaoka [12] obtained a similar result by using similar techniques as that in [19]. Till now, many CCA-secure PRE schemes with some special properties were proposed, such as type-based PRE [24,8,10,26,27], PRE with invisible proxy [14], attribute-based PRE [15], PRE with key privacy (anonymous PRE) [1,20,21].

The first ID-based PRE (IBPRE) scheme was proposed by Green and Ateniese [11]. Chu and Tzeng [9] proposed the first replayable CCA-secure IBPRE scheme in the standard model. Tang et. al. [23] proposed an inter-domain IBPRE scheme. Wang et. al.[25] proposed the first CCA-secure IBPRE scheme in the standard model. Shao et. al. [22] proposed an identity-based conditional proxy re-encryption. Recently, Shao and Cao [17] proposed a generic construction for IBPRE from non-anonymous hierarchical identity-based encryption. However, none of the above IBPRE schemes are anonymous. In particular, the identities of the recipients in some of these schemes [9,22] can be easily revealed by using pairings, or the simulator of the security proof in some of these schemes [11,23,25] cannot generate all valid re-encryption keys for the delegation between any two identities. See the details of analysis on the schemes in [11,23,25] in Section 4.

In this paper, we will propose the first anonymous IBPRE scheme.

3 Preliminaries

3.1 Definitions for Single-Use Unidirectional AIBPRE

Definition 1 (Single-Use Unidirectional AIBPRE). *A single-use unidirectional AIBPRE scheme* `AIBPRE` *is a tuple of probabilistic polynomial time (PPT) algorithms (*`KeyGen`, `Ext`, `ReKeyGen`, `Enc`, `ReEnc`, `Dec`*):*

- $\texttt{KeyGen}(1^\lambda) \to (pk, sk)$. *On input a security parameter λ, the key generation algorithm* `KeyGen` *outputs the system parameter para, and the public/private key pair (pk, sk) of PKG. In the following algorithms, the system parameter para and the PKG's public key pk are included implicitly. This algorithm is performed by the PKG.*
- $\texttt{Ext}(\texttt{id}, sk) \to d_{\texttt{id}}$. *On input a user's identity* `id`*, and the PKG's private key sk, the extract algorithm* `Ext` *outputs the private key $d_{\texttt{id}}$ corresponding to the identity* `id`*. This algorithm is performed by the PKG.*
- $\texttt{ReKeyGen}(d_{\texttt{id}_1}, \texttt{id}_2) \to rk_{\texttt{id}_1,\texttt{id}_2}$. *On input a private key $d_{\texttt{id}_1}$ and a user's identity $\texttt{id}_2$, the re-encryption key generation algorithm* `ReKeyGen` *outputs a re-encryption key $rk_{\texttt{id}_1,\texttt{id}_2}$. This algorithm is performed by the delegator whose identity is $\texttt{id}_1$.*

- $\texttt{Enc}(\texttt{id}, m) \rightarrow C$. *On input a user's identity* $\texttt{id}$*, and a message* m *from the message space* $\mathcal{M}$*, the encryption algorithm* $\texttt{Enc}$ *outputs a ciphertext* C *under the identity* $\texttt{id}$*. This algorithm is performed by the encryptor.*
- $\texttt{ReEnc}(rk_{\texttt{id}_1,\texttt{id}_2}, C_1) \rightarrow C_2$. *On input a re-encryption key* $rk_{\texttt{id}_1,\texttt{id}_2}$ *and a ciphertext* C_1*, the re-encryption algorithm* $\texttt{ReEnc}$ *outputs a re-encrypted ciphertext* C_2 *under the identity* $\texttt{id}_2$ *or a special symbol* $\texttt{reject}$*. This algorithm is performed by the proxy holding the re-encryption key* $rk_{\texttt{id}_1,\texttt{id}_2}$*.*
- $\texttt{Dec}(d_{\texttt{id}}, C) \rightarrow m$. *On input a private key* $d_{\texttt{id}}$ *and a ciphertext* C*, the decryption algorithm* $\texttt{Dec}$ *outputs* m *in the message space or a special symbol* $\texttt{reject}$*. This algorithm is performed by the decryptor (the delegator or the delegatee).*

Correctness. The correctness property has two requirements. For any message m in the message space $\mathcal{M}$ and $(pk, sk) \leftarrow \texttt{KeyGen}(1^\lambda)$, the following two conditions must hold: $\texttt{Dec}(\texttt{Ext}(\texttt{id}, sk), \texttt{Enc}(\texttt{id}, m)) = m$ and $\texttt{Dec}(\texttt{Ext}(\texttt{id}_2, sk), \texttt{ReEnc}(\texttt{ReKeyGen}(\texttt{Ext}(\texttt{id}_1, sk), \texttt{id}_2), C)) = m$, where C is the ciphertext of the message m under $\texttt{id}_1$ from algorithm $\texttt{Enc}$.

3.2 Security Models for Single-Use Unidirectional AIBPRE

Indistinguishability of Encryptions under Chosen-Ciphertext Attack for Single-Use Unidirectional AIBPRE. This security model aims to guarantee the confidentiality of the message, i.e., the ciphertext can only be decrypted by the intended recipient(s).

Note that we have two types of ciphertexts (original ciphertexts from $\texttt{Enc}$ and re-encrypted ciphertexts from $\texttt{ReEnc}$) in AIBPRE; hence, there are two situations in this security model.

The challenge ciphertext is an original ciphertext.

***Setup*:** The Challenger $\mathcal{C}$ runs $\texttt{KeyGen}(1^\lambda)$ with the security parameter λ, and then sends the system parameter *para* and the PKG's public key *pk* to the adversary $\mathcal{A}$, but keeps the PKG's private key *sk* secret.

Phase 1: $\mathcal{A}$ issues queries $q_1, \cdots, q_{n_1}$ where query q_i is one of:

- *Extract oracle* $\mathcal{O}_{ext}$: On input $\texttt{id}$ by $\mathcal{A}$, $\mathcal{C}$ returns $d_{\texttt{id}}$ by running $\texttt{Ext}(\texttt{id}, sk)$. If an identity has been queried to $\mathcal{O}_{ext}$, it is considered as a corrupted one; otherwise, it is an uncorrupted one.
- *Re-encryption key generation oracle* $\mathcal{O}_{rk}$: On input $(\texttt{id}_1, \texttt{id}_2)$ by $\mathcal{A}$, $\mathcal{C}$ returns the re-encryption key $rk_{\texttt{id}_1,\texttt{id}_2} = \texttt{ReKeyGen}(\texttt{Ext}(\texttt{id}_1, sk), \texttt{id}_2)$.
- *Re-encryption oracle* $\mathcal{O}_{re}$: On input $(\texttt{id}_1, \texttt{id}_2, C_1)$ by $\mathcal{A}$, $\mathcal{C}$ returns the re-encrypted ciphertext $C_2 = \texttt{ReEnc}(\texttt{ReKeyGen}(\texttt{Ext}(\texttt{id}_1, sk), \texttt{id}_2), C)$.
- *Decryption oracle* $\mathcal{O}_{dec}$: On input $(\texttt{id}, C)$, $\mathcal{C}$ returns $\texttt{Dec}(\texttt{Ext}(\texttt{id}, sk), C)$.

These queries may be asked adaptively, that is, each query q_i may depend on the replies to $q_1, \cdots, q_{i-1}$.

Challenge: Once $\mathcal{A}$ decides that Phase 1 is over, it outputs two equal length plaintexts m_0, m_1 from the message space $\mathcal{M}$, and an identity $\texttt{id}^*$ on which it wishes to challenge. There are two restrictions on the identity $\texttt{id}^*$, (i) $\texttt{id}^*$ has not appeared in any query to $\mathcal{O}_{ext}$; (ii) if $(\texttt{id}^*, \bigstar)$ has appeared in any query to $\mathcal{O}_{rk}$, then $\bigstar$ should not appear in any query to $\mathcal{O}_{ext}$. $\mathcal{C}$ picks a random bit $\mathbf{b} \in \{0,1\}$ and sets $C^* = \texttt{Enc}(\texttt{id}^*, m_{\mathbf{b}})$. It sends C^* as the challenge to $\mathcal{A}$.

Phase 2: Almost the same as that in Phase 1, but with the following restrictions.

- $\mathcal{O}_{ext}$: On input $\texttt{id}$ by $\mathcal{A}$, if $\texttt{id}_1 = \texttt{id}^*$, $\mathcal{C}$ outputs $\texttt{reject}$.
- $\mathcal{O}_{rk}$: On input $(\texttt{id}_1, \texttt{id}_2)$ by $\mathcal{A}$, if $\texttt{id}_1 = \texttt{id}^*$, and $\texttt{id}_2$ has appeared in a query to $\mathcal{O}_{ext}$, $\mathcal{C}$ outputs $\texttt{reject}$.
- $\mathcal{O}_{re}$: On input $(\texttt{id}_1, \texttt{id}_2, C_1)$ by $\mathcal{A}$, if $(\texttt{id}_1, C_1) = (\texttt{id}^*, C^*)$, and $\texttt{id}_2$ has appeared in a query to $\mathcal{O}_{ext}$, $\mathcal{C}$ outputs $\texttt{reject}$.
- $\mathcal{O}_{dec}$: On input $(\texttt{id}, C)$, if $(\texttt{id}, C)$ is a derivative[1] of $(\texttt{id}^*, C^*)$, $\mathcal{C}$ outputs $\texttt{reject}$.

Guess: Finally, the adversary $\mathcal{A}$ outputs a guess $\mathbf{b}' \in \{0,1\}$ and wins the game if $\mathbf{b} = \mathbf{b}'$.

The advantage $\mathbf{Adv}^{\texttt{ID-IE-CCA-O}}_{\texttt{AIBPRE}}(\lambda)$ is defined as $|\Pr[\mathbf{b} = \mathbf{b}'] - 1/2|$. The scheme $\texttt{AIBPRE}$ is said to be *ID-IE-CCA-O* secure if all efficient adversaries $\mathcal{A}$ specified as above, the advantage $\mathbf{Adv}^{\texttt{ID-IE-CCA-O}}_{\texttt{AIBPRE}}(\lambda)$ is negligible.

The challenge ciphertext is a re-encrypted ciphertext.

Phase 1: Identical to that in the challenge original ciphertext case.

Challenge: Once the adversary $\mathcal{A}$ decides that Phase 1 is over, it outputs two equal length plaintexts m_0, m_1 from the message space, two uncorrupted identities $\texttt{id}_1^*$ and $\texttt{id}_2^*$ on which it wishes to challenge. $C^* = \texttt{ReEnc}(rk^*, \texttt{Enc}(\texttt{id}_2^*, m_{\mathbf{b}}))$, where rk^* is a re-encryption key from $\texttt{id}_1^*$ to $\texttt{id}_2^*$. It sends C^* as the challenge to $\mathcal{A}$.

Phase 2: Almost the same as that in Phase 1, but with the following restrictions.

- $\mathcal{O}_{ext}$: On input $\texttt{id}$, if $\texttt{id} = \texttt{id}_i^*$ $(i \in \{1,2\})$, $\mathcal{C}$ outputs $\texttt{reject}$.
- $\mathcal{O}_{dec}$: On input $(\texttt{id}, C)$, if $(\texttt{id}, C) = (\texttt{id}^*, C^*)$, $\mathcal{C}$ outputs $\texttt{reject}$.

Guess: Identical to that in the challenge original ciphertext case.

The advantage $\mathbf{Adv}^{\texttt{ID-IE-CCA-R}}_{\texttt{AIBPRE}}(\lambda)$ is defined as $|\Pr[\mathbf{b} = \mathbf{b}'] - 1/2|$. The scheme $\texttt{AIBPRE}$ is said to be *ID-IE-CCA-R* secure if all efficient adversaries $\mathcal{A}$ specified as above, the advantage $\mathbf{Adv}^{\texttt{ID-IE-CCA-R}}_{\texttt{AIBPRE}}(\lambda)$ is negligible.

---

[1] Derivatives of $(\texttt{id}^*, C^*)$ is adapted from [6]:
1. $(\texttt{id}^*, C^*)$ is a derivative of itself.
2. If $\mathcal{A}$ has queried $\mathcal{O}_{re}$ on input $(\texttt{id}^*, \texttt{id}, C^*)$ and obtained $(\texttt{id}, C)$, then $(\texttt{id}, C)$ is a derivative of $(\texttt{id}^*, C^*)$.
3. If $\mathcal{A}$ has queried $\mathcal{O}_{rk}$ on input $(\texttt{id}^*, \texttt{id})$, and $C = \texttt{ReEnc}(\mathcal{O}_{rk}(\texttt{id}^*, \texttt{id}), C^*)$, then $(\texttt{id}, C)$ is a derivative of $(\texttt{id}^*, C^*)$.

Remark 1. As mentioned in [2], collusion resistance is necessary for proxy re-encryption. This security guarantees that the delegatee colluding with the proxy cannot obtain the private key of the delegator. It is easy to show that the ID-IE-CCA-R security implies collusion resistance [16]. In particular, the adversary corrupting the delegatee and the proxy can easily decrypt the re-encrypted ciphertexts of the delegator, if collusion resistance is not held.

Anonimity under Chosen-Ciphertext Attack for Single-Use Unidirectional AIBPRE. This security model aims to guarantee the anonymity of the recipient(s). Informally speaking, ID-Ano-CCA security has three-fold. 1). the adversary cannot reveal the identity of the original ciphertext's recipient (anonymity of the original ciphertext), 2). the adversary cannot reveal the identity of the re-encrypted ciphertext's recipient (anonymity of the re-encrypted ciphertext), and 3). the adversary holding a re-encryption key cannot reveal the identities corresponding to the re-encryption key (anonymity of the re-encryption key). However, in this security model, only the first one and third one are dealt with. It is because that Shao et al. [20] has shown that the anonymity of the re-encrypted ciphertext is implied by the anonymity of the original ciphertext or the re-encryption key.

The challenge is an original ciphertext.

Phase 1: Identical to that in the ID-IE-CCA-O game for single-use unidirectional AIBPRE.

Challenge: Once $\mathcal{A}$ decides that Phase 1 is over, it outputs two identities $\mathtt{id}_0^*$ and $\mathtt{id}_1^*$, and a message m^*, on which it wishes to challenge. There are two restrictions on the identities $\mathtt{id}_0^*$ and $\mathtt{id}_1^*$, (i) $\mathtt{id}_i^*$ ($i \in \{0,1\}$) has not appeared in any query to $\mathcal{O}_{ext}$; (ii) if $(\mathtt{id}_i^*, \bigstar)$ ($i \in \{0,1\}$) has appeared in any query to $\mathcal{O}_{rk}$, then $\bigstar$ should not appear in any query to $\mathcal{O}_{ext}$. $\mathcal{C}$ picks a random bit $\mathbf{b} \in \{0,1\}$ and computes $C^* = \mathtt{Enc}(\mathtt{id}_{\mathbf{b}}^*, m^*)$. At last, $\mathcal{C}$ sends C^* as the challenge to $\mathcal{A}$.

Phase 2: Almost the same as that in Phase 1, but with the following restrictions.

- $\mathcal{O}_{ext}$: On input $\mathtt{id}$ by $\mathcal{A}$, if $\mathtt{id} = \mathtt{id}_i^*$ ($i \in \{0,1\}$), $\mathcal{C}$ outputs `reject`.
- $\mathcal{O}_{rk}$: On input $(\mathtt{id}_1, \mathtt{id}_2)$ by $\mathcal{A}$, if $\mathtt{id}_1 = \mathtt{id}_i^*$ ($i \in \{0,1\}$), and $\mathtt{id}_2$ has appeared in a query to $\mathcal{O}_{ext}$, $\mathcal{C}$ outputs `reject`.
- $\mathcal{O}_{re}$: On input $(\mathtt{id}_1, \mathtt{id}_2, C_1)$ by $\mathcal{A}$, if $(\mathtt{id}_1, C_1) = (\mathtt{id}_i^*, C^*)$ ($i \in \{0,1\}$), and $\mathtt{id}_2$ has appeared in a query to $\mathcal{O}_{ext}$, $\mathcal{C}$ outputs `reject`.
- $\mathcal{O}_{dec}$: On input $(\mathtt{id}, C)$, if $(\mathtt{id}, C)$ is not a derivative of $(\mathtt{id}_i^*, C^*)$ ($i \in \{0,1\}$), $\mathcal{C}$ outputs `reject`.

Guess: Finally, the adversary $\mathcal{A}$ outputs a guess $\mathbf{b}' \in \{0,1\}$ and wins the game if $\mathbf{b} = \mathbf{b}'$.

The advantage $\mathbf{Adv}_{\mathtt{AIBPRE}}^{\mathtt{ID\text{-}Ano\text{-}CCA\text{-}O}}(\lambda)$ is defined as $|\Pr[\mathbf{b} = \mathbf{b}'] - 1/2|$. The scheme `PRE` is said to be *ID-Ano-CCA-O* secure if all efficient adversaries $\mathcal{A}$ specified as above, the advantage $\mathbf{Adv}_{\mathtt{AIBPRE}}^{\mathtt{ID\text{-}Ano\text{-}CCA\text{-}O}}(\lambda)$ is negligible.

The challenge is a re-encryption key.

Phase 1: Identical to that in the ID-Ano-CCA-O game for single-use unidirectional AIBPRE.

Challenge: Once the adversary $\mathcal{A}$ decides that Phase 1 is over, it outputs two identities $\mathtt{id}_I^*$ and $\mathtt{id}_J^*$, on which it wishes to challenge, where $\mathtt{id}_I^*$ and $\mathtt{id}_J^*$ are two uncorrupted identities. $\mathcal{C}$ picks a random bit $\mathbf{b} \in \{0,1\}$. If $\mathbf{b} = 0$, then it sets $rk_{\mathtt{id}_I^*,\mathtt{id}_J^*}$ as a random key from the re-encryption key space; otherwise, it sets $rk_{\mathtt{id}_I^*,\mathtt{id}_J^*} = \mathtt{ReKeyGen}(d_{\mathtt{id}_I^*}, \mathtt{id}_J^*)$, where $d_{\mathtt{id}_I^*}$ is the corresponding private key of $\mathtt{id}_I^*$. At last, $\mathcal{C}$ sends $rk_{\mathtt{id}_I^*,\mathtt{id}_J^*}$ as the challenge to $\mathcal{A}$.

Phase 2: It runs almost the same as that in Phase 1, but with the following restrictions.
- $\mathcal{O}_{ext}$: On input $\mathtt{id}$ by $\mathcal{A}$, if $\mathtt{id} = \mathtt{id}_I^*$ or $\mathtt{id} = \mathtt{id}_J^*$, $\mathcal{C}$ outputs $\mathtt{reject}$; otherwise, $\mathcal{C}$ responds as in Phase 1.
- $\mathcal{O}_{dec}$: The input $(\mathtt{id}, C)$ cannot satisfy the following situations simultaneously:
  - $\mathtt{id} = \mathtt{id}_J$;
  - C is a re-encrypted ciphertext computed by the challenge re-encryption key.

Guess: Finally, the adversary $\mathcal{A}$ outputs a guess $\mathbf{b}' \in \{0,1\}$ and wins the game if $\mathbf{b} = \mathbf{b}'$.

The advantage $\mathbf{Adv}_{\mathtt{AIBPRE}}^{\mathtt{ID\text{-}Ano\text{-}CCA\text{-}R}}(\lambda)$ is defined as $|\Pr[\mathbf{b} = \mathbf{b}'] - 1/2|$. The scheme $\mathtt{PRE}$ is said to be *ID-Ano-CCA-R* secure if all efficient adversaries $\mathcal{A}$ specified as above, the advantage $\mathbf{Adv}_{\mathtt{AIBPRE}}^{\mathtt{ID\text{-}Ano\text{-}CCA\text{-}R}}(\lambda)$ is negligible.

*Remark 2 (**Restrictions**).* Like that in [20,21], the restrictions in $\mathcal{O}_{dec}$ in Phase 2 of the ID-Ano-CCA-R game for single-use unidirectional AIBPRE are reasonable, since if not, the adversary can trivially decide whether the challenge re-encryption key is a random value or a real re-encryption key as follows. The adversary first encrypts a message m with $\mathtt{id}_I^*$ to get a ciphertext C, and then it re-encrypts C with the challenge re-encryption key to get another ciphertext C'. At last, the adversary queries the decryption oracle with $(\mathtt{id}_J^*, C')$. If the resulting message equals to m, then the challenge re-encryption key is a real one; otherwise, it is a random value.

The other question on the restrictions is whether it is too weak, since it is not easy for the decryptor to decide whether a re-encrypted ciphertext is computed by a specific re-encryption key. However, it is possible to make the decryptor have the ability: If every re-encrypted ciphertext computed from the same re-encryption key contains the same value, which can be obtained by the decryptor but not the proxy. In our proposal, the re-encrypted ciphertexts re-encrypted by the same re-encryption key always contain the same c_5. See the details in Section 4.2.

3.3 Bilinear Groups

In this subsection, we briefly review the definitions about bilinear maps and bilinear map groups, which follow that in [4,5].

1. $\mathbb{G}$ and $\mathbb{G}_T$ are two (multiplicative) cyclic groups of prime order q;
2. g is a generator of $\mathbb{G}$;
3. e is a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$.

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two groups as above. An *admissible bilinear map* is a map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ with the following properties:

1. *Bilinearity*: For all $P, Q, R \in \mathbb{G}$, $e(P \cdot Q, R) = e(P, R) \cdot e(Q, R)$ and $e(P, Q \cdot R) = e(P, Q) \cdot e(P, R)$.
2. *Non-degeneracy*: If $e(P, Q) = 1$ for all $Q \in \mathbb{G}$, then $P = \mathcal{O}$, where $\mathcal{O}$ is a point at infinity.

We say that $\mathbb{G}$ is a bilinear group if the group action in $\mathbb{G}$ can be computed efficiently and there exists a group $\mathbb{G}_T$ and an efficiently computable bilinear map as above. We denote `BSetup` as an algorithm that, on input the security parameter λ, outputs the parameters for a bilinear map as $(q, g, \mathbb{G}, \mathbb{G}_T, e)$, where $q \in \Theta(2^\lambda)$.

3.4 Complexity Assumptions

Definition 2 (Decisional Bilinear Diffie-Hellman Assumption). *Let* $(q, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow$ ***BSetup***(1^k). *The decisional Bilinear Diffie-Hellman problem (DBDH) in* $(\mathbb{G}, \mathbb{G}_T)$ *is defined as follows: given 5-tuple* $(g, g^a, g^b, g^c, S) \in \mathbb{G}^4 \times \mathbb{G}_T$ *as input, decide whether* $S = e(g, g)^{abc}$. *An algorithm* $\mathcal{A}$ *has advantage* ε *in solving the DBDH problem in* $(\mathbb{G}, \mathbb{G}_T)$ *if*

$$|Pr[\mathcal{A}(g, g^a, g^b, g^c, e(g, g)^{abc}) = 0] - Pr[\mathcal{A}(g, g^a, g^b, g^c, S) = 0]| \geq \epsilon,$$

where the probability is taken over the random choices of $a, b, c \in \mathbb{Z}_q^*$, $S \in \mathbb{G}$ *and the random bits of* $\mathcal{A}$.

We say that the (t, ϵ)*-decisional Bilinear Diffie-Hellman (DBDH) assumption holds in* $(\mathbb{G}, \mathbb{G}_T)$ *if no t-time algorithm has advantage* ϵ *at least in solving the DBDH problem in* $(\mathbb{G}, \mathbb{G}_T)$.

Definition 3 (modified Decisional Bilinear Diffie-Hellman Assumption). *Let* $(q, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow$ ***BSetup***(1^k). *The modified decisional Bilinear Diffie-Hellman problem (mDBDH) in* $(\mathbb{G}, \mathbb{G}_T)$ *is defined as follows: given 4-tuple* $(g, g^a, g^b, S) \in \mathbb{G}^3 \times \mathbb{G}_T$ *as input, decide whether* $S = e(g, g)^{a^2 b}$. *An algorithm* $\mathcal{A}$ *has advantage* ε *in solving the mDBDH problem in* $(\mathbb{G}, \mathbb{G}_T)$ *if*

$$|Pr[\mathcal{A}(g, g^a, g^b, e(g, g)^{a^2 b}) = 0] - Pr[\mathcal{A}(g, g^a, g^b, S) = 0]| \geq \epsilon,$$

where the probability is taken over the random choices of $a, b \in \mathbb{Z}_q^*$, $S \in \mathbb{G}$ *and the random bits of* $\mathcal{A}$.

We say that the (t, ϵ)*-modified decisional Bilinear Diffie-Hellman (mDBDH) assumption holds in* $(\mathbb{G}, \mathbb{G}_T)$ *if no t-time algorithm has advantage* ϵ *at least in solving the mDBDH problem in* $(\mathbb{G}, \mathbb{G}_T)$.

In this paper, we drop the t and ε and refer to the DBDH (mDBDH) assumption rather than the (ε, t)-DBDH (mDBDH) assumption.

4 The Proposed Scheme

4.1 A Wrong Design

In this subsection, we propose an AIBPRE scheme based on the AIBE scheme—BF-IBE scheme [4]. For simplicity, we only show the CPA-secure version here. Actually the schemes in [11,23,25] follow the method used in the scheme.

`KeyGen`: On input the security parameter 1^λ, it outputs the system parameter $(\mathbb{G}, \mathbb{G}_T, q, g, H_i, i = 1, 2, 3, 4, 5)$, the key pair of the PKG $(pk, sk) = (y, x)$, where $(q, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \texttt{BSetup}(1^\lambda)$, $H_i, (i = 1, 2)$ are hash functions $H_i : \{0,1\}^* \to \mathbb{G}$ for $i = 1, 2$, and $y = g^x$. The system parameter is included in the following algorithms implicitly.

`Ext`: On input an identity `id` and the private key $sk = x$ of the PKG, it outputs the private key corresponding to `id`: $d_{\texttt{id}} = H_1(\texttt{id})^x$.

`ReKeyGen`: On input the delegator's private key $d_{\texttt{id}_1}$ and the delegatee's identity $\texttt{id}_2$, it outputs the re-encryption key from $\texttt{id}_1$ to $\texttt{id}_2$: $rk_{\texttt{id}_1,\texttt{id}_2} = (rk^{(1)}_{\texttt{id}_1,\texttt{id}_2}, rk^{(2)}_{\texttt{id}_1,\texttt{id}_2}) = (g^{\bar{x}}, d_{\texttt{id}_1} \cdot H_2(e(y, H_1(\texttt{id}_2))^{\bar{x}}))$, where $\bar{x}$ is a random number from $\mathbb{Z}_q^*$.

`Enc`: On input $m \in \mathbb{G}_T$, an identity `id`, it outputs the ciphertext $C = (c_1, c_2) = (g^r, m \cdot e(H_1(\texttt{id}), y)^r)$.

`ReEnc`: On input an original ciphertext $C = (c_1, c_2)$ under identity $\texttt{id}_1$, and a re-encryption key $rk_{\texttt{id}_1,\texttt{id}_2} = (rk^{(1)}_{\texttt{id}_1,\texttt{id}_2}, rk^{(2)}_{\texttt{id}_1,\texttt{id}_2})$ from $\texttt{id}_1$ to $\texttt{id}_2$, it outputs the re-encrypted ciphertext $C' = (c'_1, c'_2, c'_3, c'_4) = (c_1, c_2, rk^{(1)}_{\texttt{id}_1,\texttt{id}_2}, e(c_1, rk^{(2)}_{\texttt{id}_1,\texttt{id}_2}))$.

`Dec`: On input a ciphertext C under `id`, and a private key $d_{\texttt{id}}$, the algorithm is as follows.
- If C is an original ciphertext (c_1, c_2), compute $m = c_2 / e(d_{\texttt{id}}, c_1)$.
- If C is a re-encrypted ciphertext (c'_1, c'_2, c'_3, c'_4), compute $m = c'_2 \cdot e(c'_1, H_2(c'_3, d_{\texttt{id}}))/c'_4$.

At first glance, the above scheme is anonymous, since from original ciphertexts, re-encrypted ciphertexts, and re-encryption keys, the adversary cannot reveal the identities of the corresponding users. However, it is not true.

On the one hand, for the re-encryption key challenge case of the anonymity security model, the simulator should have the ability to generate *all* valid re-encryption keys for the delegation between any two identities. On the other hand, without knowing the private key $d_{\texttt{id}_I^*}$ (that is unknown to the simulator), the simulator cannot generate valid any re-encryption key for any delegation from $\texttt{id}_I^*$ to any corrupted identity. With this conflict, we deduce that the above scheme is not proven-anonymous at least.

4.2 Description of the Proposal

In this section, we give the first anonymous ID-based proxy re-encryption.

`KeyGen`: On input the security parameter λ, it outputs the system parameter $(\mathbb{G}, \mathbb{G}_T, q, g, h, H_i, i = 1, 2, 3, 4, 5)$, the key pair of the PKG $(pk, sk) = (y, x)$, where $(q, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \texttt{BSetup}(1^\lambda)$, h is a random number from $\mathbb{G}$, $H_i, (i = 1, 2, 3, 4, 5)$ are hash functions $H_i : \{0,1\}^* \rightarrow \mathbb{G}$ for $i = 1, 2, 5$, $H_3 : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$ and $H_4 : \{0,1\}^* \rightarrow \mathbb{G}^2$, and $y = g^x$. The system parameter is included in the following algorithms implicitly.

`Ext`: On input an identity `id` and the private key $sk = x$ of the PKG, it outputs the private key corresponding to `id`: $d_{\texttt{id}} = H_1(\texttt{id})^x$.

`ReKeyGen` : On input the delegator's private key $d_{\texttt{id}_1}$ and the delegatee's identity $\texttt{id}_2$, it outputs the re-encryption key from $\texttt{id}_1$ to $\texttt{id}_2$: $rk_{\texttt{id}_1,\texttt{id}_2} = (rk^{(1)}_{\texttt{id}_1,\texttt{id}_2}, rk^{(2)}_{\texttt{id}_1,\texttt{id}_2}) = (g^{\bar{x}}, d_{\texttt{id}_1} \cdot h^{\bar{x}} \cdot H_2(e(y, H_1(\texttt{id}_2))^{\bar{x}}))$, where $\bar{x}$ is a random number from $\mathbb{Z}_q^*$.

`Enc`: On input $m \in \mathbb{G}$, an identity `id`, it outputs the ciphertext $C = (c_1, c_2, c_3, c_4) = (h^r, g^r, (m||\sigma) \oplus H_4(e(H_1(\texttt{id}), y)^r), H_5(c_1||c_2||c_3)^r)$ where σ is a random number from $\mathbb{G}$, and $r = H_3(m||\sigma)$.

`ReEnc`: On input an original ciphertext $C = (c_1, c_2, c_3, c_4)$ under identity $\texttt{id}_1$, and a re-encryption key $rk_{\texttt{id}_1,\texttt{id}_2} = (rk^{(1)}_{\texttt{id}_1,\texttt{id}_2}, rk^{(2)}_{\texttt{id}_1,\texttt{id}_2})$ from $\texttt{id}_1$ to $\texttt{id}_2$, it outputs the re-encrypted ciphertext $C' = (c_2, c_3, c_5, c_6)$ if $e(c_1, g) \overset{?}{=} e(h, c_2)$ and $e(c_1, H_5(c_1||c_2||c_3)) \overset{?}{=} e(h, c_4)$ both hold; otherwise, it outputs `reject`.

$$\begin{aligned} c_5 &= rk^{(1)}_{\texttt{id}_1,\texttt{id}_2} = g^{\bar{x}} \\ c_6 &= \frac{e(rk^{(2)}_{\texttt{id}_1,\texttt{id}_2}, c_2)}{e(c_1, rk^{(1)}_{\texttt{id}_1,\texttt{id}_2})} = \frac{e(H_1(\texttt{id}_1)^x \cdot h^{\bar{x}} \cdot H_2(e(y, H_1(\texttt{id}_2)^{\bar{x}})), g^r)}{e(h^r, g^{\bar{x}})} \\ &= e(H_1(\texttt{id}_1)^x, g^r) \cdot e(H_2(e(y, H_1(\texttt{id}_2)^{\bar{x}})), g^r) \end{aligned}$$

`Dec`: Since there exist two types of ciphertext, we have two situations in this algorithm.

- If C is an original ciphertext (c_1, c_2, c_3, c_4), then the decryptor is $\texttt{id}_1$. The decryptor first checks $e(c_1, g) \overset{?}{=} e(h, c_2)$ and $e(c_1, H_5(c_1||c_2||c_3)) \overset{?}{=} e(h, c_4)$. If one of the equations does not hold, then output `reject` and abort; otherwise, do the following steps.
  - Compute $m||\sigma = c_3 \oplus H_4(e(d_{\texttt{id}_1}, c_2)) = (m||\sigma) \oplus H_4(e(H_1(\texttt{id}_1), y)^r) \oplus H_4(e(H_1(\texttt{id}_1)^x, g^r))$, and $r = H_2(m||\sigma)$.
  - Check $g^r \overset{?}{=} c_2$. If it holds, then output m; otherwise, output `reject`.
- If C is a re-encrypted ciphertext (c_2, c_3, c_5, c_6), then the decryptor is $\texttt{id}_2$. The decryptor does the following steps.
  - Compute

$$\begin{aligned} R &= \frac{c_6}{e(H_2(e(d_{\texttt{id}_2}, c_5)), c_2)} = \frac{e(H_1(\texttt{id}_1)^x, g^r) \cdot e(H_2(e(y, H_1(\texttt{id}_2)^{\bar{x}})), g^r)}{e(H_2(e(H_1(\texttt{id}_2)^x, g^{\bar{x}})), g^r)} \\ &= e(H_1(\texttt{id}_1)^x, g^r) \end{aligned}$$

  - Compute $m||\sigma = c_3 \oplus H_4(R) = (m||\sigma) \oplus H_4(e(H_1(\texttt{id}_1), y)^r) \oplus H_4(e(H_1(\texttt{id}_1)^x, g^r))$, and $r = H_3(m||\sigma)$.
  - Check $g^r \overset{?}{=} c_2$. If it holds, then output m; otherwise, output `reject`.

Correctness. It is easy to check the correctness of our proposal, we omit it here.

4.3 Security Analysis

Due to the limited space, we only present the theorems here.

Theorem 1 (ID-IE-CCA Security). *Our proposal is ID-IE-CCA secure based on the DBDH assumption in the random oracle model.*

Theorem 2 (ID-IK-CCA Security). *Our proposal is ID-IK-CCA secure based on the DBDH assumption and mDBDH assumption in the random oracle model.*

5 Conclusion

In this paper, we propose the concept of anonymous ID-based proxy re-encryption (AIBPRE), including the definition and the security models. Furthermore, the first concrete AIBPRE scheme is also proposed. Based on the DBDH assumption and the mDBDH assumption, we give the security proofs (including CCA security and anonymity) in the random oracle model. An interesting open problem is to find efficient AIBPRE proven-secure in the standard model.

References

1. Ateniese, G., Benson, K., Hohenberger, S.: Key-Private Proxy Re-encryption. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 279–294. Springer, Heidelberg (2009)
2. Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. In: NDSS. The Internet Society (2005)
3. Blaze, M., Bleumer, G., Strauss, M.: Divertible Protocols and Atomic Proxy Cryptography. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 127–144. Springer, Heidelberg (1998)
4. Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
5. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. SIAM Journal of Computing 32(3), 586–615 (2003)
6. Canetti, R., Hohenberger, S.: Chosen-ciphertext secure proxy re-encryption. In: ACM Conference on Computer and Communications Security, pp. 185–194. ACM (2007)
7. Chow, S.S.M., Weng, J., Yang, Y., Deng, R.H.: Efficient Unidirectional Proxy Re-Encryption. In: Bernstein, D.J., Lange, T. (eds.) AFRICACRYPT 2010. LNCS, vol. 6055, pp. 316–332. Springer, Heidelberg (2010)
8. Chu, C.-K., Weng, J., Chow, S.S.M., Zhou, J., Deng, R.H.: Conditional Proxy Broadcast Re-Encryption. In: Boyd, C., González Nieto, J. (eds.) ACISP 2009. LNCS, vol. 5594, pp. 327–342. Springer, Heidelberg (2009)
9. Chu, C.-K., Tzeng, W.-G.: Identity-Based Proxy Re-Encryption Without Random Oracles. In: Garay, J.A., Lenstra, A.K., Mambo, M., Peralta, R. (eds.) ISC 2007. LNCS, vol. 4779, pp. 189–202. Springer, Heidelberg (2007)

10. Fang, L., Susilo, W., Wang, J.: Anonymous Conditional Proxy Re-Encryption without Random Oracle. In: Pieprzyk, J., Zhang, F. (eds.) ProvSec 2009. LNCS, vol. 5848, pp. 47–60. Springer, Heidelberg (2009)
11. Green, M., Ateniese, G.: Identity-Based Proxy Re-Encryption. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 288–306. Springer, Heidelberg (2007)
12. Hanaoka, G., Kawai, Y., Kunihiro, N., Matsuda, T., Weng, J., Zhang, R., Zhao, Y.: Generic Construction of Chosen Ciphertext Secure Proxy Re-Encryption. In: Dunkelman, O. (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 349–364. Springer, Heidelberg (2012)
13. Ivan, A.-A., Dodis, Y.: Proxy cryptography revisited. In: NDSS. The Internet Society (2003)
14. Jia, X., Shao, J., Jing, J., Liu, P.: Cca-secure type-based proxy re-encryption with invisible proxy. In: CIT, pp. 1299–1305. IEEE Computer Society (2010)
15. Liang, X., Cao, Z., Lin, H., Shao, J.: Attribute Based Proxy Re-encryption with Delegating Capabilities. In: ACM ASIACCS 2009, pp. 276–286 (2009)
16. Libert, B., Vergnaud, D.: Unidirectional Chosen-Ciphertext Secure Proxy Re-encryption. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 360–379. Springer, Heidelberg (2008)
17. Shao, J., Cao, Z.: Multi-use unidirectional identity-based proxy re-encryption from hierarchical identity-based encryption. Information Sciences (to appear)
18. Shao, J., Cao, Z.: CCA-Secure Proxy Re-encryption without Pairings. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 357–376. Springer, Heidelberg (2009)
19. Shao, J., Cao, Z., Liu, P.: Sccr: a generic approach to simultaneously achieve cca security and collusion-resistance in proxy re-encryption. Security and Communication Networks 4(2), 122–135 (2011)
20. Shao, J., Liu, P., Wei, G., Ling, Y.: Anonymous proxy re-encryption. Security and Communication Networks 5(5), 439–449 (2012)
21. Shao, J., Liu, P., Zhou, Y.: Achieving key privacy without losing cca security in proxy re-encryption. Journal of Systems and Software 85(3), 655–665 (2012)
22. Shao, J., Wei, G., Ling, Y., Xie, M.: Identity-based conditional proxy re-encryption. In: IEEE ICC 2011 (2011)
23. Tang, Q., Hartel, P., Jonker, W.: Inter-domain Identity-Based Proxy Re-encryption. In: Yung, M., Liu, P., Lin, D. (eds.) Inscrypt 2008. LNCS, vol. 5487, pp. 332–347. Springer, Heidelberg (2009)
24. Tang, Q.: Type-Based Proxy Re-encryption and Its Construction. In: Chowdhury, D.R., Rijmen, V., Das, A. (eds.) INDOCRYPT 2008. LNCS, vol. 5365, pp. 130–144. Springer, Heidelberg (2008)
25. Wang, H., Cao, Z., Wang, L.: Multi-use and unidirectional identity-based proxy re-encryption schemes. Inf. Sci. 180(20), 4042–4059 (2010)
26. Weng, J., Deng, R.H., Chu, C., Ding, X., Lai, J.: Conditional Proxy Re-Encryption Secure against Chosen-Ciphertext Attack. In: ACM ASIACCS 2009, pp. 322–332 (2009)
27. Weng, J., Yang, Y., Tang, Q., Deng, R.H., Bao, F.: Efficient Conditional Proxy Re-encryption with Chosen-Ciphertext Security. In: Samarati, P., Yung, M., Martinelli, F., Ardagna, C.A. (eds.) ISC 2009. LNCS, vol. 5735, pp. 151–166. Springer, Heidelberg (2009)

On the Optimality of Lattices for the Coppersmith Technique

Yoshinori Aono[1], Manindra Agrawal[2], Takakazu Satoh[3], and Osamu Watanabe[4]

[1] National Institute of Information and Communications Technology, Tokyo, Japan
aono@nict.go.jp
[2] Department of Computer Science and Engineering, Indian Institute of Technology, Kanpur, India
manindra@iitk.ac.in
[3] Department of Mathematics, Tokyo Institute of Technology, Tokyo, Japan
[4] Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, Tokyo, Japan
watanabe@is.titech.ac.jp

Abstract. We investigate the Coppersmith technique [7] for finding solutions of a univariate modular equation within a range given by range parameter U. This paper provides a way to analyze a general type of limitation of the lattice construction. Our analysis bounds the possible range of U from above that is asymptotically equal to the bound given by the original result of Coppersmith. To show our result, we establish a framework for the technique by following the reformulation of Howgrave-Graham [14], and derive a condition for the technique to work. We then provide a way to analyze a bound of U for achieving the condition. Technically, we show that (i) the original result of Coppersmith achieves an optimal bound for U when constructing a lattice in a standard way. We then show evidence supporting that (ii) a non-standard lattice construction is generally difficult. We also report on computer experiments demonstrating the tightness of our analysis. Some of the detailed arguments are omitted due to the space limit; see the full-version [1].

Keywords: Lattice, Coppersmith technique, Univariate equation, Impossibility result, RSA.

1 Introduction

Coppersmith [7] introduced a polynomial-time algorithm, which we refer to as the *Coppersmith technique*, for finding solutions of a modular equation

$$F(x) = x^D + a_{D-1}x^{D-1} + \cdots + a_0 \equiv 0 \pmod{N} \tag{1}$$

within the range of $|x| < N^{1/D-\varepsilon}$, and showed that it can be used to design an attacking algorithm for RSA cryptography. (Here, for any A, $0 < A < N$, the notation $|x| < A$ under modulo N means that x is an integer satisfying $0 \le x < A$ or $N - A < x < N$.) Since his work and the reformulation by Howgrave-Graham

[14], it has gained attention in relation to attack several cryptographies; (e.g., [3,4]). The technique has also been generalized for the multivariate case.

The outline of the Coppersmith technique is (i) converting a given modular equation to a certain algebraic equation keeping the same small solutions by using a lattice reduction algorithm and (ii) solving the algebraic equation by a numerical method. One key point of this technique is constructing a good lattice for the lattice reduction algorithm. For instance, considering a bivariate modular equation for RSA cryptanalysis the original result of [4] has been improved essentially by defining better lattices [10,2]. There should clearly be some limit in such improvements. Here, we focus on the univariate case and investigate the optimality of the lattice construction for the Coppersmith technique. We demonstrate that for solving (1) the range of $|x| < N^{1/D-\varepsilon}$ Coppersmith achieved based on his lattice construction is "in some sense" optimal.

Note that some investigations have shown the limit of polynomial-time algorithms. Konyagin and Steger [17] gave an upper bound of the number of roots of (1) within the range of $|x| < U$, which becomes exponential in $\log N$ when $U = N^{1/D+\varepsilon}$ (The bound is attained by an equation of the form $x^r \equiv 0 \pmod{p^r}$ [21].) This somewhat extreme example provides that there are *some* equations such that no polynomial-time algorithm works to find *all* solutions within the range of $|x| < N^{1/D+\varepsilon}$; however it is insufficient for showing the hardness of solving particular equations. In fact, for most such equations, the number of solutions is easily shown to be quite small. Our objective is to provide a technique to analyze the limit of the Coppersmith technique that is applicable for equations for attacking cryptographies.

Our Results: In this paper, we show a general type limitation of lattices used in the Coppersmith technique for solving any univariate equation (1), by which we can show that the Coppersmith's bound $U = N^{1/D-\varepsilon}$ is best (except for the choice of ε). Our result consists of two technical results. We investigate a certain condition — the *lattice condition* — which is sufficient for the Coppersmith technique to work. This condition is essentially determined by a set of polynomials used for constructing a lattice. We first prove that this bound is not satisfied for $U \geq N^{1/D}$ if the lattice is constructed based on "standard" integer coefficient polynomials. This paper establishes the notion of standard polynomials, in short, it is a natural generalization of a way to select polynomials that have been used in the previous work. We then consider a non-standard lattice construction, that is, constructing a lattice based on non-standard integer coefficient polynomials. We show that any non-standard construction indeed leads either (i) a reduction of the original equation (1) to a strictly simpler one, or (ii) derives a non-trivial factor of the modulo N that we assume difficult to compute. Moreover, neither reduction requires the Coppersmith technique. That is, we show that such a non-standard construction will lead a better way to solve the original problem than the Coppersmith technique. Thus, from these results, we can claim that the range larger than $N^{1/D}$ cannot be achieved by the Coppersmith technique using any lattice construction. Note that the lattice condition is sufficient, and the Coppersmith technique sometimes works even if the lattice condition is not satisfied. We thus discuss the tightness of the lattice condition for showing the limits of the Coppersmith technique. The lattice condition is derived from two inequalities — one for a key algebraic property and one for the length of a short lattice vector. While it seems difficult to show that they are tight, we justify our analysis from the following two points: (i) a limit similar to $U < N^{1/D}$ still holds unless significantly better inequalities can be used, and (ii) computer experiments indicate that these inequalities cannot be significantly improved.

Another contribution of this paper is description of a method for improving lattices. To show the limitations of standard lattice construction, we give a method for decreasing the determinant of a given lattice. Thus, this method may be used to improve lattices in the Coppersmith technique It should also be noted that this method can be extended for multivariate situations, though we state the procedure for the univariate situation.

Related Work: Before Coppersmith's work, there were similar ideas for attacking cryptographic schemes. Vallée, Girault and Toffin [26] proposed a lattice based attack for the Okamoto-Shiraishi signature scheme [22] that uses a quadratic inequality. Håstad [13] also proposed a procedure for solving simultaneous modular equations by converting them to one modular equation. Based on such previous works, Coppersmith [7] proposed his method for solving general univariate equations and its application for recovering an RSA message with its $(1-1/e)$-fraction of MSBs. After his work, Howgrave-Graham [14] reformulated the technique, and many applications have been proposed in the cryptographic area. Shoup [25], for instance, gave an interesting application for proving the security of the RSA-OAEP encryption scheme with $e = 3$. Our result would show the limitations of these approaches. For example, the RSA message cannot be recovered from its MSBs that have length of less than $(1 - 1/e)\log_2 N$ by using the direct usage of the Coppersmith technique.

The rest of this paper is as follows. Section 2 provides some necessary technical background. Section 3 follows [14] and precisely defines the Coppersmith technique and the lattice condition. Section 4 derives a necessary condition for the solution range to achieve the lattice condition under the standard lattice construction. Section 5 discusses what we are able to compute from a non-standard construction. Section 6 discusses the tightness of our analysis and reports on computer experiments. Finally, Section 7 concludes the paper with remarks. Most of technical arguments as well as some related topics are omitted due to the space limit, see the full version in ePrint Archive [1].

2 Preliminaries

Here we introduce definitions and technical lemmas. For any positive integer n, let $[n]$ to denote the set $\{1,\dots,n\}$. A vector consisting of $s \geq 2$ coordinates $a_1,\dots,a_s$ is denoted as $[a_1,\dots,a_s]$. On the other hand, for polynomials $f_1(x),\dots,f_s(x)$, we use $(f_1,\dots,f_s)$ to denote their sequence.

Let $\mathbb{Z}[x]$ denote the ring of integer coefficient univariate polynomials. Denote by $\mathbb{Z}_N$ the ring $\mathbb{Z}/N\mathbb{Z}$, and let $\mathbb{Z}_N[x]$ denote the ring of polynomials whose coefficients are in $\mathbb{Z}_N$. Use $\mathbb{Z}_N^\times$ to denote the set of units; i.e., elements that have inverses, in $\mathbb{Z}_N$. Based on this, we denote the set of units in $\mathbb{Z}_N[x]$ by $\mathbb{Z}_N[x]^\times$. We also use $\mathbb{M}_N[x]$ to denote the set of *monic polynomials* in $\mathbb{Z}_N[x]$; that is, the polynomials whose leading coefficients are one.

By $\equiv_N$ we denote the equivalence between two polynomials under modulo N; that is, for two polynomials $f(x) = \sum_{i=0}^d a_i x^i$ and $g(x) = \sum_{i=0}^e b_i x^i$, we write $f(x) \equiv_N g(x)$ if $a_i \equiv b_i \pmod N$ for any i, $0 \le i \le \max(d, e)$. Here we understand that $a_i = 0$ (resp. $b_i = 0$) for $i > d$ (resp. $i > e$.)

For any polynomial $f(x)$, we use $\deg(f)$ and $\mathrm{lc}(f)$ to denote its degree and the leading coefficient, respectively. For any positive integer c and any polynomial $f(x)$, we define $\mathrm{ord}_c(f)$ by the largest integer r such that $f(x) \equiv_{c^r} 0$ holds.

Howgrave-Graham [14] reformulated the Coppersmith technique and gave a lemma that plays a key role in analysis of the technique. We first introduce the notion of U-norm for simplifying notation. Define the *U-norm* of a polynomial $f(x) = \sum_{i=0}^d a_i x^i$ by $||f||_U = \sqrt{\sum_{i=0}^d (a_i U^i)^2}$. Thus, this is just the length of the coefficient vector of $f(Ux)$.

Lemma 1. *(***Howgrave-Graham** *[14]) Consider any polynomial $f(x) \in \mathbb{Z}[x]$ consisting of w monomials. Let W be a non-negative integer satisfying*

$$||f||_U < W/\sqrt{w}. \tag{2}$$

Then we have

$$\forall v,\ |v| < U\ [\ f(v) \equiv 0 \pmod W \Leftrightarrow f(v) = 0\ ]. \tag{3}$$

In the Coppersmith technique, we need to find a polynomial having a small U-norm via a lattice reduction algorithm. First, we introduce a way to relate polynomials with vectors (and a lattice). Consider any polynomial $f(x) = \sum_{i=0}^d a_i x^i$. Its *vectorization* is a vector $\mathcal{V}(f, U)$ defined by $[a_0, a_1 U, \ldots, a_d U^d]$. On the other hand, as its inverse transformation, for a given vector $\mathbf{v}$, we define the *functionalization* of $\mathbf{v}$ as a unique polynomial $f(x)$ such that $\mathbf{v} = \mathcal{V}(f, U)$ holds, and denote it by $\mathcal{F}(\mathbf{v}, U)$. Note that this is undefined if no such $f(x)$ exists. $\mathcal{V}(f, U)$ and $\mathcal{F}(\mathbf{v}, U)$ are clearly linear mapping w.r.t. polynomials and vectors, respectively. The Euclidean norm of $\mathbf{v}$ $(= \mathcal{V}(f, U))$ is equal to the U-norm of $f(x)$ $(= \mathcal{F}(\mathbf{v}, U))$. This relation is the motivation for our transformation.

For any $k \ge 1$ and any $w \ge k$, let $\mathbf{b}_1, \ldots, \mathbf{b}_k \in \mathbb{R}^w$ be linearly independent vectors. Then, the *lattice* spanned by these vectors is defined by the set $\{a_1\mathbf{b}_1 + \cdots + a_k\mathbf{b}_k \mid a_1, \ldots, a_k \in \mathbb{Z}\}$. We use the notation $L(\mathbf{b}_1, \ldots, \mathbf{b}_k)$ to denote it. The set of vectors $\{\mathbf{b}_1, \ldots, \mathbf{b}_k\}$ is called a *basis* of this lattice. We sometimes omit the basis if it is clear from the context. An element of a lattice is called a *lattice vector*. The *determinant* of the lattice is defined by $\det(L) = \prod_{i=1}^k |\mathbf{b}_i^*|$, where $\{\mathbf{b}_1^*, \ldots, \mathbf{b}_k^*\}$ is the Gram-Schmidt basis. Here, the notation $|\cdot|$ denotes the standard Euclidean norm.

We need to compute a non-zero short vector in a lattice, which can be computed by a lattice basis reduction algorithm such as the LLL algorithm [18].

For a lattice L, let $\mathbf{v}_1$ denote the first vector in the basis computed by the LLL algorithm. Then it has been known [18] that this length is bounded by

$$|\mathbf{v}_1| \leq A(k)\det(L)^{1/k}, \tag{4}$$

where $A(k)$ is a constant that only depends on k. It has been shown that (4) holds with $A(k) = 2^{(k-1)/4}$. Note that this upper bound may not be tight, and that better algorithms have improved this. On the other hand, it has been observed [11] that for any polynomial-time lattice reduction algorithm, we have $\delta > 1$ such that $|\mathbf{v}_1| \approx \delta^k \det(L)^{1/k}$ holds. (Here by "polynomial-time" we mean polynomial-time w.r.t. k and $\log B \overset{\text{def}}{=} \log \max_i |\mathbf{b}_i|$.)

Remarks on the Problem Setting: We consider the problem of finding all solutions for (1) within the range of $|x| < U$ for a given parameter U. We call (1) the *target equation* and the range $|x| < U$ the *target range*. Throughout this paper, we fix the usage of symbols F, D, N, and U. We use the standard unit cost time complexity, and we evaluate complexity measures in terms of $\log N$, because we can assume that $D \leq \text{poly}(\log N)$ and $U < N$. Hence, by "polynomial-time" we mean a time polynomial in $\log N$ unless otherwise stated.

We assume that N is a large composite number whose non-trivial factor cannot be found during the computation that we investigate. This is because the factor of N would give the complete solution to the original problem in almost all applications of the Coppersmith technique. Thus, we can assume that all numbers that appear during the computation are coprime to N. This is used in the argument of Section 5. Because of this, we can assume that the coefficient a_D of x^D of $F(x)$ is one as stated in (1) since otherwise we can "divide" it by multiplying a_D^{-1} modulo N because a_D must be coprime to N.

3 Framework for the Coppersmith Technique

This section introduces our framework for discussing the Coppersmith technique for a univariate equation. As mentioned in the above section, for a given target equation (1) and a target range U, our task is to find all solutions within the target range. For this task, we formulate the Coppersmith technique as an algorithm stated as Figure 1 by following Howgrave-Graham's reformulation [14].

Remarks may be necessary for some steps of the algorithm. First note that the algorithm is given two parameters $k \geq 1$ and $m \geq 2$, which are chosen (often heuristically) for the target equation. They are usually chosen as small because the time complexity of the original LLL algorithm [18] is $O(k^5 u \log^3 B)$ (e.g., [21, Chapter 5]), where u and B are the dimension of each vector, and $\log \max_i ||\mathbf{b}_i||$, respectively. We can at least assume that these parameters relating to time complexity are poly$(\log N)$, and this assumption is sufficient for our analysis. Thus, throughout the following discussion, we will consider any $k, m, u, \log B \leq (\log N)^c$ for $c > 0$ and let them be fixed.

At Step 1, we define linearly independent polynomials $g_1(x), \ldots, g_k(x) \in \mathbb{Z}[x]$, which we call *initial polynomials*, that satisfy

$$\forall v \; [\; F(v) \equiv 0 \pmod{N} \Rightarrow g_i(v) \equiv 0 \pmod{N^m} \;]. \tag{5}$$

```
Input   $F(x)$, $N$, $U$;     Parameters  $k \ge 1$, $m \ge 2$;
Output  All solutions of $F(x) \equiv 0 \pmod{N}$ satisfying $|x| < U$;
Step 1: Based on the input, define a sequence of linearly independent poly-
        nomials $g_1(x), \ldots, g_k(x)$ that satisfy (5);
Step 2: Define vectors $\mathbf{b}_1, \ldots, \mathbf{b}_k$ by $\mathbf{b}_i = \mathcal{V}(g_i, U)$ for $i \in [k]$, and carry out
        the LLL algorithm on $L(\mathbf{b}_1, \ldots, \mathbf{b}_k)$; Denote the obtained reduced
        basis by $\mathbf{v}_1, \ldots, \mathbf{v}_k$;
Step 3: Define a polynomial $h(x) = \mathcal{F}(\mathbf{v}_1, U)$, and solve the equation
        $h(x) = 0$ numerically; Output all integer roots within the target
        range satisfying (1);
```

Fig. 1. Outline of Coppersmith technique

As we will see, the choice of these polynomials determines the lattice used in the algorithm, and this is crucial for the performance of the algorithm. Again, they are defined somewhat heuristically in each application of the technique. We can at least assume that their degrees are bounded by $\mathrm{poly}(\log N)$. From the role of the parameter m in the above, we refer to m as an *initial exponent*.

At Step 3, we enumerate all roots of $h(x)$. Here, we simply assume that a numerical algorithm achieves this task efficiently (which is the case in the reported applications). Note that the degree of $h(x)$ is $\mathrm{poly}(\log N)$; hence, the number of roots is polynomially bounded. Finally, among all obtained roots, output integers within the target range satisfying (1).

For designing an algorithm following the outline of Figure 1, the key point is the choice of initial polynomials that determines the lattice $L(\mathbf{b}_1, \ldots, \mathbf{b}_k)$ used to compute a final $h(x)$.

Here, we follow [14] and derive a condition for initial polynomials that is sufficient for guaranteeing the correctness of the algorithm. Clearly, the algorithm works correctly when $\forall v,\ |v| < U\ [\, F(v) \equiv 0 \pmod{N} \Rightarrow h(v) = 0\,]$. On the other hand, noting that $\mathbf{v}_1$ is an integer linear combination of $\mathbf{b}_1, \ldots, \mathbf{b}_k$, we can show that $h(x)$ $(= \mathcal{F}(\mathbf{v}_1, U))$ is an integer linear combination of initial polynomials. Then, from the requirement (5) for initial polynomials it follows that

$$\forall v\ [\ F(v) \equiv 0 \pmod{N} \Rightarrow h(v) \equiv 0 \pmod{N^m}\ ].$$

Thus, our above goal is satisfied if we have

$$\forall v,\ |v| < U\ [\ h(v) \equiv 0 \pmod{N^m} \Leftrightarrow h(v) = 0\ ]. \tag{6}$$

By Lemma 1, we see that $||h||_U < N^m/\sqrt{d_{\max}+1}$ $(< N^m/\sqrt{\deg(h)+1})$ is sufficient for (6), where $d_{\max}$ is the largest degree of initial polynomials. Then, our sufficient condition for the algorithm to work is derived by evaluating $||h||_U$. First, by definition of $h(x)$ and $\mathcal{F}(\cdot, U)$, and the bound (4), we have $|\mathbf{v}_1| = ||h||_U \le A(k)\det(L)^{1/k}$, where $L = L(\mathbf{b}_1, \ldots, \mathbf{b}_k)$. Therefore, (6) is implied by

$$\det(L)^{1/k}/N^m < \left(A(k)\sqrt{d_{\max}+1}\right)^{-1}. \tag{7}$$

We call this the *lattice condition* (as a sufficient condition for the Coppersmith technique to work). Note that this is a condition for initial polynomials because the basis of L is $\{\mathcal{V}(g_i, U)\}_{i\in[k]}$. In fact, by using the initial polynomials derived from the original work by Coppersmith [7], we can show that this condition is satisfied if $U \leq N^{1/D-\varepsilon}$ (for any $\varepsilon > 0$ if N and m are large enough), thereby confirming in our framework that the original method [7] works for this range of U.

In this paper, we discuss when the condition (7) cannot be achieved by *any* lattice; i.e., any set of initial polynomials, thereby showing the technique's limit. Thus, for our following discussion, we will use a somewhat stronger and much simpler condition

$$\det(L)^{1/k}/N^m < 1 \tag{8}$$

for our target condition, which we refer as a *simplified lattice condition* (for discussing when the lattice condition cannot be achieved). Recall that $A(k) = 2^{(k-1)/4}$ for the LLL algorithm and it has been believed that $A(k)$ cannot be smaller than δ^k for $\delta > 1$ for any polynomial-time lattice reduction algorithm. Then, since $(\sqrt{d_{\max}+1}A(k))^{-1} < 1$ holds for any k, this simplified condition is necessary for (7). See section 6 for analysis with a more relaxed condition.

4 Analysis for Canonical Initial Polynomials

We consider standard lattice construction and investigate its properties. Based on this investigation, we derive a lower bound for U such that the simplified lattice condition (8) fails to hold, which we may regard as the limit of U that the Coppersmith technique works under the standard lattice construction.

Note that initial polynomials need to satisfy the condition (5). One trivial way to define such polynomials $g(x)$ is by

$$g(x) = \sum_{i=0}^{m} q_i(x) N^{m-i} (F(x))^i, \quad \text{where } q_i(x) \in \mathbb{Z}[x]. \tag{9}$$

This is an integer linear combination of polynomials that were usually called "shift polynomials" in previous work. Formally, we define the following notion.

Definition 1. *Consider the ideal $\mathfrak{a} = \langle F(x), N\rangle_{\mathbb{Z}[x]}$ in the polynomial ring $\mathbb{Z}[x]$. For any non-zero polynomial $f(x) \in \mathbb{Z}[x]$, let $\nu(f)$ be the $\mathfrak{a}$-adic order of $f(x)$, i.e., an integer s that satisfies $f(x) \in \mathfrak{a}^s$ and $f(x) \notin \mathfrak{a}^{s+1}$. For the zero polynomial, define $\nu(0) = \infty$. We say $f(x)$ is an s-canonical polynomial if $\nu(f) \geq s$.*

We simply say that $f(x)$ is *canonical* if $\nu(f) \geq m$ for the initial exponent m. Initial polynomials (or similar ones) used in the previous work are all canonical and we can consider that using canonical polynomials is a standard way to define initial polynomials. This section discusses the case in which initial polynomials are all canonical.

Consider any initial polynomials $g_1(x), \ldots, g_k(x)$. Assume that they are all canonical and linearly independent as requested in the algorithm. Consider a sequence $(g_1, \ldots, g_k)$ and denote it by $\mathbf{G}$. For any sequence $\mathbf{F} = (f_1, \ldots, f_k)$ of linearly independent polynomials, we use $L(\mathbf{F})$ to denote a lattice $L(\mathcal{V}(f_1, U), \ldots,$

$\mathcal{V}(f_k, U)$). Note that $L(\mathbf{G})$ is $L(\mathbf{b}_1, \ldots, \mathbf{b}_k)$ used in the algorithm in Figure 1 for the initial polynomials $g_1(x), \ldots, g_k(x)$.

Our task is to provide a good lower bound of $\det(L(\mathbf{G}))$. For this, we transform $\mathbf{G}$ to a polynomial sequence with some good properties for our analysis. Here, we explain the outline of our transformations and proof outline of the first main theorem. Technical discussions with necessary lemmas are given in the full-version [1].

We say a polynomial sequence $\widetilde{\mathbf{G}} = (\widetilde{g}_1, \ldots, \widetilde{g}_k)$ has a *strictly increasing degree sequence* if it holds that $\deg(\widetilde{g}_1) < \cdots < \deg(\widetilde{g}_k)$. We first transform $\mathbf{G}$ to a sequence $\widetilde{\mathbf{G}} = (\widetilde{g}_1, \ldots, \widetilde{g}_k)$ with this property while maintaining that $\det(L(\mathbf{G})) = \det(L(\widetilde{\mathbf{G}}))$ and that all $\widetilde{g}_i(x)$'s are canonical. We proved this transformation is always possible. (Roughly, $L(\mathbf{G})$ is transformed to row echelon form by the Gaussian elimination.) Next, we define a sequence $\mathbf{H} = (h_1, \ldots, h_k)$ by defining each $h_i(x) = \widetilde{g}_i(x)/N^{r_i}$ where we let $r_i = \mathrm{ord}_N(\widetilde{g}_i)$; that is, r_i is the largest integer r such that every coefficient of $\widetilde{g}_i(x)$ is divisible by N^r. Let $s_i = \nu(h_i)$. Then, since $\widetilde{g}_i(x)$ is canonical, we have $s_i + r_i = \nu(\widetilde{g}_i) \geq m$. Hence we have

$$\det(L(\mathbf{G})) = \det(L(\widetilde{\mathbf{G}})) = \left(\prod_{i=1}^{k} N^{r_i}\right) \times \det(L(\mathbf{H})) \geq \left(\prod_{i=1}^{k} N^{m-s_i}\right) \times \det(L(\mathbf{H})). \tag{10}$$

Finally, we transform $\mathbf{H}$ again to $\widehat{\mathbf{H}} = (\widehat{h}_1, \ldots, \widehat{h}_k)$ with "everywhere linearly independent reduction" which is defined as the property that for any integers $a_1, \ldots, a_k$ and any integer $B \geq 2$, $\sum_{i=1}^{k} a_i \widehat{h}_i(x) \equiv_B 0 \;\Leftrightarrow\; \forall i,\ 1 \leq i \leq k\ [\ a_i \equiv 0 \pmod{B}\ ]$. This transformation can be efficiently performed. Moreover, it can be shown that (i) $\det(L(\mathbf{H})) \geq \det(L(\widehat{\mathbf{H}}))$ and (ii) $\widehat{\mathbf{H}}$ also has a strictly increasing degree sequence. Then, the determinant term of (10) can be bounded by

$$\det(L(\mathbf{H})) \geq \det(L(\widehat{\mathbf{H}})) \geq \prod_{i=1}^{k} |\widehat{a}_i U^{\widehat{d}_i}| \geq \prod_{i=1}^{k} U^{\widehat{d}_i}, \tag{11}$$

where $\widehat{d}_i$ and $\widehat{a}_i$ are used for $\deg(\widehat{h}_i)$ and $\mathrm{lc}(\widehat{h}_i)$, respectively. Note that the second inequality is trivial if the matrix is square; we proved the general case. On the other hand, we define integer sequence $\hat{s}_1, \ldots, \hat{s}_k$ as $\hat{s}_i = \max_{(*)} \nu\left(\sum_{j=1}^{i} a_j \widehat{f}_j(x)\right)$, where the condition $(*)$ is that $a_1, \ldots, a_i \in \mathbb{Z}$, $a_i \neq 0$, and at least one a_j is not divisible by N. Then we can show that $\hat{s}_i \geq s_i$ and $\widehat{d}_i \geq \hat{s}_i D$. Thus, from (10) and (11), we have

$$\begin{aligned}\det(L(\mathbf{G})) &\geq \left(\prod_{i=1}^{k} N^{m-\hat{s}_i}\right) \times \left(\prod_{i=1}^{k} U^{\widehat{d}_i}\right) \\ &\geq \left(\prod_{i=1}^{k} N^{m-\hat{d}_i/D}\right) \times \left(\prod_{i=1}^{k} U^{\hat{d}_i}\right) \;=\; N^{mk} \cdot \left(\frac{U}{N^{1/D}}\right)^{\sum \hat{d}_i}.\end{aligned} \tag{12}$$

It is then easy to see that if $U \geq N^{1/D}$, we have $\det(L(\mathbf{G})) \geq N^{mk}$. Thus, $\det(L(\mathbf{G}))^{1/k} \geq N^m$ and the simplified lattice condition (8) fails. This proves our first main theorem.

Theorem 1. *If $U \geq N^{1/D}$, then the simplified lattice condition (8) fails to hold for any lattice constructed from canonical initial polynomials.*

5 Computation from Non-canonical Polynomials

This section considers the possibility of using non-canonical initial polynomials. Recall that in the lattice construction for a given polynomial $F(x)$ and an initial exponent m, an initial polynomial $g(x)$ is said to be canonical if $\nu(g) \geq m$ holds. A *non-canonical initial polynomial* is defined as a polynomial $g(x)$ satisfying (5) and $\nu(g) < m$ w.r.t. $F(x)$ and N. We discuss what we are able to compute if we can indeed construct a non-canonical polynomial. We show technical evidence supporting that there is no polynomial-time algorithm computing such a non-canonical polynomial for any $F(x)$, N, and m. Technically, we show that if $F(x)$ and its derivative has no common factor (a property we call "separability" following the polynomial theory over a field; e.g., [12, Def. 2]), then by using such a non-canonical initial polynomial, it is possible to compute either a non-trivial factor of N or a polynomial $G(x)$ with $\deg(G) \leq \deg(F) - 2$ that keeps the same set of solutions. This computation can also be done in polynomial-time.

5.1 Technical Preliminaries

Our investigation is based on arithmetic computations under modulo N. Since it was assumed that N is not prime; there may be points at which we need to be careful. On the other hand, as explained in the introduction, we can assume that no factor of N appears during these computations; that is, we can treat N as a prime number in the following analysis. Below, we clarify the points where careful arguments are necessary.

We use standard arithmetics in $\mathbb{Z}_N[x]$. There is no problem with addition, subtraction, and multiplication, which can be defined the same as in $\mathbb{Z}[x]$. On the other hand, the division is defined as follows. For any $f(x), g(x) \in \mathbb{Z}_N[x]$, $g(x) \not\equiv_N 0$, consider polynomials $q(x) \in \mathbb{Z}_N[x]$ and $r(x) \in \mathbb{Z}_N[x]$ such that satisfy $f(x) \equiv_N q(x)g(x) + r(x)$ and $\deg(r) < \deg(g)$ (recall that $\equiv_N$ means the polynomial equivalence under modulo N). Note that $q(x)$ and $r(x)$ are unique when the leading coefficient of $g(x)$ is coprime to N. Thus, under our assumption, we can consider $q(x)$ and $r(x)$ the *quotient* and the *remainder* of $f(x)$ divided by $g(x)$, and denote them by $\mathrm{quo}(f, g)$ and $\mathrm{rem}(f, g)$, respectively. We say that $g(x)$ *divides* $f(x)$ under modulo N or $g(x)$ an *N-divisor* of $f(x)$ (and write it as $g(x)|_N f(x)$) if $r(x) \equiv_N 0$ in the above. For any two polynomials $f(x)$ and $g(x)$, we say they are N-coprime to each other if $h(x)|_N f(x)$ and $h(x)|_N g(x)$ implies that $h(x) \in \mathbb{Z}_N[x]^\times$.

5.2 From a Non-canonical Polynomial

We discuss what we are able to compute from a non-canonical initial polynomial for given $F(x)$, N, and m. We need to assume that $F(x)$ is *separable*, that is, $F(x)$ and its derivative are N-coprime to each other. Our result is given by the following theorem. The proof is given in Appendix B in the full-version.

Theorem 2. *Assume that our target polynomial $F(x)$ is separable. For $F(x)$, N, and m, suppose that we have a non-canonical initial polynomial $g(x)$; that is, it satisfies both $\nu(g) \le m-1$ and the condition (5). Then we can compute in polynomial-time w.r.t. $\log N$ and $\deg(g)$, either a non-trivial factor of N or a polynomial $G(x)$ with $\deg(G) \le \deg(F) - 2$ satisfying*

$$\forall v \; [\; F(v) \equiv 0 \pmod{N} \Rightarrow G(v) \equiv 0 \pmod{N} \;]. \tag{13}$$

Moreover, $G(x)$ is an N-divisor of $F(x)$, and hence, the separability of $G(x)$ is immediate from that of $F(x)$.

Note that this theorem gives a polynomial-time algorithm that reduces a given target equation to a simpler one based on any non-canonical initial polynomial for the target polynomial. We expect that this reduction itself is impossible for various cases. We have shown one example from the RSA cryptography. Roughly, it claims that if we have a polynomial time algorithm for computing a non-canonical initial polynomial for any separable target equation, then it can be used to break the RSA cryptosystem with $e = \mathrm{poly}(\log N)$ efficiently; the detailed argument is given in the full-version.

6 Tightness of Our Analysis

In Section 3 we derive the (simplified) lattice condition (7) (and (8))

$$\det(L)^{1/k}/N^m < \left(A(k)\sqrt{d_{\max}+1}\right)^{-1} < 1$$

as a sufficient condition that the Coppersmith technique works, and then in the following sections, we show that this condition fails to hold for any initial polynomials when $U \ge N^{1/D}$, thereby claiming that $N^{1/D}$ is the limit of U for the Coppersmith technique. Apparently this argument is not mathematically correct, and it may be possible that the Coppersmith technique works using initial polynomials that do not satisfy the lattice condition, from which we may need to revise the limit $N^{1/D}$. Here, we provide evidence supporting that this situation is quite unlikely to happen.

Recall that the lattice condition is derived by using two inequalities. We recall these inequalities and state them by using the symbols from Section 3 used to derive the lattice condition. First is the inequality (2) stated below. This is a sufficient condition to guarantee that (3) of Lemma 1 holds and that the algorithm works with the designed initial polynomials.

$$(2)\quad ||h||_U < N^m/\sqrt{d_{\max}+1} \qquad (4)\quad |\mathbf{v}_1| \le A(k)\det(L)^{1/k}$$

The second is the upper bound (4) for the length of a short lattice vector obtained by a lattice algorithm (such as LLL) in a lattice L constructed from the initial polynomials. From these inequalities, the lattice condition $A(k)\det(L)^{1/k} < N^m/\sqrt{d_{\max}+1}$ is immediately sufficient for $||h||_U$ to satisfy (2), which guarantees that the algorithm yields a correct answer. We also assume that $(A(k)\sqrt{d_{\max}+1})^{-1} < 1$ for the simplified lattice condition.

We believe that the inequalities (2) and (4) are more or less tight for at least $h(x)$ and L appearing in the execution of the Coppersmith technique. But it is possible that they are not as sharp as we expect. We first show that a similar limit $N^{1/D+\varepsilon}$ can be derived even if much stronger inequalities could be used. For this, consider the following situation for a large β and γ: (i) the property (3) holds if $||h||_U < \beta N^m$, and (ii) a lattice algorithm obtains a short vector satisfying $||\mathbf{v}_1||_U \le \det(L)^{1/k}/\gamma$. That is, the algorithm works so long as $\det(L)^{1/k}/\gamma < \beta N^m$ ($\Leftrightarrow \det(L)^{1/k}/N^m < \beta\gamma$) holds. Note that β and γ are big numbers that may grow depending on parameters k, $d_{\max}$, etc. Thus, the lattice condition is relaxed considerably. by this condition Even for this relaxed condition, we can show that $U < N^{1/D+\varepsilon}$ is necessary to satisfy it. To see this, we use the lower bound (12) for $\det(L(\mathbf{G}))$, where $L(\mathbf{G})$ is the lattice constructed from any given canonical initial polynomials. From this bound, it is easy to see that if $U \ge N^{1/D}$,

$$\det(L(\mathbf{G}))^{1/k}/N^m \;\ge\; N^{mk}\cdot\left(\frac{U}{N^{1/D}}\right)^{\sum \hat{d}_i} \;\ge\; N^{mk}\cdot\left(\frac{U}{N^{1/D}}\right)^{k(k-1)/2}$$

holds, where the last inequality follows from the fact that $\widehat{d}_i \ge i-1$ for all $i \in [k]$ (because $\widehat{\mathbf{H}}$ has a strictly increasing degree sequence), and hence, $\sum \widehat{d}_i \ge k(k-1)/2$. Thus, even the relaxed lattice condition fails to hold if the right-hand side $\ge \beta\gamma$, which is equivalent to

$$U \;\ge\; N^{1/D+\frac{2\log(\beta\gamma)}{(k-1)\log N}}. \tag{14}$$

Therefore, even if much stronger inequalities could be used, so long as $\beta\gamma << N$, we can claim that the relaxed lattice condition fails if $U \ge N^{1/D+\varepsilon}$ for any constant $\varepsilon > 0$ and any sufficiently large N.

6.1 Justifications from Computer Experiments

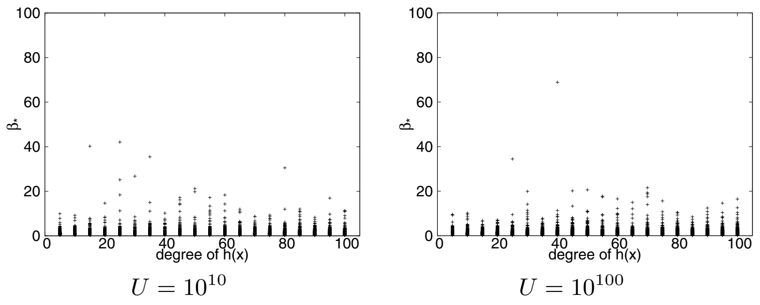
We can provide evidence from computer experiments that supports tightness of (2) and (4). First, consider (4), an upper bound for the length of a short vector given by a lattice reduction algorithm. Due to the importance of obtaining a short lattice vector, the improvement of lattice reduction algorithms and analysis of their performance have been studied extensively. Various computer experiments have also been conducted (e.g., [20,11]). From such investigation, it has been believed that for any lattice algorithm P, there is $\delta_P > 1$ such that $||\mathbf{v}_1||_U \approx \delta_P^k \det(L)^{1/k}$ holds; that is, the coefficient $A(k)$ of (4) is δ_P^k. In particular, from a large number of computer experiments, we can assume that the bound (4) holds with, at least, $A(k) = 1$.

Next consider the inequality $||h||_U < \beta_0 W$ that is a sufficient condition for

$$(3) \qquad \forall v,\ |v| < U\ [\ h(v) \equiv 0 \pmod{W} \Rightarrow h(v) = 0\ ],$$

where $\beta_0 = (\sqrt{d_{\max}+1})^{-1}$ and $W = N^m$ in our analysis. Since this is only a sufficient condition, it may be possible that (3) holds even if $||h||_U$ is much larger. While it seems quite difficult to show that this is unlikely, it is possible to relate $||h||_U$ and a condition closely related to (3). Note that if $W_* \overset{\text{def}}{=} \max_{|x|<U, x\in\mathbb{Z}} |h(x)| < W$, then (3) holds. On the other hand, (3) fails when $W = \max_{|x|<U, x\in\mathbb{Z}} |h(x)|$. This indicates that the condition $\max_{|x|<U, x\in\mathbb{Z}} |h(x)| < W$ and (3) are closely related. On the other hand, letting $\beta_* \overset{\text{def}}{=} \|h\|_U / W_*$, we have that $\max_{|x|<U, x\in\mathbb{Z}} |h(x)| < W \iff ||h||_U < \beta_* W$. Thus, we investigate how large β_* could become by computing it for randomly generated polynomials $h(x)$.

Here is the outline of our experiment. For a sufficiently large U (i.e., $U = 10^{10}$ and $= 10^{100}$) and a degree parameter d (i.e., $d = 5, 10, ..., 100$), we generate a polynomial $h(x) = \sum_{i=0}^{d} a_i x^i$ by choosing coefficients a_i randomly so that $|a_i U^i|$ is located in $[0.1BU^d, 10BU^d]$ for a large B (i.e., B is an integer sampled from $[U/2, U]$). This random generation is motivated by our observation that actual polynomials $h(x)$ derived in the Coppersmith technique have coordinates that usually satisfy $|a_i U^i|/U^{d+1} = \Theta(1)$. We then compute $W_* := \max_{|x|<U, x\in\mathbb{Z}} |h(x)|$. Since $h(x)$ is a polynomial of a relatively small degree, this computation can be performed easily by examining integer points near all $\xi \in \mathbb{R}$ satisfying $h'(\xi) = 0$. We then compute $\beta_* = \|h\|_U / W_*$. For each choice of parameters, such β_*'s are computed for 500 randomly generated polynomials. From this experiment (see Figure 2), we can claim that $\beta_* \leq 99$ and $||h||_U \geq 99W \implies \max_{|x|<U} |h(x)| > W$; in other words, it is unlikely that inequality (2) can be improved to $||h||_U < 100N^m$.



For each $U = 10^{10}$ and $= 10^{100}$, and $d = 5, 10, ..., 100$, the graph shows β_*'s computed for 500 randomly generated polynomials. The horizontal axis is the degree d of generated polynomials, and the vertical axis is β_*.

Fig. 2. Experimental values of β_*

7 Concluding Remarks

We investigated the optimality of lattice constructions used in the Coppersmith technique for finding small roots of a univariate modular equation (1). For this purpose, we provide a framework of the technique and a sufficient condition, i.e., the lattice condition (7) (and its simplified version (8)) in which the technique works. Then, for any lattice constructed from canonical initial polynomials, we derive a way to estimate a lower bound for the lattice determinant and prove that the condition fails to satisfy if $U \geq N^{1/d}$ (Theorem 1). A similar limit can be shown unless the Coppersmith technique works under a significantly relaxed condition, and we show a computer experiments indicating that it is unlikely that the lattice condition is significantly relaxed. Thus, these results are reasonable evidence of the limit of the standard lattice construction for the technique.

We also discuss the possibility of constructing a better lattice by using polynomials constructed in a non-standard way. We show that such a construction itself would lead to quite a strong method for solving the original problem or factorizing a large number.

From these results, we can claim that our bound $U < N^{1/d}$ is sharp for the direct usage of the Coppersmith technique solving (1) for various target polynomials considered in, for instance, cryptographic applications.

Coppersmith [8] employed a family of integer valued polynomials and Chebychev polynomial representations to extend the range $U < N^{1/d}$. Their approaches indeed increases the range by a poly($\log N$) factor. Although our framework considers only polynomials with integer coefficients, it can be adopted for the above framework and provide a similar limitation result. A detailed argument is given in the full-version of this paper.

Acknowledgments. We would like to thank the anonymous referees for helpful comments.

References

1. Aono, Y., Agrawal, M., Satoh, T., Watanabe, O.: On the optimality of lattices for the Coppersmith technique. Cryptology ePrint Archive, 2012/134
2. Aono, Y.: A New Lattice Construction for Partial Key Exposure Attack for RSA. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 34–53. Springer, Heidelberg (2009)
3. Boneh, D., Durfee, G.: Cryptanalysis of RSA with Private Key d Less than $N^{0.292}$. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 1–11. Springer, Heidelberg (1999)
4. Blömer, J., May, A.: New Partial Key Exposure Attacks on RSA. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 27–43. Springer, Heidelberg (2003)
5. Blömer, J., May, A.: A Tool Kit for Finding Small Roots of Bivariate Polynomials over the Integers. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 251–267. Springer, Heidelberg (2005)
6. Castagnos, G., Joux, A., Laguillaumie, F., Nguyen, P.Q.: Factoring pq^2 with Quadratic Forms: Nice Cryptanalyses. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 469–486. Springer, Heidelberg (2009)

7. Coppersmith, D.: Finding a Small Root of a Univariate Modular Equation. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 155–165. Springer, Heidelberg (1996)
8. Coppersmith, D.: Finding Small Solutions to Small Degree Polynomials. In: Silverman, J.H. (ed.) CaLC 2001. LNCS, vol. 2146, pp. 20–31. Springer, Heidelberg (2001)
9. Coron, J.-S., Joux, A., Kizhvatov, I., Naccache, D., Paillier, P.: Fault Attacks on RSA Signatures with Partially Unknown Messages. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 444–456. Springer, Heidelberg (2009)
10. Ernst, M., Jochemsz, E., May, A., de Weger, B.: Partial Key Exposure Attacks on RSA up to Full Size Exponents. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 371–386. Springer, Heidelberg (2005)
11. Gama, N., Nguyen, P.Q.: Predicting Lattice Reduction. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 31–51. Springer, Heidelberg (2008)
12. Gianni, P., Trager, B.: Square-free algorithms in positive characteristic. Applicable Algebra in Engineering, Communication and Computing 7(1), 1–14 (1996)
13. Håstad, J.: Solving simultaneous modular equations of low degree. SIAM Journal on Computing 17(2), 336–341 (1988)
14. Howgrave-Graham, N.: Finding Small Roots of Univariate Modular Equations Revisited. In: Darnell, M.J. (ed.) Cryptography and Coding 1997. LNCS, vol. 1355, pp. 131–142. Springer, Heidelberg (1997)
15. Jochemsz, E., May, A.: A Strategy for Finding Roots of Multivariate Polynomials with New Applications in Attacking RSA Variants. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 267–282. Springer, Heidelberg (2006)
16. Kunihiro, N.: Solving Generalized Small Inverse Problems. In: Steinfeld, R., Hawkes, P. (eds.) ACISP 2010. LNCS, vol. 6168, pp. 248–263. Springer, Heidelberg (2010)
17. Konyagin, S.V., Steger, T.: On polynomial congruences. Mathematical Notes 55(6), 596–600 (1994)
18. Lenstra, A.K., Lenstra Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261, 515–534 (1982)
19. Milne, J.S.: Étale cohomology. Princeton Math. Series, vol. 33. Princeton Univ. Press (1980)
20. Nguyên, P.Q., Stehlé, D.: LLL on the Average. In: Hess, F., Pauli, S., Pohst, M. (eds.) ANTS 2006. LNCS, vol. 4076, pp. 238–256. Springer, Heidelberg (2006)
21. Nguyen, P.Q., Vallée, B.: The LLL Algorithm: Survey and Applications. Springer, Heidelberg (2009)
22. Okamoto, T., Shiraishi, A.: A fast signature scheme based on quadratic inequalities. In: Proc. of the Symposium on Security and Privacy, pp. 123–132. IEEE (1985)
23. Pólya, G., Szegő, G.: Problems and Theorems in Analysis, vol. II. Springer (1976)
24. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM 21(2), 120–128 (1978)
25. Shoup, V.: OAEP Reconsidered. Journal of Cryptology 15(4), 223–249 (2002), http://shoup.net/papers/oaep.pdf
26. Vallée, B., Girault, M., Toffin, P.: How to Break Okamoto's Cryptosystem by Reducing Lattice Bases. In: Günther, C.G. (ed.) EUROCRYPT 1988. LNCS, vol. 330, pp. 281–291. Springer, Heidelberg (1988)

Revocable Identity-Based Encryption from Lattices

Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Khoa Nguyen

Division of Mathematical Sciences
School of Physical & Mathematical Sciences
Nanyang Technological University, Singapore
s080001@e.ntu.edu.sg
{hoonwei,lingsan,hxwang}@ntu.edu.sg
nguy0106@e.ntu.edu.sg

Abstract. In this paper, we present an identity-based encryption (IBE) scheme from lattices with efficient key revocation. We adopt multiple trapdoors from the Agrawal-Boneh-Boyen and Gentry-Peikerty-Vaikuntanathan lattice IBE schemes to realize key revocation, which in turn, makes use of binary-tree data structure. Using our scheme, key update requires logarithmic complexity in the maximal number of users and linear in the number of revoked users for the relevant key authority. We prove that our scheme is selective secure in the standard model and under the LWE assumption, which is as hard as the worst-case approximating short vectors on arbitrary lattices.

Keywords: Lattice-based Cryptography, Identity-based Encryption, Key Revocation.

1 Introduction

The concept of *identity-based encryption* (IBE) was proposed by Shamir [31]. It allows a sender to encrypt a message using the recipient's identity as a public key. The private key corresponding to the public key or identity is generated by a key authority (or private key generator). IBE began to be studied extensively only after the seminal work of Boneh and Franklin [11] on practical pairing-based IBE systems, see for example [12, 32, 33]. Meanwhile, there also exist proposals on IBE systems based on quadratic residuosity [16, 13], although it is still not known how to build such systems that are secure in the standard model. In recent years, however, lattice-based IBE [20, 14, 1, 2] has received considerable attention from the cryptographic research community. Lattices have becoming an attractive and powerful tool to build a broad range of cryptographic primitives [6, 23, 8, 27, 28, 19]. This is so as many lattice-based constructions are quite efficient and typically simple to implement. Moreover they are all believed to be secure against attacks using quantum computers, a property not achievable by cryptographic primitives based on factoring or discrete logarithm.

A system user's public key may need to be removed for various reasons. For example, the private key corresponding to the public key has been stolen; the user has lost her private key; or the user is no longer a legitimate system user. In these cases, it is important that the public/private key pair be revoked and replaced by new keys. In the IBE setting, Boneh and Franklin [11] suggested that the sender appends the current validity period to the intended identity during encryption and the recipient periodically receives a new private key. Unfortunately, such solution requires the key authority to perform work that is linear in the number of non-revoked users. Further, the key authority needs to create and transmit a new key to each non-revoked user through some form of authenticated and secure channel. Boldyreva, Goyal and Kumar [10] recently proposed a revocable IBE (RIBE) scheme that significantly reduces the key authority's workload (in terms of key revocation) to logarithmic (instead of linear) in the number of users, while keeping the scheme efficient for senders and receivers. Their RIBE scheme uses key revocation techniques based on binary-tree data structure, also used in [5, 25], and builds on a fuzzy IBE (FIBE) scheme introduced by Sahai and Waters [29] that is secure in the selective-ID model. Note that Boldyreva et al's RIBE is the first IBE scheme that supports non-interactive key revocation (in the sense that non-revoked users need not interact with the key authority in order to update their keys). Prior to their work, all revocation techniques require interactions between users and the key authority or some kind of trusted hardware. Moreover, the use of a binary-tree reduces the amount of work in key update from being proportional to logarithmic complexity in the maximal number of users. Libert and Vergnaud [22] subsequently proposed an RIBE scheme in the adaptive-ID model using similar key revocation techniques as with [10]. However, instead of making use of an FIBE scheme, they adopt a variant [21] of the Waters IBE scheme [32]. Nevertheless, all the above RIBE schemes are constructed from bilinear pairings.

In the spirit of expanding the study of lattice-based IBE, we show, in this paper, how to construct an RIBE scheme in the lattice setting.

1.1 Our Results

Our construction of RIBE from lattices makes use of the following building blocks: (i) lattice IBE proposed by Agrawal, Boneh, and Boyen [1]; (ii) trapdoors for lattice IBE proposed by Gentry, Peikerty, and Vaikuntanathan [20]; and (iii) the binary-tree data structure for key update used in [5, 25, 10, 22]. More specifically, we extend the lattice IBE scheme of [1] with trapdoors from [20] to enable non-interactive key revocation. As with prior work, the binary-tree data structure is used to improve the efficiency of secret key update, allowing us to achieve key update with logarithmic complexity in the maximal number of users and linear in the number of revoked users for the key authority.

We note that our RIBE scheme is not a straightforward combination of the aforementioned building blocks because we require that our user public key comprises two components: identity and time, in order to obtain the "non-interactive" property. Hence, our construction requires two instances of Agrawal et al.'s IBE scheme to deal with users' identities and times respectively. Further, we require a random n-vector $\mathbf{u}$ to be part of the public parameters that plays the role of linking identity to time for each node associated to the binary-tree. Briefly speaking, this can be achieved by randomly splitting the vector $\mathbf{u}$ into two vectors $\mathbf{u}_1, \mathbf{u}_2$ for each node to indicate identity and time, respectively.

We prove that our RIBE scheme is selective secure in the standard model and under the LWE assumption, which is as hard as the worst-case approximation of short vectors on arbitrary lattices [28, 26]. Simply applying the simulation techniques of [1] to our lattice setting does not work, since the trapdoors can respond to only all key (short vector) queries for all identities $\mathrm{id} \neq \mathrm{id}^*$ and times $\mathrm{t} \neq \mathrm{t}^*$. We address this by adopting the simulation trapdoors of [20]. That is, we sample a short vector from some distribution to generate $\mathbf{u}_1$ or $\mathbf{u}_2$ instead of generating both $\mathbf{u}_1$ and $\mathbf{u}_2$ randomly for each node in the simulation. Such $\mathbf{u}_1$ or $\mathbf{u}_2$ is indistinguishable from the uniform distribution. The sampled short vectors will be used to respond to a query for the challenge identity id^* and a query for the challenge time t^*.

1.2 Related Work

Our work, which focuses on how to construct revocable IBE from lattices, is concurrent but independent from the very recent proposal of lattice FIBE in [4]. There is some similarity between [4] and our work, that is, attributes are embedded in the shares $\mathbf{u}_i$ of vector $\mathbf{u}$ in the construction and the shares $\mathbf{u}_i$ of the challenge attributes are generated by sampling random short vectors in the simulation. However, our approach is different in the sense that we directly and randomly split the vector $\mathbf{u}$ instead of using the Shamir secret-sharing scheme and the Lagrange interpolation formula. This makes our system more efficient. Another difference is that we make use of only one matrix associated with a trapdoor basis instead of ℓ matrices, where ℓ is the maximal number of attributes. Our method could also be applied to their large universe scheme and this significantly reduces the size of the master secret key.

We note that the idea of using more than one trapdoor in the keys has also been mentioned in Agrawal et al.'s hierarchical IBE (HIBE) scheme [3] and in the completely non-malleable public-key encryption scheme by Sapehi et al. [30].

2 Definitions

2.1 Notation

Throughout the paper we say that a function $\epsilon : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is negligible if $\epsilon(n)$ is smaller than all polynomial fractions for sufficiently large n. We say that an event happens with overwhelming probability if it happens with probability at least $1 - \epsilon(n)$ for some negligible function ϵ. We say that integer vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n \in \mathbb{Z}^m$ are $\mathbb{Z}_q$-linearly independent if they are linearly independent when reduced modulo q.

The statistical distance of two random variables X and Y over a discrete domain Ω is defined as $\Delta(X; Y) := \frac{1}{2} \sum_{s \in \Omega} |\Pr[X = s] - \Pr[Y = s]|$. We say that X is δ-uniform over Ω if $\Delta(X; U_\Omega) \leq \delta$ where U_Ω is a uniform random variable over Ω. Let $X(\lambda)$ and $Y(\lambda)$ be ensembles of random variables, we say that X and Y are statistically close if $d(\lambda) := \Delta(X(\lambda); Y(\lambda))$ is a negligible function of λ.

2.2 Syntax of RIBE

Here, we recall the definitions of security for RIBE as defined in [10].

Definition 1. *An identity-based encryption with efficient revocation or simply revocable IBE scheme has seven probabilistic polynomial-time (PPT) algorithms* **Setup***,* **PriKeyGen***,* **KeyUpd***,* **DecKeyGen***,* **Enc***,* **Dec***, and* **KeyRev** *with associated message space $\mathcal{M}$, identity space $\mathcal{I}$, and time space $\mathcal{T}$.*

- **Setup**$(1^\lambda, N)$ *takes as input a security parameter λ and a maximal number of users N. It outputs a public parameters PP, a master key MK, a revocation list RL (initially empty), and a state ST. (This is run by the key authority.)*
- **PriKeyGen**$(\mathsf{PP}, \mathsf{MK}, \mathrm{id}, \mathsf{ST})$ *takes as input the public parameters PP, the master key MK, an identity $\mathrm{id} \in \mathcal{I}$, and the state ST. It outputs a private key $\mathsf{SK}_{\mathrm{id}}$ and an updated state ST. (This is stateful and run by the key authority.)*
- **KeyUpd**$(\mathsf{PP}, \mathsf{MK}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$ *takes as input the public parameters PP, the master key MK, a key update time $\mathrm{t} \in \mathcal{T}$, the revocation list RL, and the state ST. It outputs a key update KU_{t}. (This is run by the key authority.)*
- **DecKeyGen**$(\mathsf{SK}_{\mathrm{id}}, \mathsf{KU}_{\mathrm{t}})$ *takes as input a private key $\mathsf{SK}_{\mathrm{id}}$ and key update KU_{t}. It outputs a decryption key $\mathsf{DK}_{\mathrm{id,t}}$ or a special symbol $\perp$ indicating that* id *was revoked. (This is deterministic and run by the receiver.)*
- **Enc**$(\mathsf{PP}, \mathrm{id}, \mathrm{t}, \mathrm{m})$ *takes as input the public parameters PP, an identity $\mathrm{id} \in \mathcal{I}$, an encryption time $\mathrm{t} \in \mathcal{T}$, and a message $\mathrm{m} \in \mathcal{M}$. It outputs a ciphertext $\mathsf{CT}_{\mathrm{id,t}}$. (This is run by the sender. For simplicity and wlog we assume that* id, t *are efficiently computable from $\mathsf{CT}_{\mathrm{id,t}}$.)*
- **Dec**$(\mathsf{PP}, \mathsf{DK}_{\mathrm{id,t}}, \mathsf{CT}_{\mathrm{id,t}})$ *takes as input the public parameters PP, a decryption key $\mathsf{DK}_{\mathrm{id,t}}$, and a ciphertext $\mathsf{CT}_{\mathrm{id,t}}$. It outputs a message $\mathrm{m} \in \mathcal{M}$. (This is deterministic and run by the receiver.)*
- **KeyRev**$(\mathrm{id}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$ *takes as input an identity to be revoked $\mathrm{id} \in \mathcal{I}$, a revocation time $\mathrm{t} \in \mathcal{T}$, the revocation list RL, and the state ST. It outputs an updated revocation list RL. (This is stateful and run by the key authority.)*

The consistency condition requires that for all $\lambda \in \mathbb{N}$ and polynomials (in λ) N, all $(\mathsf{PP}, \mathsf{MK})$ output by **Setup**, all $\mathrm{m} \in \mathcal{M}, \mathrm{id} \in \mathcal{I}, \mathrm{t} \in \mathcal{T}$ and all possible valid states ST and revocation lists RL, if identity id was not revoked by time t then, for $(\mathsf{SK}_{\mathrm{id}}, \mathsf{ST}) \overset{\$}{\leftarrow}$ **PriKeyGen**$(\mathsf{PP}, \mathsf{MK}, \mathrm{id}, \mathsf{ST})$, $\mathsf{KU}_{\mathrm{t}} \overset{\$}{\leftarrow}$ **KeyUpd**$(\mathsf{PP}, \mathsf{MK}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$, $\mathsf{DK}_{\mathrm{id,t}} \leftarrow$ **DecKeyGen**$(\mathsf{SK}_{\mathrm{id}}, \mathsf{KU}_{\mathrm{t}})$ we have **Dec**$(\mathsf{PP}, \mathsf{DK}_{\mathrm{id,t}},$ **Enc**$(\mathsf{PP}, \mathrm{id}, \mathrm{t}, \mathrm{m})) = \mathrm{m}$.

Boldyreva et al. formalized and defined the selective-revocable-ID security in the following experiments. Their definition captures not only the standard notion of selective-ID security but also takes into account key revocation:

Initial: The adversary first outputs the challenge identity id^* and time t^*, and also some information state it wants to preserve.
Setup: It is run to generate public parameters PP, a master key MK, a revocation list RL (initially empty), and a state ST. Then PP is given to $\mathcal{A}$.
Query: $\mathcal{A}$ may adaptively make a polynomial number of queries of the following oracles (the oracles share state):

- The private key generation oracle **PriKeyGen**$(\cdot)$ takes as input an identity id and runs **PriKeyGen**$(\mathsf{PP}, \mathsf{MK}, \mathrm{id}, \mathsf{ST})$ to return a private key $\mathsf{SK}_{\mathrm{id}}$.
- The key update generation oracle **KeyUpd**$(\cdot)$ takes as input time t and runs **KeyUpd**$(\mathsf{PP}, \mathsf{MK}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$ to return a key update KU_{t}.
- The revocation oracle **KeyRev**$(\cdot)$ takes as input an identity id and time t and runs **KeyRev**$(\mathrm{id}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$ to update RL.

Challenge: $\mathcal{A}$ outputs the same length challenge $\mathrm{m}_{(0)}, \mathrm{m}_{(1)} \in \mathcal{M}$. A random bit β is chosen. $\mathcal{A}$ is given **Enc**$(\mathsf{PP}, \mathrm{id}^*, \mathrm{t}^*, \mathrm{m}_{(\beta)})$.
Guess: The adversary may continue to make a polynomial number of queries of the following oracles as in query phase and outputs a bit β', and succeeds if $\beta' = \beta$.
The following restrictions must always hold:

1. **KeyUpd**$(\cdot)$ and **KeyRev**$(\cdot, \cdot)$ can be queried on time which is greater than or equal to the time of all previous queries, i.e., the adversary is allowed to query only in non-decreasing order of time. Also, the oracle **KeyRev**$(\cdot, \cdot)$ cannot be queried at time t if **KeyUpd**$(\cdot)$ was queried on t.
2. If **PriKeyGen**$(\cdot)$ was queried on identity id^* then **KeyRev**$(\cdot, \cdot)$ must be queried on $(\mathrm{id}^*, \mathrm{t})$ for some $\mathrm{t} \leq \mathrm{t}^*$, i.e., identity id^* must be in RL when **KeyUpd**$(\cdot)$ is queried at time t^*.

We define the advantage of $\mathcal{A}$ as the quantity

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{IND\text{-}sRID\text{-}CPA}}(\lambda) := \Pr[\beta' = \beta] - 1/2.$$

Definition 2. *The scheme RIBE is said to be* $\mathsf{IND\text{-}sRID\text{-}CPA}$ *secure if the function* $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{IND\text{-}sRID\text{-}CPA}}(\lambda)$ *is negligible in* λ *for any efficient* $\mathcal{A}$ *and polynomial* n.

3 Background on Lattices

In this section, we describe the required concepts from lattices.

3.1 Integer Lattices

Let $\mathbf{B} := [\mathbf{b}_1 | \ldots | \mathbf{b}_m] \in \mathbb{R}^{m \times m}$ be an $m \times m$ matrix whose columns are linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_m \in \mathbb{R}^m$. The m-dimensional full-rank lattice Λ generated by $\mathbf{B}$ is the set,

$$\Lambda := \mathcal{L}(\mathbf{B}) := \left\{ \mathbf{y} \in \mathbb{R}^m \text{ s.t. } \exists \mathbf{s} \in \mathbb{Z}^m, \mathbf{y} = \mathbf{B}\mathbf{s} = \sum_{i=1}^{m} \mathbf{s}_i \mathbf{b}_i \right\}$$

Here, we are interested in integer lattices, i.e, when $\mathcal{L}$ is a subset of $\mathbb{Z}^m$.

Definition 3. *For a prime q, $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{u} \in \mathbb{Z}_q^n$, define:*

$$\Lambda_q^{\perp}(\mathbf{A}) := \{\mathbf{e} \in \mathbb{Z}^m \text{ s.t. } \mathbf{Ae} = \mathbf{0} \,(\bmod q)\}$$
$$\Lambda_q^{\mathbf{u}}(\mathbf{A}) := \{\mathbf{e} \in \mathbb{Z}^m \text{ s.t. } \mathbf{Ae} = \mathbf{u} \,(\bmod q)\}$$

3.2 The Gram-Schmidt Norm and Trapdoors for Lattices

Let S be a set of vectors $S := \{\mathbf{s}_1, \ldots, \mathbf{s}_k\}$ in $\mathbb{R}^m$, we use $\|S\|$ to denote the Euclidean norm of the longest vector in S, i.e., $\|S\| := \mathsf{max}_i \sqrt{s_{i,1}^2 + \ldots + s_{i,m}^2}$ for $1 \leq i \leq k$, where $\mathbf{s}_i := (s_{i,1}, \ldots, s_{i,m})$. We use $\tilde{S} := \{\tilde{\mathbf{s}}_1, \ldots, \tilde{\mathbf{s}}_k\} \subset \mathbb{R}^m$ to denote the Gram-Schmidt orthogonalization of the vectors $\mathbf{s}_1, \ldots, \mathbf{s}_k$ in that order. We refer to $\|\tilde{S}\|$ as the Gram-Schmidt norm of S.

The problem of generating a random lattice $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ together with a full short basis $\mathbf{T_A}$ of $\Lambda_q^{\perp}(\mathbf{A})$ has been previously investigated by [7, 9]. Here we use a better result with tighter parameters which was recently discovered by Micciancio and Peikert [24].

Theorem 1. *Let $n \geq 1$, $q \geq 2$ be integers and $m = \lceil 2n \log q \rceil$. There is an efficient PPT algorithm* $\mathsf{TrapGen}(q, n)$ *that outputs a pair $(\mathbf{A} \in \mathbb{Z}_q^{n \times m}, \mathbf{T_A} \in \mathbb{Z}_q^{m \times m})$ such that $\mathbf{A}$ is statistically close to a uniform matrix in $\mathbb{Z}_q^{n \times m}$ and $\mathbf{T_A}$ is a basis for $\Lambda_q^{\perp}(\mathbf{A})$ satisfying $\|\widetilde{\mathbf{T_A}}\| \leq \mathcal{O}(\sqrt{n \log q})$ and $\|\mathbf{T_A}\| \leq \mathcal{O}(n \log q)$ with all but negligible probability in n.*

3.3 Discrete Gaussians

Let Λ be an m-dimensional lattice. For any vector $\mathbf{c} \in \mathbb{R}^m$ and any positive parameter $\sigma \in \mathbb{R}_{>0}$, define:

- $\rho_{\sigma,\mathbf{c}}(\mathbf{x}) := \exp\left(-\pi \frac{\|\mathbf{x}-\mathbf{c}\|^2}{\sigma^2}\right)$: a Gaussian-shaped function on $\mathbb{R}^m$ with center $\mathbf{c}$ and parameter σ,
- $\rho_{\sigma,\mathbf{c}}(\Lambda) := \sum_{\mathbf{x} \in \Lambda} \rho_{\sigma,\mathbf{c}}(\mathbf{x})$: the (always converging) sum of $\rho_{\sigma,\mathbf{c}}$ over Λ,
- $\mathcal{D}_{\Lambda,\sigma,\mathbf{c}}$: the discrete Gaussian distribution over Λ with parameters σ and center $\mathbf{c}$,

$$\forall \mathbf{y} \in \Lambda, \quad \mathcal{D}_{\Lambda,\sigma,\mathbf{c}}(\mathbf{y}) := \frac{\rho_{\sigma,\mathbf{c}}(\mathbf{y})}{\rho_{\sigma,\mathbf{c}}(\Lambda)}.$$

For notational convenience, we abbreviate $\rho_{\sigma,\mathbf{0}}$ and $\mathcal{D}_{\Lambda,\sigma,\mathbf{0}}$ as ρ_σ and $\mathcal{D}_{\Lambda,\sigma}$.

The following lemmas from [20] is essential for our security proof.

Lemma 1. *There is an efficient PPT algorithm* $\mathsf{SampleGaussian}$ *that, given a basis $\mathbf{B}$ of an m-dimensional lattice $\Lambda = \mathcal{L}(\mathbf{B})$, a parameter $\sigma \geq \|\widetilde{\mathbf{B}}\| \cdot \omega(\sqrt{\log m})$, and a center $\mathbf{c} \in \mathbb{R}^m$, outputs a sample from a distribution that is statistically close to $\mathcal{D}_{\Lambda,\sigma,\mathbf{c}}$.*

Let $\mathbf{B}_z$ be the standard basis for $\mathbb{Z}^m$, we use the $\mathsf{SampleGaussian}(\mathbf{B}_z, \sigma, 0)$ algorithm to sample from distribution $\mathcal{D}_{\mathbb{Z}^m,\sigma}$.

Lemma 2. *Let n and q be positive integers with q prime, and let $m \geq 2n \log q$. Then for all but a $2q^{-n}$ fraction of all $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and for any $\sigma \geq \omega(\sqrt{\log m})$, the distribution of the syndrome $\mathbf{u} = \mathbf{A}\mathbf{e}$ mod q is statistically close to uniform over $\mathbb{Z}_q^n$, where $\mathbf{e}$ is from $\mathcal{D}_{\mathbb{Z}^m,\sigma}$.*

3.4 Sampling Algorithms

The following SampleLeft [14, 1] and SampleRight [1] algorithms will be used to sample short vectors in our construction and in the simulation, respectively. Let $\mathbf{A}$ and $\mathbf{C}$ be matrices in $\mathbb{Z}_q^{n \times m}$ and let $\mathbf{R}$ be a matrix in $\{-1, 1\}^{m \times m}$. By using either a trapdoor for $\Lambda_q^{\perp}(\mathbf{A})$ or a trapdoor $\Lambda_q^{\perp}(\mathbf{C})$, we can sample a short vector $\mathbf{e}$ in $\Lambda_q^{\perp}(\mathbf{F})$ for some $\mathbf{u}$ in $\mathbb{Z}_q^n$, where $\mathbf{F} := (\mathbf{A}|\mathbf{AR} + \mathbf{C}) \in \mathbb{Z}_q^{n \times 2m}$. With appropriate parameters, the distribution of $\mathbf{e}$ produced by these two algorithms is statistically indistinguishable.

Theorem 2. *Let $q > 2$ and $m > n$. Then there is an efficient PPT algorithm* SampleLeft *that takes as input a rank n matrix $\mathbf{A}$ in $\mathbb{Z}_q^{n \times m}$, a matrix $\mathbf{M}$ in $\mathbb{Z}_q^{n \times m_1}$, a "short" basis $\mathbf{T_A}$ of $\Lambda_q^{\perp}(\mathbf{A})$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$, and a gaussian parameter $\sigma > \|\widetilde{\mathbf{T_A}}\| \cdot \omega(\sqrt{\log(m + m_1)})$. It outputs a vector $\mathbf{e} \in \mathbb{Z}^{m+m_1}$ distributed statistically close to $\mathcal{D}_{\Lambda_q^{\mathbf{u}}(\mathbf{F}_1),\sigma}$ where $\mathbf{F}_1 := (\mathbf{A}|\mathbf{M})$. In particular, $\mathbf{e} \in \Lambda_q^{\mathbf{u}}(\mathbf{F}_1)$.*

Theorem 3. *Let $q > 2$ and $m > n$. There is an efficient PPT algorithm* SampleRight *that takes as input matrices $\mathbf{A}, \mathbf{C}$ in $\mathbb{Z}_q^{n \times m}$ where $\mathbf{C}$ is rank n, a uniform random matrix $\mathbf{R} \in \{-1, 1\}^{m \times m}$, a basis $\mathbf{T_C}$ of $\Lambda_q^{\perp}(\mathbf{C})$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$, and a gaussian parameter $\sigma > \|\widetilde{\mathbf{T_C}}\| \cdot \sqrt{m}\omega(\log(m))$. It outputs a vector $\mathbf{e} \in \mathbb{Z}^{2m}$ distributed statistically close to $\mathcal{D}_{\Lambda_q^{\mathbf{u}}(\mathbf{F}_2),\sigma}$ where $\mathbf{F}_2 := (\mathbf{A}|\mathbf{AR} + \mathbf{C})$. In particular, $\mathbf{e} \in \Lambda_q^{\mathbf{u}}(\mathbf{F}_2)$.*

We will also need the following lemma, generalization of the left over hash lemma due to Dodis et al. [18], in our proof.

Lemma 3. *Suppose that $m > (n + 1) \log q + \omega(\log n)$ and that q is prime. Let $\mathbf{A}, \mathbf{B}$ be matrices chosen uniformly in $\mathbb{Z}_q^{n \times m}$ and let $\mathbf{R}$ be an $m \times m$ matrix chosen uniformly in $\{1, -1\}^{m \times m}$ mod q. Then, for all vectors $\mathbf{w}$ in $\mathbb{Z}_q^m$, the distribution of $(\mathbf{A}, \mathbf{AR}, \mathbf{R}^{\top}\mathbf{w})$ is statistically close to the distribution of $(\mathbf{A}, \mathbf{B}, \mathbf{R}^{\top}\mathbf{w})$.*

3.5 The LWE Hardness Assumption

The security of our construction can be reduced to the LWE (learning with errors) problem defined by Regev [28].

Definition 4. *Consider a prime q, a positive integer n, and a distribution χ over $\mathbb{Z}_q$, all public. An $(\mathbb{Z}_q, n, \chi)$-LWE problem instance consists of access to an unspecified challenge oracle $\mathcal{O}$, being, either, a noisy pseudo-random sampler $\mathcal{O}_{\mathbf{s}}$ carrying some constant random secret key $\mathbf{s} \in \mathbb{Z}_q^n$, or, a truly random sampler $\mathcal{O}_{\$}$, whose behaviors are respectively as follows:*

– $\mathcal{O}_{\mathbf{s}}$: *outputs samples of the form* $(\mathbf{u}_i, v_i) = (\mathbf{u}_i, \mathbf{u}_i^\top \mathbf{s} + x_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$*, where,* $\mathbf{s} \in \mathbb{Z}_q^n$ *is a uniformly distributed persistent value invariant across invocations,* $x_i \in \mathbb{Z}_q$ *is a fresh sample from* χ*, and* $\mathbf{u}_i$ *is uniform in* $\mathbb{Z}_q^n$*.*
– $\mathcal{O}_{\$}$: *outputs truly uniform random samples from* $\mathbb{Z}_q^n \times \mathbb{Z}_q$*.*

The $(\mathbb{Z}_q, n, \chi)$-LWE problem allows repeated queries to the challenge oracle $\mathcal{O}$. We say that an algorithm $\mathcal{A}$ decides the $(\mathbb{Z}_q, n, \chi)$-LWE problem if $|\Pr[\mathcal{A}^{\mathcal{O}_{\mathbf{s}}} = 1] - \Pr[\mathcal{A}^{\mathcal{O}_{\$}} = 1]|$ is non-negligible for a random $\mathbf{s} \in \mathbb{Z}_q^n$.

Regev [28] and Peikert [26] showed that for some noise distribution χ, denoted $\overline{\Psi}_\alpha$, the LWE problem is at least as hard as the worst-case SIVP and GapSVP under a quantum reduction if the parameters are appropriately set.

Definition 5. *Consider a real parameter* $\alpha = \alpha(n) \in (0,1)$ *and a prime* q*. Let* $\mathbb{T} := \mathbb{R}/\mathbb{Z}$ *be the group of reals* $[0,1)$ *with addition modulo* 1*. Let* Ψ_α *be the distribution over* $\mathbb{T}$ *of a normal variable with mean* 0 *and standard deviation* $\alpha/\sqrt{2\pi}$ *then reduced modulo* 1*. Let* $\lfloor x \rceil := \lfloor x + \frac{1}{2} \rfloor$ *be the nearest integer to the real* $x \in \mathbb{R}$*. We then define* $\overline{\Psi}_\alpha$ *as the discrete distribution over* $\mathbb{Z}_q$ *of the random variable* $\lfloor xX \rceil$ *mod* q *where the random variable* $X \in \mathbb{T}$ *has distribution* Ψ_α*.*

3.6 Encoding Identities as Matrices

In our construction and proof of security, we require an injective encoding function $\mathsf{H} : \mathbb{Z}_q^n \to \mathbb{Z}_q^{n \times n}$ to map identities in $\mathbb{Z}_q^n$ to matrices in $\mathbb{Z}_q^{n \times n}$. Concrete construction of such a function can be found in [1, 17].

Definition 6. *Let* q *be a prime and* n *a positive integer. We say that a function* $\mathsf{H} : \mathbb{Z}_q^n \to \mathbb{Z}_q^{n \times n}$ *is an encoding with full-rank differences (FRD) if:*

1. *for all distinct* $\mathbf{u}, \mathbf{v} \in \mathbb{Z}_q^n$*, the matrix* $\mathsf{H}(\mathbf{u}) - \mathsf{H}(\mathbf{v}) \in \mathbb{Z}_q^{n \times n}$ *is full rank;*
2. H *is computable in polynomial time in* $n \log q$*.*

4 Lattice RIBE

4.1 The Binary-Tree Data Structure

Our construction makes use of binary-tree data structure, as with [5, 25, 10, 22]. We denote the binary-tree by BT and its root node by root. If ν is a leaf node then $\mathsf{Path}(\nu)$ denotes the set of nodes on the path from ν to root (both ν and root inclusive). If θ is a non-leaf node then θ_ℓ, θ_r denote the left and right child of θ, respectively. We assume that all nodes in the tree are uniquely encoded as strings, and the tree is defined by all of its node descriptions.

Each user is assigned to a leaf node ν. Upon registration, the key authority provides the user with a set of distinct private keys for each node in $\mathsf{Path}(\nu)$.

At time t, the key authority determines the minimal set Y of nodes in BT such that none of the nodes in RL with corresponding time $\leq$ t (users revoked on or before t) have any ancestor (or, themselves) in the set Y, and all other leaf nodes (corresponding to non-revoked users) have exactly one ancestor (or, themselves) in the set. This algorithm, denoted by $\mathsf{KUNodes}$, takes as input a binary tree BT, a revocation list RL and a time t) and can be formally specified as follows:

```
$\mathsf{KUNodes}(\mathsf{BT}, \mathsf{RL}, \mathrm{t})$
  $\mathsf{X}, \mathsf{Y} \leftarrow \emptyset$
  $\forall(\nu_i, \mathrm{t}_i) \in \mathsf{RL}$
    if $\mathrm{t}_i \leq \mathrm{t}$ then add $\mathsf{Path}(\nu_i)$ to $\mathsf{X}$
  $\forall\theta \in \mathsf{X}$
    if $\theta_\ell \notin \mathsf{X}$ then add $\theta_\ell$ to $\mathsf{Y}$
    if $\theta_r \notin \mathsf{X}$ then add $\theta_r$ to $\mathsf{Y}$
  If $\mathsf{Y} = \emptyset$ then add $\mathsf{root}$ to $\mathsf{Y}$
  Return $\mathsf{Y}$
```

The $\mathsf{KUNodes}$ algorithm marks all the ancestors of revoked nodes as revoked and outputs all the non-revoked children of revoked nodes.

The key authority then publishes a key update for all nodes of Y.

A user assigned to leaf ν is then able to form an effective decryption key for time t if the set Y contains a node in $\mathsf{Path}(\nu)$. By doing so, every update of the revocation list RL only requires the key authority to perform logarithmic work in the maximal number of users and linear in the number of revoked users.

4.2 The Agrawal et al. IBE Scheme

We use the Agrawal et al. lattice IBE scheme [1] as a building block for our construction. Briefly, their IBE scheme can be described as follows.

The public parameters in the scheme of [1] consist of three random $n \times m$ matrices over $\mathbb{Z}_q$ denoted by $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$ as well as a vector $\mathbf{u} \in \mathbb{Z}_q^n$. The master secret is a trapdoor $\mathbf{T_A}$ for the lattice $\Lambda_q^\perp(\mathbf{A})$. The secret key for an identity id is a short vector $\mathbf{e} \in \mathbb{Z}^{2m}$, which is generated using the $\mathsf{SampleLeft}$ algorithm of Theorem 2 and satisfies $\mathbf{F}_{\mathrm{id}}\mathbf{e} = \mathbf{u}$ in $\mathbb{Z}_q$ where $\mathbf{F}_{\mathrm{id}} := (\mathbf{A}|\mathbf{B} + \mathsf{H}(\mathrm{id})\mathbf{C}) \in \mathbb{Z}_q^{n\times 2m}$. In the security proof for a selective IBE security game, the adversary announces an identity id^* that it plans to attack. Instead of using a trapdoor for $\Lambda_q^\perp(\mathbf{A})$, it samples $\mathbf{C}$ at random and obtains a trapdoor $\mathbf{T_C}$ for $\Lambda_q^\perp(\mathbf{C})$. It also chooses the public parameter $\mathbf{A}$ at random and sets $\mathbf{B} := \mathbf{AR} - \mathsf{H}(\mathrm{id}^*)\mathbf{C}$, where $\mathbf{R}$ is a random matrix in $\{1, -1\}^{m\times m}$. Since $\mathbf{AR}$ is uniform and independent in $\mathbb{Z}_q^{n\times m}$, $\mathbf{B}$ is uniformly distributed as required. We then have

$$\mathbf{F}_{\mathrm{id}} := (\mathbf{A}|\mathbf{A}\cdot\mathbf{R} + \mathbf{C}') \in \mathbb{Z}_q^{n\times 2m},$$

where $\mathbf{C}' := (\mathsf{H}(\mathrm{id}) - \mathsf{H}(\mathrm{id}^*))\mathbf{C}$. To respond to a private key query for an identity $\mathrm{id} \neq \mathrm{id}^*$, the simulator could produce a short vector $\mathbf{e}$ satisfying $\mathbf{F}_{\mathrm{id}}\mathbf{e} = \mathbf{u}$ in $\mathbb{Z}_q$ by using the $\mathsf{SampleRight}$ algorithm of Theorem 3 and the basis $\mathbf{T_C}$. This is so since $\mathrm{id} \neq \mathrm{id}^*$ is full rank by the definition of FRD in Section 3.6 and therefore

$\mathbf{T_C}$ is also a trapdoor for the lattice $\Lambda_q^\perp(\mathbf{C}')$. When $\mathrm{id} = \mathrm{id}^*$, the matrix $\mathbf{F}_{\mathrm{id}}$ no longer depends on $\mathbf{C}$ and the simulator's trapdoor is removed. The simulator can then produce a challenge ciphertext that helps to solve the given LWE challenge.

4.3 Intuition of Our Construction

We first consider how to create a link between an identity and a time for each node. In our construction, we use two instances of Agrawal et al.'s IBE scheme and its techniques to deal with users' identities and times respectively, but require only a single random vector $\mathbf{u} \in \mathbb{Z}_q^n$ in the public parameters. We split it into two random vectors $\mathbf{u}_1, \mathbf{u}_2$ for each node corresponding to identity and time, respectively. The randomly split $\mathbf{u}$ links identity to time for each node. Moreover, our technique does not require information about $\mathbf{u}_1, \mathbf{u}_2$ to be included in ciphertexts, and hence does not increase the size of ciphertexts.

Clearly, the simulator can answer all private key queries for all identities $\mathrm{id} \neq \mathrm{id}^*$, key update queries for all time $\mathrm{t} \neq \mathrm{t}^*$ by two trapdoors $\mathbf{T}_{\mathbf{C}_1}, \mathbf{T}_{\mathbf{C}_2}$. The main difficulty in the simulation is as follows. The simulator may be required to answer either a key update query at time t^* with node in $\mathsf{Path}(\nu^*)$ or a private key query for identity id^* and a key update query at time t^* without any node in $\mathsf{Path}(\nu^*)$, where id^* is assigned in ν^* (id^* must be revoked before or at time t^*). In other words, the simulator should answer either a query for identity id^* or a query for time t^* for each node. To overcome this difficulty, we use the $\mathsf{SampleGaussian}$ algorithm of Lemma 1 to sample a short vector and generate either $\mathbf{u}_1$ or $\mathbf{u}_2$ (instead of generating one of them randomly). Such $\mathbf{u}_1$ or $\mathbf{u}_2$ is indistinguishable from the uniform distribution, which is guaranteed by Lemma 2. More precisely, there are two possibilities for those nodes in $\mathsf{Path}(\nu^*)$ (we can pick a node ν^* beforehand and assign id^* to it if necessary) depending on whether or not identity id^* will be queried:

– If identity id^* is queried, then it must be revoked before or at time t^*. In this case, we set $\mathbf{u}_1$ to be the product of $\mathbf{F}_{\mathrm{id}^*}$ and a short vector $\mathbf{e}$ sampled by $\mathsf{SampleGaussian}(\mathbf{B}_z, \sigma, 0)$.
– If identity id^* is not queried. In this case, we set $\mathbf{u}_2$ to be the product of $\mathbf{F}_{\mathrm{t}^*}$ and a short vector $\mathbf{e}$ sampled by $\mathsf{SampleGaussian}(\mathbf{B}_z, \sigma, 0)$.

For those nodes that are not in $\mathsf{Path}(\nu^*)$, we set $\mathbf{u}_2$ to be the product of $\mathbf{F}_{\mathrm{t}^*}$ and a short vector $\mathbf{e}$ sampled by $\mathsf{SampleGaussian}(\mathbf{B}_z, \sigma, 0)$. We have probability 1/2 to simulate the correct game and the adversary cannot distinguish which one is simulated.

4.4 Our RIBE Scheme

We now describe our RIBE scheme from lattices. At the end of each algorithm, we provide some intuition and/or remark (marked by the symbol "//") about the algorithm.

Setup(λ, N) On input a security parameter λ and a maximal number N of users, set the parameters q, n, m, σ, α as specified in Section 4.5 below. Next perform the following steps:

1. Use the $\mathsf{TrapGen}(q, n)$ algorithm to select a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ with a basis $\mathbf{T_A}$ for $\Lambda_q^{\perp}(\mathbf{A})$ such that $||\widetilde{\mathbf{T_A}}|| \leq \mathcal{O}(\sqrt{n \log q})$.
2. Select four uniformly random matrices $\mathbf{B}_1$, $\mathbf{B}_2$, $\mathbf{C}_1$, and $\mathbf{C}_2$ in $\mathbb{Z}_q^{n \times m}$.
3. Select a uniformly random vector $\mathbf{u} \overset{\$}{\leftarrow} \mathbb{Z}_q^n$.
4. Let RL be an empty set and BT be a binary-tree with at least N leaf nodes, set $\mathsf{ST} := \mathsf{BT}$. Select an FRD map H as defined in Section 3.6.
5. Output RL, ST, the public parameters, and the master key MK,

$$\mathsf{PP} := \{\mathsf{H}, \mathbf{A}, \mathbf{B}_1, \mathbf{B}_2, \mathbf{C}_1, \mathbf{C}_2, \mathbf{u}\}, \quad \mathsf{MK} := \{\mathbf{T_A}\}.$$

PriKeyGen$(\mathsf{PP}, \mathsf{MK}, \mathrm{id}, \mathsf{RL}, \mathsf{ST})$ On input the public parameters PP, the master key MK, an identity $\mathrm{id} \in \mathbb{Z}_q^n$, the revocation list RL, and the state ST, it picks an unassigned leaf node ν from BT and stores id in that node. It then performs the following steps:

1. For any $\theta \in \mathsf{Path}(\nu)$, if $\mathbf{u}_{\theta,1}, \mathbf{u}_{\theta,2}$ are undefined, then pick $\mathbf{u}_{\theta,1} \overset{\$}{\leftarrow} \mathbb{Z}_q^n$, set $\mathbf{u}_{\theta,2} := \mathbf{u} - \mathbf{u}_{\theta,1}$, and store them in node θ. Sample $\mathbf{e}_{\theta,1} \in \mathbb{Z}^{2m}$ as $\mathbf{e}_{\theta,1} \leftarrow \mathsf{SampleLeft}(\mathbf{A}, \mathbf{B}_1 + \mathsf{H}(\mathrm{id})\mathbf{C}_1, \mathbf{T_A}, \mathbf{u}_{\theta,1}, \sigma)$.
2. Output $\mathsf{SK}_{\mathrm{id}} := \{(\theta, \mathbf{e}_{\theta,1})\}_{\theta \in \mathsf{Path}(\nu)}$, ST.

//The algorithm computes the id-component of the decryption key for all the nodes on the path from ν to root.

KeyUpd$(\mathsf{PP}, \mathsf{MK}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$ On input the public parameters PP, the master key MK, a time $\mathrm{t} \in \mathbb{Z}_q^n$, the revocation list RL, and the state ST, it performs the following steps:

1. $\forall \theta \in \mathsf{KUNodes}(\mathsf{BT}, \mathsf{RL}, \mathrm{t})$, if $\mathbf{u}_{\theta,1}, \mathbf{u}_{\theta,2}$ are undefined, then pick $\mathbf{u}_{\theta,1} \overset{\$}{\leftarrow} \mathbb{Z}_q^n$, set $\mathbf{u}_{\theta,2} := \mathbf{u} - \mathbf{u}_{\theta,1}$, and store them in node θ. Sample $\mathbf{e}_{\theta,2} \in \mathbb{Z}^{2m}$ as $\mathbf{e}_{\theta,2} \leftarrow \mathsf{SampleLeft}(\mathbf{A}, \mathbf{B}_2 + \mathsf{H}(\mathrm{t})\mathbf{C}_2, \mathbf{T_A}, \mathbf{u}_{\theta,2}, \sigma)$.
2. Output $\mathsf{KU}_{\mathrm{t}} := \{(\theta, \mathbf{e}_{\theta,2})\}_{\theta \in \mathsf{KUNodes}(\mathsf{BT}, \mathsf{RL}, \mathrm{t})}$.

//The algorithm first finds a minimal set of nodes which contains an ancestor (or, the node itself) of all the non-revoked nodes. It then computes the t-component of the decryption key for all the nodes in that set.

DecKeyGen$(\mathsf{SK}_{\mathrm{id}}, \mathsf{KU}_{\mathrm{t}})$ On input a private secret key $\mathsf{SK}_{\mathrm{id}} := \{(i, \mathbf{e}_{i,1})\}_{i \in \mathsf{I}}$, $\mathsf{KU}_{\mathrm{t}} := \{(j, \mathbf{e}_{j,2})\}_{j \in \mathsf{J}}$ for some set of nodes I, J, it runs the following steps:

1. $\forall (i, \mathbf{e}_{i,1}) \in \mathsf{SK}_{\mathrm{id}}, (j, \mathbf{e}_{j,2}) \in \mathsf{KU}_{\mathrm{t}}$, if $\exists (i, j)$ s.t. $i = j$ then $\mathsf{DK}_{\mathrm{id,t}} \leftarrow (\mathbf{e}_{i,1}, \mathbf{e}_{j,2})$; else (if $\mathsf{SK}_{\mathrm{id}}$ and KU_{t} do not have any node in common) $\mathsf{DK}_{\mathrm{id,t}} \leftarrow \perp$.
2. Output $\mathsf{DK}_{\mathrm{id,t}}$.

// We can drop the subscripts i, j since they are equal, i.e., $\mathsf{DK}_{\mathrm{id,t}} := (\mathbf{e}_1, \mathbf{e}_2)$. The algorithm finds components of $\mathsf{SK}_{\mathrm{id}}$ and KU_{t} such that $\mathbf{F}_{\mathrm{id}}\mathbf{e}_1 + \mathbf{F}_{\mathrm{t}}\mathbf{e}_2 = \mathbf{u}$ since they are in the same node.

Enc$(\mathsf{PP}, \mathrm{id}, \mathrm{t}, \mathrm{m})$ On input the public parameters PP, an identity id, a time $\mathrm{t} \in \mathbb{Z}_q^n$, and a message m, it runs the following steps:

1. Set $\mathbf{F}_{\mathrm{id},\mathrm{t}} \leftarrow (\mathbf{A}|\mathbf{B}_1 + \mathsf{H}(\mathrm{id})\mathbf{C}_1|\mathbf{B}_2 + \mathsf{H}(\mathrm{t})\mathbf{C}_2) \in \mathbb{Z}_q^{n\times 3m}$.
2. Choose a uniformly random $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$.
3. For $i = 1, 2$, choose a uniformly random matrix $\mathbf{R}_i \xleftarrow{\$} \{-1, 1\}^{m\times m}$.
4. Choose noise $x \xleftarrow{\overline{\Psi}_\alpha} \mathbb{Z}_q$ and noise vectors $\mathbf{y} \xleftarrow{\overline{\Psi}_\alpha^m} \mathbb{Z}_q^m$ and for $i = 1, 2$ set $\mathbf{z}_i \leftarrow \mathbf{R}_i^\top \mathbf{y} \in \mathbb{Z}_q^m$. (The distribution $\overline{\Psi}_\alpha$ is as defined by Definition 5)
5. Set $c_0 \leftarrow \mathbf{u}^\top \mathbf{s} + x + \mathrm{m}\lfloor \frac{q}{2} \rfloor \in \mathbb{Z}_q$, $\mathbf{c}_1 \leftarrow \mathbf{F}_{\mathrm{id},\mathrm{t}}^\top \mathbf{s} + \begin{bmatrix} \mathbf{y} \\ \mathbf{z}_1 \\ \mathbf{z}_2 \end{bmatrix} \in \mathbb{Z}_q^{3m}$.
6. Output the ciphertext $\mathsf{CT}_{\mathrm{id},\mathrm{t}} := (c_0, \mathbf{c}_1) \in \mathbb{Z}_q \times \mathbb{Z}_q^{3m}$.

Dec$(\mathsf{PP}, \mathsf{DK}_{\mathrm{id},\mathrm{t}}, \mathsf{CT}_{\mathrm{id},\mathrm{t}})$ On input the public parameters PP, a decryption key $\mathsf{DK}_{\mathrm{id},\mathrm{t}} := (\mathbf{e}_1, \mathbf{e}_2)$, and a ciphertext $\mathsf{CT}_{\mathrm{id},\mathrm{t}} := (c_0, \mathbf{c}_1)$, it runs the following steps:

1. Parse $\mathbf{c}_1$ as $\begin{bmatrix} \mathbf{c}_{1,0} \\ \mathbf{c}_{1,1} \\ \mathbf{c}_{1,2} \end{bmatrix}$, where $\mathbf{c}_{1,i} \in \mathbb{Z}_q^m$.
2. Compute $w \leftarrow c_0 - \mathbf{e}_1^\top \begin{bmatrix} \mathbf{c}_{1,0} \\ \mathbf{c}_{1,1} \end{bmatrix} - \mathbf{e}_2^\top \begin{bmatrix} \mathbf{c}_{1,0} \\ \mathbf{c}_{1,2} \end{bmatrix} \in \mathbb{Z}_q$.
3. Compare w and $\lfloor \frac{q}{2} \rfloor$ treating them as integers in $\mathbb{Z}$. If they are close, i.e., if $\left| w - \lfloor \frac{q}{2} \rfloor \right| < \lfloor \frac{q}{4} \rfloor$, output 1, otherwise output 0.

KeyRev$(\mathrm{id}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$ On input an identity id, a time t, the revocation list RL, and the state ST, the algorithm adds $(\mathrm{id}, \mathrm{t})$ to RL for all nodes ν associated with identity id and returns RL.

4.5 Parameters, Correctness and Security

As in [3], the following error term is bounded by $[q\sigma m\alpha\, \omega(\sqrt{\log m}) + \mathcal{O}(\sigma m^{3/2})]$, that is

$$w = c_0 - \mathbf{e}_1^\top \begin{bmatrix} \mathbf{c}_{1,0} \\ \mathbf{c}_{1,1} \end{bmatrix} - \mathbf{e}_2^\top \begin{bmatrix} \mathbf{c}_{1,0} \\ \mathbf{c}_{1,2} \end{bmatrix} = \mathrm{m}\lfloor \frac{q}{2} \rfloor + \underbrace{x - \mathbf{e}_1^\top \begin{bmatrix} \mathbf{y} \\ \mathbf{z}_1 \end{bmatrix} - \mathbf{e}_2^\top \begin{bmatrix} \mathbf{y} \\ \mathbf{z}_2 \end{bmatrix}}_{\text{error term}}.$$

We can similarly set the parameters (q, m, σ, α) to ensure that the error term is less than $q/5$ and the system works:

$$m = 2n^{1+\delta}, \qquad q = m^2\sqrt{n} \cdot \omega(\log n),$$
$$\sigma = m \cdot \omega(\log n), \qquad \alpha = [m^2 \cdot \omega(\log n)]^{-1},$$

and round up m to the nearest larger integer and q to the nearest larger prime. We choose δ such that $n^\delta > \lceil \log q \rceil = \mathcal{O}(\log n)$.

We show that our RIBE construction is secure in the following theorem:

Theorem 4. *The RIBE system is* IND-sRID-CPA *secure provided that the* $(\mathbb{Z}_q, n, \bar{\Psi}_\alpha)$*-LWE assumption holds.*

We have given some intuition of our security proof earlier in Section 4.3. Due to space constraints, the detail is given in the full version of this paper [15].

5 Open Problem

We have proven our RIBE scheme to be selective-ID secure under the LWE assumption. However, we leave open the problem of how to construct an adaptive-ID secure RIBE scheme [22].

Acknowledgments. We thank the referees for helpful feedback. Research of the authors is supported in part by the National Research Foundation of Singapore under Research Grant NRF-CRP2-2007-03.

References

[1] Agrawal, S., Boneh, D., Boyen, X.: Efficient Lattice (H)IBE in the Standard Model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010)

[2] Agrawal, S., Boneh, D., Boyen, X.: Lattice Basis Delegation in Fixed Dimension and Shorter-Ciphertext Hierarchical IBE. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 98–115. Springer, Heidelberg (2010)

[3] Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (h)ibe in the standard model, `http://crypto.stanford.edu/~dabo/pubs/papers/latticebb.pdf`

[4] Agrawal, S., Boyen, X., Vaikuntanathan, V., Voulgaris, P., Wee, H.: Fuzzy identity based encryption from lattices. IACR Cryptology ePrint Archive, 2011/414 (2011)

[5] Aiello, W., Lodha, S.P., Ostrovsky, R.: Fast Digital Identity Revocation. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 137–152. Springer, Heidelberg (1998)

[6] Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: STOC, pp. 99–108 (1996)

[7] Ajtai, M.: Generating Hard Instances of the Short Basis Problem. In: Wiedermann, J., Van Emde Boas, P., Nielsen, M. (eds.) ICALP 1999. LNCS, vol. 1644, pp. 1–9. Springer, Heidelberg (1999)

[8] Ajtai, M., Dwork, C.: A public-key cryptosystem with worst-case/average-case equivalence. In: STOC, pp. 284–293 (1997)

[9] Alwen, J., Peikert, C.: Generating shorter bases for hard random lattices. In: STACS, pp. 75–86 (2009)

[10] Boldyreva, A., Goyal, V., Kumar, V.: Identity-based encryption with efficient revocation. In: ACM Conference on Computer and Communications Security, pp. 417–426 (2008)

[11] Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. SIAM J. Comput. 32(3), 586–615 (2003)

[12] Boneh, D., Boyen, X., Goh, E.-J.: Hierarchical Identity Based Encryption with Constant Size Ciphertext. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Heidelberg (2005)

[13] Boneh, D., Gentry, C., Hamburg, M.: Space-efficient identity based encryption without pairings. In: FOCS, pp. 647–657 (2007)

[14] Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai Trees, or How to Delegate a Lattice Basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer, Heidelberg (2010)
[15] Chen, J., Lim, H.W., Ling, S., Wang, H., Nguyen, T.T.K.: Revocable identity-based encryption from lattices. IACR Cryptology ePrint Archive, 2011/583 (2011)
[16] Cocks, C.: An identity based encryption scheme based on quadratic residues. In: IMA Int. Conf., pp. 360–363 (2001)
[17] Cramer, R., Damgård, I.: On the Amortized Complexity of Zero-Knowledge Protocols. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 177–191. Springer, Heidelberg (2009)
[18] Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. SIAM J. Comput. 38(1), 97–139 (2008)
[19] Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC, pp. 169–178 (2009)
[20] Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC, pp. 197–206 (2008)
[21] Libert, B., Vergnaud, D.: Towards Black-Box Accountable Authority IBE with Short Ciphertexts and Private Keys. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 235–255. Springer, Heidelberg (2009)
[22] Libert, B., Vergnaud, D.: Adaptive-ID Secure Revocable Identity-Based Encryption. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 1–15. Springer, Heidelberg (2009)
[23] Micciancio, D.: Generalized compact knapsacks, cyclic lattices, and efficient one-way functions from worst-case complexity assumptions. In: FOCS, pp. 356–365 (2002)
[24] Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. IACR Cryptology ePrint Archive, 2011/501 (2011)
[25] Naor, M., Nissim, K.: Certificate revocation and certificate update. IEEE Journal on Selected Areas in Communications 18(4), 561–570 (2000)
[26] Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: STOC, pp. 333–342 (2009)
[27] Regev, O.: New lattice-based cryptographic constructions. J. ACM 51(6), 899–942 (2004)
[28] Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC, pp. 84–93 (2005)
[29] Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
[30] Sepahi, R., Steinfeld, R., Pieprzyk, J.: Lattice-Based Completely Non-malleable PKE in the Standard Model (Poster). In: Parampalli, U., Hawkes, P. (eds.) ACISP 2011. LNCS, vol. 6812, pp. 407–411. Springer, Heidelberg (2011)
[31] Shamir, A.: Identity-Based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
[32] Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
[33] Waters, B.: Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009)

On Area, Time, and the Right Trade-Off

A. Poschmann[1] and M.J.B. Robshaw[2,⋆]

[1] Nanyang Technological University, Singapore
aposchmann@ntu.edu.sg
[2] Applied Cryptography Group, Orange Labs, France
matt.robshaw@orange-ftgroup.com

Abstract. Recently one of the most active fields of cryptography has been the design of lightweight algorithms. Often the explicit goal is to minimise the physical area for an implementation. While reducing area is an important consideration, beyond a certain threshold there is little point minimising area further. Indeed, it can be counter-productive and does not necessarily lead to the most appropriate solution. To provide a clear demonstration of this, we consider two lightweight algorithms that have been proposed for deployment on UHF RFID tags and which appear in a forthcoming ISO standard. Our results show that by choosing an implementation strategy that reduces but not necessarily minimises the area, very significant savings in time and substantial reductions to other physical demands on tag performance can be delivered. In particular, given the crucial importance of transaction time in the deployment of most contactless applications, our work illustrates that the most suitable practical implementation does not always conform to expectations.

1 Introduction

The challenging physical constraints posed by RFID tags have been a significant spur to cryptographic research. The world of RFID tag deployment is dominated by two operating frequencies; HF and UHF. Tags that communicate with an interrogator using HF (13.56 MHz) are well-established. Most familiarly we see these in public-transport applications and they will play a significant role in the proliferation of *Near Field Communication (NFC)* applications. Happily, the physical constraints for short-range HF-based tags are not onerous: sufficient power can be delivered to the tag to deploy standard cryptographic primitives like Triple-DES [21] and AES [22]. By contrast, the much cheaper and much smaller tags standardised by EPCglobal [4] communicate using UHF (860–960 MHz). As well as size and price, a significant advantage of UHF tags is that they can be read at a distance. For these tags, though, space and power consumption are at a premium.

The UHF RFID tag is a remarkable piece of engineering; not only are these passive devices small and cheap enough to be attached to millions of objects—for *track-and-trace* applications in the supply chain—but they must operate in

⋆ The author gratefully acknowledges the support of the Singapore National Research Foundation under Research Grant NRF-CRP2-2007-03.

multi-tag and multi-reader environments with exceptional reliability. This success is spurring their wide-spread deployment and, at the same time, leading calls for an extension of their functionality. One particular application of interest is that of product authentication and calls to use cheap UHF RFID tags as part of an anti-counterfeiting solution are well-known [1,15,17]. But the favored long-term solution to the problem, that of providing dynamic cryptographic tag authentication, requires lightweight cryptographic algorithms and standardised solutions would be a plus. This consideration helps provide a focus to the work in this paper. Our goal is to consider the issue of area-time trade-offs and as targets we consider two parts of the forthcoming ISO/IEC 29192 standard [14] dedicated to lightweight cryptography.

1.1 Area and Time: The Obvious Trade-Off?

Of course it is easy to say (and it is true) that there always exist trade-offs in implementation strategy. Indeed area and time offer the basis for the most well-known trade-offs in hardware implementation. Yet such a statement can sometimes mask considerable complexities. To those that rarely work at the hardware level, an area-time tradeoff often suggests that a factor t increase in area allows a t-fold parallelisation of algorithm components. This would yield a factor t reduction in processing time (also called *latency*) and a constant *area-time product* (AT-product). However when we consider the range of trade-offs available, the AT-product is rarely, if ever, constant since there are fixed-size components in any implementation. Further, for systems that result from an accumulation of optimisations, understanding the net gain for any given trade-off requires careful analysis.

The pioneering papers in the field of lightweight cryptography effectively defined the operating parameters for the field. With good justification, the primacy of area was singled out as the most influential parameter in an implementation. However many other factors effect the suitability of a solution in deployment and these factors are often set aside. For example HB$^+$ [16] was particularly lightweight in terms of on-the-tag operations. But it was observed in [5] that the scheme would require more than 80 000 bits of communication between reader and tag. Similarly, transaction time is rarely considered as a limiting factor in much of the academic work on RFID tags; yet for contactless applications this can be critical. Our goal in this paper, therefore, is to explore a range of implementation strategies for certain core technologies. By doing so we hope to highlight a more complete range of factors that determine the suitability of a solution for a particular application.

1.2 This Paper

In this paper we consider the value of implementation strategies that might differ from the "minimum-area" approach. While our focus will be on area and time, we will keep our eye on other factors. Maximum and average power consumption are important issues, particularly for UHF RFID tags that will be read at

Table 1. Area-time trade-offs for the block cipher PRESENT with 80-bit keys. The area-time product is proportional to the energy consumption per bit encrypted.

		Datapath width				
		64	32	16	8	4
scaleable	S-layer	379.61	200.97	111.65	66.99	44.66
	key `xor`	170.88	85.44	42.72	21.36	10.68
	sub-total	550.49	286.41	154.37	88.35	55.34
overhead	MUXes	0.00	149.12	74.56	37.28	18.64
	counter	0.00	5.00	10.00	15.00	20.00
	sub-total	0.00	154.12	84.56	52.28	38.64
fixed	flip-flops	864.00	864.00	864.00	864.00	864.00
	counter	54.00	54.00	54.00	54.00	54.00
	sub-total	918.00	918.00	918.00	918.00	918.00
	Area (GE)	1 468.49	1 358.53	1 156.93	1 058.63	1 011.98
	Time (clk)	31	65	129	258	516
	Area-Time product	45 523	88 304	149 244	273 127	522 182
	Factor increase	1.0	1.9	3.3	6.0	11.5

a distance. For the lightweight designs we consider in this paper, power consumption is dominated by static power consumption which is proportional to the physical area of the implementation. This suggests that we would always be particularly interested in minimum-area implementations. However an important consideration, particularly say for battery-powered sensor nodes, can be the energy consumption of an implementation. And since energy is given by the product of power and time, energy consumption will be effectively proportional to the area-time product.

To illustrate the issues in this paper, we have decided to concentrate on two algorithms that feature in ISO 29192, a forthcoming standard dedicated to lightweight cryptographic techniques. This multi-part document covers block ciphers (part two), stream ciphers (part three), and asymmetric techniques (part four) and we decided to consider PRESENT and CRYPTOGPS here. It is possible that the reader will find the paper somewhat unbalanced, with more space being attributed to the trade-off in CRYPTOGPS than to PRESENT. However this is a direct result of the more complex trade-off offered by CRYPTOGPS which takes some analytic and implementation effort to fully understand.

To compare the area of different implementations, it is typical to use the concept of the *gate equivalent (GE)*. The physical area of an implementation is divided by the size of a `nand` gate to give, what is intended to be, a broadly technology-neutral estimate of the size of an implementation. Since there can be significant variations in the area reported for different technologies, it is not perfect. But it nevertheless remains a reasonable measure, provided we avoid claiming too much if the difference between implementations is modest.

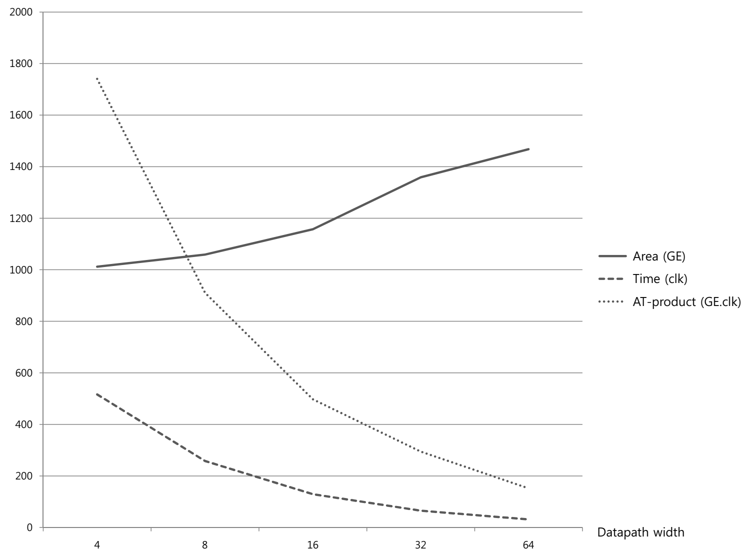


Fig. 1. Area (GE), time (cycles), and AT-product (scaled down by a multiplicative factor of 300 for convenience) for different implementations of 80-bit PRESENT

2 Area and Time for PRESENT

PRESENT is a compact block cipher with a classical SPN structure [3]. Designed to take either 80- or 128-bit keys, its simple regular structure means that the cipher offers a wide range of implementation options. The net results of some of these trade-offs are presented in Table 1. As can be seen from the table, the components in an implementation can be separated into three classes; components that are scaleable, components that are fixed-size, and any overheads that are required to deal with a specific implementation strategy. All the figures given in Table 1 were derived using *Synopsys DesignCompiler A-2007.12-SP1* to synthesize the VHDL code to the *Virtual Silicon* (VST) standard cell library *UMCL18G212T3* [29].

The results in Table 1 are worth considering in some detail. When considered solely in terms of area, there is not much to choose between the different implementations; the area lies between 1 000 and 1 500 GE, a variation of 50%. Yet, at the same time, the time to encrypt 64 bits varies by a factor of more than 16 and energy consumption per bit can vary by a factor of more than 11. This is illustrated in Fig. 1. It can therefore be hard to decide which approach would be most suitable for a given application. But it seems almost certain that the smallest implementation (of around 1 000 GE) will only be suitable on an exceptional basis. Instead it is likely that the implementation of around 1 500 GE will provide the best trade-off; in short the largest implementation.

Table 2. A headline comparison of the efficient implementation of the two block ciphers that feature in ISO 29192-2 and the AES

PRESENT (this paper)			CLEFIA ([28,2])		
Area (GE)	Time (clk)	AT-product	Area (GE)	Time (clk)	AT-product
1 468	31	45 523	6 050	18	108 900
1 359	65	88 304	5 490	36	197 640
1 157	129	149 244	2 678	176	471 328
1 059	258	273 127	2 594	192	498 048
1 012	516	522 182	2 488	328	816 064
			AES ([20,12])		
			Area (GE)	Time (clk)	AT-product
			3 100	160	496 000
			2 400	210	504 000

For the sake of completeness we provide some area-time tradeoffs for the implementation of CLEFIA [2,28] which also appears in ISO 29192-2 and AES which does not. These are 128-bit block ciphers that use a 128-bit key and it is, therefore, only to be expected that they will both require more space for an implementation. Given these different operational parameters it is quite difficult to make a meaningful comparison between PRESENT and CLEFIA or AES. For these last two ciphers the implementation results, by other authors, are presented in Table 2. Since different synthesis tools and technologies have been used, a close comparison between CLEFIA and AES cannot be made. However it is interesting that the trade-offs available to CLEFIA and AES appear to be broadly comparable and these figures suggest that CLEFIA and AES are likely to offer very similar implementation characteristics in terms of area, time, and energy efficiency.

3 The Trade-Off for CRYPTOGPS

CRYPTOGPS is a commitment-challenge-response scheme from ISO/IEC 29192 part 4. It is due to Girault, Poupard, and Stern and well-established in the literature [6,10,27]. Several variants have previously been standardised [13] and, over the years, several optimisations [9,11,13] have been proposed. Since our focus here is on the implementation rather than the specifics of the scheme, detailed descriptions can be found elsewhere. In short, the system-wide parameters include an elliptic-curve and an elliptic curve point P. Each tag contains a unique private key s, typically 160-bits long, while the associated public key $V = -sG$ is assumed to be available to the reader. Typical descriptions of CRYPTOGPS incorporate several optimisations. Among them is a storage/computation trade-off based around *coupons* which has been discussed in a variety of papers [7,11]. Coupons are not to everyone's tastes nor are they suitable for all use-cases. However limited-use tokens are familiar in a wide range of applications from pre-paid telephone cards to public transport ticketing and it might be argued that coupons are ideally suited to many RFID applications where tags might be read 10-20 times and then either discarded or recommissioned. Certainly the small additional cost in memory when storing coupons is more than offset by removing elliptic-curve operations from the tag.

The use of coupons is illustrated below where a hash function HASH [23] is used to reduce their size. Note the hash function is only needed during coupon generation and on the reader during tag verification; it is not implemented on the tag itself. In addition, optimisations outlined in ISO 9798-5 [13] suggest that the r_i be pseudo-random and re-generated on-the-fly at the time of use (instead of being stored). To do this we can use PRESENT in OFB mode with an 80-bit key k. We assume that the tag-specific keys s and k are fixed for the life of the tag while the IV used to begin OFB encryption will be variable (as required by any realistic CRYPTOGPS implementation). The form of the IV would likely contain some tag and/or coupon identifying information, but this is essentially an issue for the application architecture. The appropriate size of the coupons and challenges [7,10,11] will depend on the risk analysis and use-case.

Tag		**Reader**
COUPON PRE-COMPUTATION WITH PRNG		
For $1 \leq i \leq t$		
Let $r_i = \text{PRNG}_k(i)$		
where $\lvert r_i \rvert = \rho$		
Set $x_i = \text{HASH}(r_i P)$		
Store coupon x_i		
PROTOCOL USING ON-TAG PRNG		
At time i fetch x_i	$\xrightarrow{x_i}$	
	$\xleftarrow{c}$	Pick $c \in_R \{0,1\}^\delta$
Gen. $r_i = \text{PRNG}_k(i)$		
$y_i = r_i + (s \times c)$	$\xrightarrow{y_i}$	$\text{HASH}(y_i P + cV) \overset{?}{=} x_i$

For this paper our interest lies in a special trade-off that is available to CRYPTOGPS. The main tag computation consists of two components. One is to recompute r_i. The other is to compute $y_i = r_i + sc$. This simple computation involves integer addition and integer multiplication; there is no modular reduction and we will denote this version the `mult` variant. While avoiding a modular reduction is a major step towards UHF tag deployment, another optimisation has sought to overcome the cost of the multiplication itself. This is done by means of what is termed a *Low Hamming Weight challenge* [9]. Here the interrogator chooses a challenge that is longer than usual but which has a very low Hamming weight. Since there are few ones in the challenge and they can be judiciously spaced; the multiplication $(s \times c)$ on the tag is turned into a modest number of additions. This gives the potential to further optimise the on-tag computation and we will denote this version the `lhw` variant.

It is instructive to look at some sample parameter values, offering the same security, which highlight some of the essential differences between the `mult` and `lhw` variants. Even though we must increase the length of the challenge c in the `lhw` variant, to maintain the same security level for a comparable `mult` variant, the challenge is sparse and allows a variety of compact representations (see [19,26] for more details). In fact it turns out that the cost of transmitting the challenge for the `mult`-variant or the compressed challenge for the `lhw`-variant is the same, though there is a very minor penalty in decoding the compressed challenge on the tag. We summarise the basic trade-offs below for some sample parameter sets. The contrast between the two variants is clear. By avoiding an integer multiplication (in the `lhw` variant) we need a longer c, which means we need a longer r_i which takes longer to generate. This increases the time required to compute the response y_i as well as the time required to transmit it.

	`mult`-variant	`lhw`-variant
$\{\|s\|, \|k\|, \|c\|\}$	$\{160, 80, 36\}$	$\{160, 80, 1\,179\}$
communication of c (bits)	36	36
on-tag computation required	`add`, `mult`	`add`
length of r_i (bits)	276	1 419
communication of y_i (bits)	276	1 419

Note that this is a very different area-time trade-off than for Sect. 2. In the case of PRESENT the components of the computation remained unchanged; throughout we used the same S-boxes and the same diffusion layer but these were packaged in different ways. In the case of CRYPTOGPS however, the operation itself is changed from a multiplication to an addition and the trade-off is far more complicated. In previous implementation papers of CRYPTOGPS the single goal of minimising area meant that the `lhw`-variant was used. The range of implementations included FPGA [8] as well as synthesized [18,19] and fabricated [26] ASICs. The most useful comparison point is that reported in [26] which offers a full *fabricated* implementation using 0.25 μm technology. There the `lhw`-variant was implemented with PRESENT in OFB mode as a way of regenerating the r_i. Using two alternative implementations of PRESENT, serial and round-based, the results showed that the full cost of CRYPTOGPS would likely be bracketed by 2 876 GE and 724 cycles as the most time-efficient implementation and 2 403 GE but 9 319 clock cycles for the most space-efficient variant. The headline comparison between all this prior work is provided in Fig. 2.

3.1 Implementations of the `Mult`-Variant

Implementations of the `lhw`-variant showed that CRYPTOGPS was conceptually feasible on passive UHF RFID tags, particular if we focus on the key indicators of area and power consumption, see Fig. 2. However even then, the variant in Fig. 2 most likely to be preferred in practice is the one with the largest area requirement; yielding a response in 724 cycles with an area of 2 876 GE. A closer look at the implementation costs [26] is revealing, and motivates our exploration of the area-time trade-off for CRYPTOGPS.

	w/o PRNG		*with* PRNG			
	synthesized [18,19]		synthesized [26]		fabricated [26]	
variant	*area* (GE)	*time* (cycles)	*area* (GE)	*time* (cycles)	*area* (GE)	*time* (cycles)
1-bit	317	1 088	-	-	-	-
4-bit	-	-	2 143	9 319	2 403	9 319
8-bit	431	136	2 433	724	2 876	724
16-bit	900	68	-	-	-	-

Fig. 2. The performance profiles offered by different architectures, denoted *variant*, for existing work on the `lhw`-variant of CRYPTOGPS. Work on the left [18,19] is solely concerned with the computation of y_i, while work on the right [26] also incorporates the regeneration of r_i.

Total Implementation: 2 876 GE							
`PRESENT`		`Addwc`		`Controller`		`S_Storage`	
[GE]	%	[GE]	%	[GE]	%	[GE]	%
1 751	60.9	60	2.1	905	31.5	159	5.5

Fig. 3. The area breakdown for the CRYPTOGPS implementation of [26]

As we can see from Fig. 3 there are four main components to the implementation; PRESENT for the regeneration of r_i, the addition used in the CRYPTOGPS response computation (called `Addwc` in [26]) the controller, and some storage. We observe that the bulk of the implementation cost is due to PRESENT. This is not a problem with PRESENT itself, but rather an indication of the very low computation overhead incurred when supporting the `lhw`-variant of CRYPTOGPS. While the computation time of 724 cycles is reasonable, amounting to seven ms if the digital component is clocked at 100 KHz, we note from Sect. 3 that the size of the response y_i is quite large. The implications would depend on the application and use-case, but low-end communication rates provided in data sheets [24] suggest that returning y_i for the `lhw`-variant could take anywhere up to five times longer than its computation. This potential drawback could be avoided by moving to a version of CRYPTOGPS that uses multiplication. There will be a cost since the space required to support `mult` will certainly be several multiples of that required to support integer addition. However, since 61% of the `lhw` implementation is dominated by PRESENT (see Fig. 3) then the impact on the *overall* cost of implementation in moving to `mult` will not be significant. Indeed, we will show that the additional total area overhead should be only around 20%.

Implementing Multiplication. In this, and later sections, we will refer to specific operand sizes that correspond to previously proposed parameter sets. The computation at the heart of CRYPTOGPS is $y_i = r_i + sc$. In the `mult` variant, the computation of sc can be done in a variety of ways and the most important parameter is the word-size used for this multiplication. Do we perform a multiplication bit-by-bit or do we build the multiplication out of a series of

Table 3. Time (`clk`) to generate b bits when using n-by-n-bit operations within the multiplication. Area estimates (GE) are provided at the foot of the table.

bits generated (bits)	size of operand (bits) $n = 4$	$n = 8$	$n = 16$	$n = 32$	$n = 36$
$b = 16$	112	44	16	16	16
$b = 32$	256	116	52	32	32
$b = 48$	400	188	88	52	48
$b = 64$	544	260	124	68	64
area (GE)	820	850	950	1 020	1 040

n-by-n-bit operations? In addition to the impact on the area, our choice has timing and latency implications and an indication of the trade-offs is given in Table 3 where we give the area and the number of cycles to generate the least significant 16, 32, 48, and 64 bits of a 36- by 160-bit product respectively.

Combining with PRESENT. To find the most suitable implementation we need to recall that when we compute sc we are also required to compute r_i. Using PRESENT in OFB mode, *i.e.* following [26], means that after each encryption operation (32 clock cycles), 64 bits of r_i are available. It makes sense for the relevant 64 bits of sc to be available at the same time, so that they can be immediately added to the lower parts of the product as it is generated rather than having to store any intermediate values.

It turns out for our choices of n that `mult` requires more clock cycles than PRESENT. As a consequence we tried to reduce the timing requirements of `mult` while keeping the area requirements low. We chose a basic Shift-And-Add algorithm, where the least significant n bits of s are added to an intermediate result if the i^{th} bit of the challenge, bit c_i, is one; zero is added otherwise. Then the intermediate value is shifted by one bit to the right and the next bit of the challenge is used to determine whether the same chunk of s or zero is added. Once all bits of c are processed, the next n bits of s are used and the procedure repeats. In this way the first n output bits sc_i are available after i clock cycles. After that it takes 36 clock cycles for each consecutive n output bits until all 160 bits of s have been processed. At this time all 196 bits of sc are available.

The combination of a Shift-And-Add multiplier and round-based PRESENT has the advantage that parts of sc can be immediately added to r_i as they become available, while the time to compute the next part of sc (36 clocks) is also used to generate the next 64 bits of r_i (which requires only 32 clocks) in parallel.

As a side-issue, it is interesting to note that this discussion really helps to highlight the role and impact of latency in a design. While the time required to generate r_i should drop by a factor close to five—from 23 iterations of PRESENT to five iterations—the overall time to compute the response y_i will only drop by a factor of two. This is entirely due to the larger timing requirements carried by `mult` when compared to PRESENT.

Table 4. The area of the different components under two implementation strategies for PRESENT and multiplication

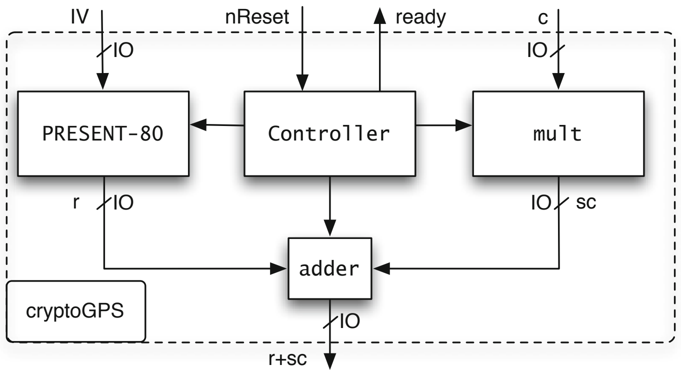
	PRESENT		MUL		Controller		Adder		Total
	[GE]	%	[GE]	%	[GE]	%	[GE]	%	[GE]
$n = 36$, $IO = 32$	1 689	54.8	1 034	33.5	185	6.0	175	5.7	3 083
$n = 4$, $IO = 4$	1 651	58.3	832	29.4	319	11.3	28	1.0	2 830

Implementation Results for $|c| = 36$. For our implementation of CRYPTOGPS we followed the work of [26]. We are not in a position to go through the fabrication process, but we can get good estimates on the performance of our implementations from synthesis. We used *Synopsys DesignCompiler A-2007.12-SP1* to synthesize our VHDL code to the *Virtual Silicon* (VST) standard cell library *UMCL18G212T3* [29], which is based on the *UMC L180 0.18µm 1P6M* logic process with a typical voltage of 1.8 V. We used *Synopsys Power Compiler* version *A-2007.12-SP1* to estimate the power consumption of our ASIC implementations. For synthesis and for power estimation we advised the compiler to keep the hierarchy and use a clock frequency of 100 KHz.

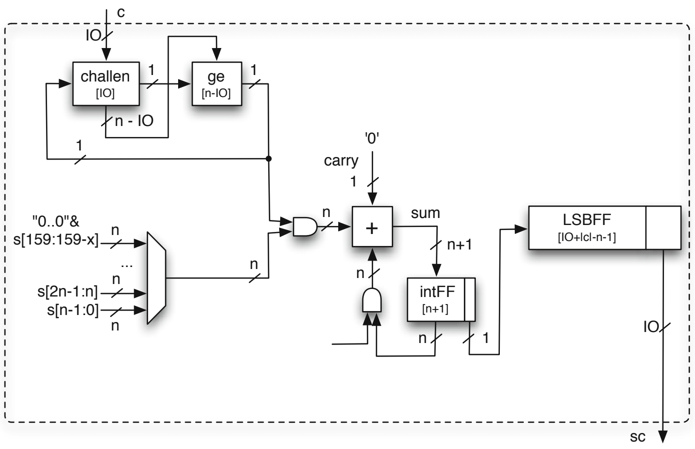
The input and output port size of CRYPTOGPS, MULT, PRESENT and the full-adder component is denoted by IO (see Fig. 4(a)). It is typically reasonable to choose $IO = n$, but for our first implementation we used an operand size of $n = 36$ (since this should give the fastest implementation). However 36 is not a common bus width and we used $IO = 32$.

The challenge c is variable and so we save c in a shift-register that can operate in two ways: 1) as a bit-serial shift-register that *rotates* the content by one bit to the right; 2) as an IO bit shift-register that *shifts* the content by IO bits to the right. When $|c|$ is not a multiple of IO the least significant bits are discarded. Since s can be fixed we chose to hardwire it, which gives a saving of 160 flip-flops. The appropriate part of s is chosen by a multiplexer, where we pad the last part of s with zeros when n does not divide $|s|$. The part in question is AND-ed with an n-bit replication of the least significant bit of the challenge register and this serves as one input to an n-bit full-adder. The sum (including the carry overhead) is stored in an $(n+1)$-bit register and the n most significant bits serve as the second input to the adder. The least significant bit is stored in a bit-serial shift-register of length $IO+|c|-n-1$ bits and the IO least significant bits serve as the output of this component which is ready every 32 clock cycles.

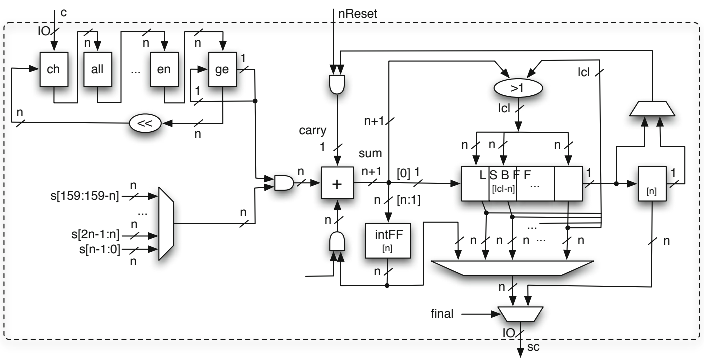
Recall that the 36 most significant bits of sc are ready at the same time, when all 160 bits of s have been processed. So to have a balanced design we chose to keep the output frequency and to save area at the cost of 29 additional clock cycles. In our implementation (depicted in Fig. 4(b)) y_i is available after 257 cycles. Power estimates for a frequency of 100 KHz, at a supply voltage of 1.8 V and using the smallest wire-load model (for circuits of about 10K GE) are 3.45μW. The total area requirement is 3 083 GE with a breakdown provided in Table 4. The overhead for PRESENT, when compared to the implementation in [3], is due to additional multiplexers that are required to a) interface between PRESENT and ADDER, and to b) feedback the ciphertext (OFB mode).



(a) Top-level hardware architecture of CRYPTOGPS.



(b) mult implementation with $N = 36$ and $IO = 32$.



(c) mult implementation with $N = 4$ and $IO = 4$.

Fig. 4. Hardware architectures of CRYPTOGPS and mult

Table 5. Comparison of time, area, and area-time (AT) product of `mult` and `lhw` variants [25,26]. All figures are extrapolated except synthesized results marked *.

Variant	$\|c\|$	(n, IO)	Time (clk)	Area (GE)	AT-product
`mult`	`36`	(4,4)	*1 533	*2 830	*4 338 811
		(8,8)	799	2 857	2 282 991
		(16,16)	432	2 980	1 287 338
		(36,32)	*257	*3 083	*792 311
	`64`	(4,4)	2 660	3 138	8 346 936
		(8,8)	1 362	3 177	4 327 496
		(16,16)	713	3 300	2 352 864
		(32,32)	389	3 424	1 331 917
`lhw`	1 179	(8,8)	880	2 556	2 249 280

Our second implementation aimed at minimal area while keeping an eye on time and we chose $n = IO = 4$ for the multiplication. We also used a different strategy to reduce the processing time. As soon as the multiplication has finished, the least significant bit shift register is stopped by a gated clock. The required part of sc is selected by a multiplexer which allows us to output one chunk every clock cycle without introducing additional latency. This saves 315 clock cycles at a cost of 63 GE for an additional multiplexer and a more complex control logic.[1] In this implementation (depicted in Fig. 4(c)) y_i is available after 1 533 cycles. The total area is 2 830 GE (Table 4) with 3μW an estimated power consumption.

Estimates for $|c| = 64$. It is interesting to consider a CRYPTOGPS variant with a 64-bit challenge. In this case r_i has a length of 304 bits, which still only requires five iterations of PRESENT, though these additional 28 bits (compared to the $|c| = 36$ variant) still impose a single figure clock cycle overhead. However the true expense lies in the multiplication as the time required for computing the product grows linearly with the challenge size $|c|$. The area increment is less severe, as the challenge size only impacts two register lengths. Hence the overhead consists of 56-bits of additional storage (299 to 336 GE). We have included timing and area estimates for this variant in Table 5.

4 Area and Time for CRYPTOGPS

While it will depend to some extent on the application, it seems that the `mult`-variant of CRYPTOGPS has been neglected in the cryptographic literature. While it is not straightforward, we can extrapolate the results in [26] to make a comparison. In that work the `lhw` challenge consisted of a 847-bit string that had a Hamming weight of five. These were the parameter choices specified in the original proposal by Girault and Lefranc [9]. Taking account of other considerations [7], we are interested to explore a different parameter set. However the net

[1] It is not obvious to what extent the 134 GE increment of the controller module is caused by this design decision and not by the additional counters required *etc.*

result of the changes to the figures given in [26] is surprisingly minor. A simple way of achieving a challenge space equivalent to 36 bits in the `lhw`-variant is to move to a slightly longer challenge of Hamming weight six. The power consumption will be effectively unchanged, as will the area for the implementation, though the computation time for y_i will increase from 724 cycles to an estimated 880 cycles[2] due to the longer generation time for r_i. The same implementation of CRYPTOGPS in [26] has been synthesized to the same standard-cell library used[3] in [25] allowing us to fairly compare area figures.

Using `mult` instead of `lhw` provides an interesting—but sophisticated—trade-off for CRYPTOGPS. Table 5 shows a comparison of our implementations using `mult` with the `lhw` variant reported in [26,25] with regards to time (in clock cycles), area (in GE), and the area-time (AT) product. All metrics are considered to be better the smaller they are. It is easy to see that the choice of $n = 4$ is not very useful, as it is 11% larger and 75% slower than `lhw`, resulting in an AT product that is nearly twice as bad. However, that changes with a growing operand size. An operand size of $n = 8$ will already yield a similar AT product as the `lhw` variant, and $n = 16$ will result in an AT product that is 43% smaller than the `lhw` variant. The real advantage of `mult` lies in larger operand sizes, such as $n = 32$, which yields an AT product that is only a third of the `lhw` variant, indicating energy savings in the same range and a much faster processing time.

5 Conclusion

In this paper we have highlighted the importance, and the very different results, that arise from different implementation strategies for lightweight cryptography.

As a first example we considered PRESENT where the different trade-off points are reasonably straightforward to establish. While many papers on security solutions for RFID would suggest that we take the minimum-area implementation of PRESENT, in reality it is hard to see an application where this would really be the preferred solution. Instead a more practical trade-off is obtained by taking the largest-area implementation of the ones we examined which takes one sixteenth the computation time and one eleventh the energy per bit of encryption. The moderate increase in area that is required will almost certainly be viewed as a reasonable cost.

We also considered the typical strategy of minimising area in implementations of CRYPTOGPS. This is achieved by replacing a conventional multiplication with a long sparse multiplication, the so-called `lhw`-variant. However we have shown in this paper that by using multiplication, and thereby accepting an increase in area, we appear to get a more reasonable trade-off. If we look purely at the relative costs of addition in `lhw` and multiplication in `mult`, then the area increases from 60 GE to over 1 000 GE, a factor of more than 16. The initial reaction

[2] To generate 1 419 bits requires 23 iterations of PRESENT instead of the 19 used in [26].

[3] Table 7.1 in [25] reports a higher cycle count since they also included the overhead from the *IO* handshake protocol; this has been excluded in [26] and here.

would be to choose the `lhw`-variant. However the true additional cost of supporting multiplication—when considered in relation to the *entire* cryptographic component—is only around 20%. So by accepting such a modest increase to a low-area implementation, the on-tag computation time can be reduced by more than 70%, the communication overhead can be reduced by 80%, and the total transaction time can be potentially reduced by more than 75%. These are significant performance savings, all at the same security level.

While we have deliberately focused on two algorithms that can be found in a forthcoming ISO standard, our work carries broader lessons for the use of cryptography in constrained devices. In particular, while area is a vital metric, the minimum-area implementation is not necessarily the most useful in practice.

References

1. Aigner, M., Burbridge, T., Ilic, A., Lyon, D., Soppera, A., Lehtonen, M.: RFID Tag Security, BRIDGE white paper, `http://www.bridge-project.eu`
2. Akishita, T., Hiwatari, H.: Very Compact Hardware Implementations of the Blockcipher CLEFIA. In: Proceedings of SAC 2010, pp. 2925–2928. IEEE (2008)
3. Bogdanov, A.A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M., Seurin, Y., Vikkelsoe, C.: PRESENT: An Ultra-Lightweight Block Cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007)
4. EPCglobal. EPC Radio-Frequency Identity Protocols, Class-1 Generation-2 UHF RFID, Protocol for Communications at 860-960 MHz, version 1.2.0 (October 23, 2008), `http://www.epcglobalinc.org`
5. Gilbert, H., Robshaw, M., Seurin, Y.: $HB^{\#}$, Increasing the Security and Efficiency of HB. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 361–378. Springer, Heidelberg (2008)
6. Girault, M.: Self-certified Public Keys. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 490–497. Springer, Heidelberg (1991)
7. Girault, M.: Low-Size Coupons for Low-Cost IC Cards. In: Domingo-Ferrer, J., Chan, D., Watson, A. (eds.) Proceedings of Smart Card Research and Advanced Applications, pp. 39–50. Kluwer Academic Press (2001)
8. Girault, M., Juniot, L., Robshaw, M.: The Feasibility of On-the-Tag Public Key Cryptography. In: RFIDsec 2007, Workshop Record (2007), `http://rfidsec07.etsit.uma.es/slides/papers/paper-32.pdf`
9. Girault, M., Lefranc, D.: Public Key Authentication with One (Online) Single Addition. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 413–427. Springer, Heidelberg (2004)
10. Girault, M., Poupard, G., Stern, J.: On the Fly Authentication and Signature Schemes Based on Groups of Unknown Order. Journal of Cryptology 19, 463–487 (2006)
11. Girault, M., Stern, J.: On the Length of Cryptographic Hash-Values Used in Identification Schemes. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 202–215. Springer, Heidelberg (1994)
12. Hämäläinen, P., Alho, T., Hännikäinen, M., Hämäläinen, T.D.: Design and Implementation of Low-Area and Low-Power AES Encryption Hardware Core. In: DSD, pp. 577–583 (2006)

13. ISO/IEC 9798: Information Technology – Security Techniques – Entity Authentication – Part 5: Mechanisms using Zero-Knowledge Techniques, http://www.iso.org
14. ISO/IEC 29192-4: Information Technology – Security Techniques – Lightweight Cryptography – Part 4: Public key techniques. Committee Draft
15. Jenkins, J., Mills, P., Maidment, R., Profit, M.: Pharma Traceability Business Case Report. BRIDGE white paper (May 2007), http://www.bridge-project.eu
16. Juels, A., Weis, S.A.: Authenticating Pervasive Devices with Human Protocols. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 293–308. Springer, Heidelberg (2005)
17. Lehtonen, M., Al-Kassab, J., Michahelles, F., Kasten, O.: Anti-counterfeiting Business Case Report. BRIDGE white paper (December 2007), http://www.bridge-project.eu
18. McLoone, M., Robshaw, M.J.B.: Public Key Cryptography and RFID Tags. In: Abe, M. (ed.) CT-RSA 2007. LNCS, vol. 4377, pp. 372–384. Springer, Heidelberg (2006)
19. McLoone, M., Robshaw, M.J.B.: New Architectures for Low-Cost Public Key Cryptography on RFID Tags. In: Proceedings of SecureComm 2005, pp. 1827–1830. IEEE Computer Society Press (2007)
20. Moradi, A., Poschmann, A., Ling, S., Paar, C., Wang, H.: Pushing the Limits: A Very Compact and a Threshold Implementation of AES. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 69–88. Springer, Heidelberg (2011)
21. National Institute of Standards and Technology. SP-800-67: Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher, Revision 1 (January 2012), http://csrc.nist.gov
22. National Institute of Standards and Technology. FIPS 197: Advanced Encryption Standard (November 2001), http://csrc.nist.gov
23. National Institute of Standards and Technology. FIPS 180-4: Secure Hash Standard (February 2011), http://csrc.nist.gov
24. NXP Semiconductors. UCODE EPC G2 Data Sheet, http://www.nxp.com
25. Poschmann, A.: Lightweight Cryptography - Cryptographic Engineering for a Pervasive World. Number 8 in IT Security. Europäischer Universitätsverlag, Published: Ph.D. Thesis, Ruhr University Bochum (2009)
26. Poschmann, A., Robshaw, M., Vater, F., Paar, C.: Lightweight Cryptography and RFID: Tackling the Hidden Overheads. In: Lee, D., Hong, S. (eds.) ICISC 2009. LNCS, vol. 5984, pp. 129–145. Springer, Heidelberg (2010)
27. Poupard, G., Stern, J.: Security Analysis of a Practical "On the Fly" Authentication and Signature Generation. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 422–436. Springer, Heidelberg (1998)
28. Sugawara, T., Homma, N., Aoki, T., Satoh, A.: High-performance ASIC implementations of the 128-bit block cipher CLEFIA. In: Proceedings of ISCAS 2008, pp. 2925–2928. IEEE (2008)
29. Virtual Silicon Inc. 0.18 μm VIP Standard Cell Library Tape Out Ready, Part Number: UMCL18G212T3, Process: UMC Logic 0.18 μm Generic II Technology: 0.18μm (July 2004)

Analysis of Xorrotation with Application to an HC-128 Variant*

Paul Stankovski, Martin Hell, and Thomas Johansson

Dept. of Electrical and Information Technology, Lund University,
P.O. Box 118, 221 00 Lund, Sweden

Abstract. Many cryptographic primitives rely on word rotations (R) and xor (X) to provide proper mixing. We give RX-system mixing a very general treatment and deduce some theoretical results on related probability distributions. Pure RX-systems are easy to break, so we show how to apply our theory to a more complex system that uses RX operations in combination with S-boxes. We construct an impractical (keystream complexity $2^{90.9}$), but new and non-trivial distinguisher for a variant of HC-128 for which modular addition is replaced with xor.

Keywords: RX, probability distribution, stream cipher, HC-128, cryptanalysis, distinguisher.

1 Introduction

Consider the Xorrotation family

$$g_{w,r_1,\ldots,r_n}(x) = x \oplus (x \lll r_1) \oplus \ldots \oplus (x \lll r_n)$$

of functions for which x is a w-bit word, $\oplus$ denotes xor and $\lll$ denotes left rotation (cyclic shift) with respect to the word length w. For the rotation amounts r_i we have $0 < r_i < w$. These bit mixing functions are often used in cryptographic primitives to provide intra-word diffusion.

While primitives that rely on modular addition (A), rotation (R) and xor (X) are commonly labeled ARX, $g_{w,r_1,\ldots,r_n}$ is RX. Pure AX- and RX-systems have been shown to be weak, see [3,8], but we will show how our theory can be used in practice by applying it to a more complex system that includes RX operations *and* S-boxes.

The Xorrotation function family was studied by Thomsen [11] and Rivest [9]. Thomsen showed that the mapping is invertible for all choices of distinct r_i with $0 \leq r_i < w$, and all word lengths $w = 2^k$ if n is odd. Very recently, Rivest gave a different and more general proof, a proof that in some sense reveals the true nature of the invertibility of the mapping. Many questions remain open, however. For example, some insight into the cases $w \neq 2^k$ and even n, separately and together, would be desirable.

* This is the short version of the paper.

While the main focus of Thomsen and Rivest was on invertibility, we are more interested in the probability distributions that $g_{w,r_1,\ldots,r_n}$ induce. That is, given an x chosen uniformly at random, what can we say about $g_{w,r_1,\ldots,r_n}(x)$ for different values of w and r_i, how much do the resulting probability distributions differ from the uniform one? To answer this question we will need some more information about the function than an assessment of its invertibility.

From a cryptanalytic perspective, a primitive that exposes a heavily biased probability distribution is prone to distinguishing attacks. This motivates the main goal of this paper, which is to show a few very general results on the probability distributions of Xorrotations. We will then show how to apply our findings toward a cryptanalysis of the eSTREAM final portfolio stream cipher HC-128 [12] designed by Wu.

There have been very few cryptanalytic results on HC-128. In Wu's design paper [12], a distinguisher was constructed based on analysis of the least significant bit of the 32-bit keystream words. The keystream complexity of his distinguisher was about $2^{160.5}$ 32-bit words according to the revised analysis of Stankovski et al. [10]. In that paper they constructed a new distinguisher with a keystream complexity of about $2^{152.5}$, which is the current best for the original HC-128. Other observations and attempts at HC-128 can be found in [2,4,5,6,7].

1.1 Contributions of This Paper

In this paper we present a general treatment of bit mixing using xor and word rotations. We deduce a few simple theoretical results on related probability distributions, results that can be used for cryptanalysis.

We further show how to apply the theory, together with some additional new observations, to produce a new analysis of a variant of the eSTREAM final portfolio stream cipher HC-128. While the keystream complexity of the resulting distinguisher is far from practical ($2^{90.9}$), these are the first results for the given HC-128 variant.

The paper is organized as follows. In Section 2 we first review distinguishers and hypothesis testing before showing some theoretical results on biased probability distributions and bit mixing using xor and word rotations. In Section 3 we apply the theory from the previous section by constructing a new distinguisher for the variant of HC-128 in which addition is replaced with xor. The paper is concluded in Section 4.

2 Biased and RX-Specific Probability Distributions

2.1 Reviewing Distinguishers and Relative Entropy

A distinguisher is a decision mechanism that takes n samples of keystream as input and outputs either "CIPHER" or "RANDOM". The decision mechanism uses the n input samples to perform a hypothesis test to determine which of the two known probability distributions that is more likely, that of the cipher, or the uniform one. The Neyman-Pearson Lemma (see [1]) provides the optimal hypothesis test, which is translated into Definition 1 for independent samples.

For a probability mass function P, we use square bracket notation $P[x]$ here to denote the probability of event x.

Definition 1 (Relative Entropy). *The relative entropy between two probability mass functions P_0 and P_1 over the same domain $\mathcal{X}$ is defined as*

$$D\left(P_0 \| P_1\right) = \sum_{x \in \mathcal{X}} P_0[x] \log \frac{P_0[x]}{P_1[x]}. \tag{1}$$

Relative entropy has a couple of aliases in literature; information divergence and Kullback-Leibler distance. In our paper we will sometimes say 'the divergence of P' meaning $D(P\|U)$, where U denotes the corresponding uniform distribution.

The error probabilities for the corresponding hypothesis test reach the point at which they start to decrease exponentially when the number of samples n used in the hypothesis test approaches

$$n \approx \frac{1}{D(P_0\|P_1)}. \tag{2}$$

In this paper, (2) will be used as a measure of the number of samples needed for our distinguisher. This is the same measure that was used in [10].

The time- and keystream complexities of the distinguisher depend on how the observed keystream is used to assemble the samples. In our application, as we will see in Section 3, this assembly is very fast, requiring only four xor operations to build one sample.

2.2 The Divergence of Probabilistically Biased Distributions

We will be using probabilistic equalities in conjunction with the divergence measure D, so we need to determine how the former influence the latter.

Definition 2 (Probabilistically biased distribution). *Let A be any distribution of size 2^w, and let U be the uniform distribution of the same size. A distribution resulting from sampling A with probability p and U with probability $1-p$ is said to be probabilistically biased with parameters (p, A), or (p, A)-biased.*

Theorem 1 (Probabilistic divergence). *The divergence of a (p, A)-biased distribution is $p^2 D(A\|U)$.*

Theorem 1 is proven in the full version of this paper.

2.3 RX-Induced Probability Distributions

Consider the function

$$f_{w,r}(x) = x \oplus (x \lll r),$$

where x is a w-bit variable and $\lll$ denotes left rotation with respect to the word length w. For all rotation amounts r in this section we enforce the constraint $0 < r < w$. This construction is often used as a basic mixing component in cryptographic primitives.We take a probability distribution approach here to provide results that are practical for cryptanalysis.

For a distribution to be of use to an analyst, it needs to boast a high divergence. This makes it easily distinguishable from a uniform distribution. In this context, all divergences of magnitude 1 and above are extremely high.

In the following we assume w-bit words, and we number the bit positions from least- to most significant bit 0 through $w-1$. Also, for all rotation amounts r we assume that $0 < r < w$.

Definition 3 (Probability distribution operator E). *A mapping $f : U \longrightarrow V$ is said to* generate *a probability distribution on V (uniformly) in the following way. Starting with an empty array of size $|V|$, let each $x \in U$ contribute probability $2^{-|U|}$ to slot $f(x)$. Summation over all possible domain values produces the probability distribution in question. The probability distribution generated by f is denoted $E(f)$.*

Thus, $f_{w,r}(x)$ and $f_{w,r_1}(x_1) \oplus \ldots \oplus f_{w,r_n}(x_n) = f_{w,r_1,\ldots,r_n}(x_1,\ldots,x_n)$ generate probability distributions $E(f_{w,r})$ and $E(f_{w,r_1,\ldots,r_n})$, respectively.

Definition 4 (r-orbit). *In a w-bit word, the bit positions reachable from bit position i as we apply r-bit rotation again and again – the* orbit *of bit position i under r-bit rotation – is given by the bit set*

$$\{(i + kr) \bmod w \mid k \in \mathbf{N}\},$$

and there are $\gcd(w,r)$ distinct orbits, each of length $\frac{w}{\gcd(w,r)}$.

Proposition 1 (Divergence of $E(f_{w,r})$). *The divergence of $E(f_{w,r})$ is $\gcd(w,r)$.*

Proof. For every given w-bit output value $y = y_{w-1} \ldots y_0$, the equation system

$$f_{w,r}(x) = y$$

has $2^{\gcd(w,r)}$ solutions. That is, restricting the domain and range of $f_{w,r}$ to only one r-orbit, that corresponding equation system has precisely two solutions, and the restricted mapping is consequently 2-to-1. There are $\gcd(w,r)$ disjoint r-orbits, so the entire mapping $f_{w,r}$ is $2^{\gcd(w,r)}$-to-1.

From this it follows that the probability distribution $E(f_{w,r})$ has precisely $2^{w-\gcd(w,r)}$ non-zero probability entries, each being equal to $2^{\gcd(w,r)-w}$ since x is uniformly distributed over the domain. Using (1) we get

$$\begin{aligned} D(E(f_{w,r})\|U) &= 2^{w-\gcd(w,r)} \left(2^{\gcd(w,r)-w} \log_2 \frac{2^{\gcd(w,r)-w}}{2^{-w}} \right) \\ &= \gcd(w,r). \end{aligned}$$

□

We state a generalized version as Theorem 2, the proof of which can be found in the full version of this paper.

Theorem 2 (Divergence of $E(f_{w,r_1,\ldots,r_n})$). *The divergence of $E(f_{w,r_1,\ldots,r_n})$ is* $\gcd(w, r_1, r_2, \ldots, r_n)$*.*

The probability distribution $E(f_{w,r_1,\cdots,r_n})$ is precisely what we will need for our cryptanalysis of the HC-128 variant, so we will be content with these findings. A deepened RX analysis along the lines of Thomsen [11] and Rivest [9] with further results on $E(g_{w,r_1,\ldots,r_n})$ is available in the full version of this paper.

3 Application to HC-128

We now illustrate how Theorems 1 and 2 can be applied in a beautiful way to produce a new distinguisher for a partly linearized version of the stream cipher HC-128. In particular, we show that HC-128 becomes weak if its $+$ operators are replaced with $\oplus$.

Despite the removal of the non-linearity provided by modular addition, it is still *not* easy to construct low-complexity distinguishers for this variant of HC-128. This is because we still have to deal with the S-boxes.

3.1 Notation and Review of HC-128

In this section we give a very brief description of the original HC-128 keystream generation process. HC-128 is defined in [12], from which we adopt and adapt the notation. HC-128 specifies both a key and initialization vector size of 128 bits. Up to 2^{64} bits of keystream can be generated with each key/IV pair. Letting x and y be 32-bit integers, we have binary operators $+, \boxminus, \oplus, ||, \ggg$ and $\lll$ that denote 32-bit addition, subtraction modulo 512, xor, concatenation, and right and left rotation, respectively. The internal state of HC-128 consists of two tables denoted P and Q. Each table contains 512 words of 32 bits each. The keystream is denoted by s and the 32-bit keystream word generated at the i^{th} step is denoted s_i; $s = s_0||s_1||s_2|| \ldots$.

Keystream generation proceeds as follows. One table entry is updated and one 32-bit keystream word is generated at each step. One full update of an entire table P or Q takes place during a *session* consisting of 512 consecutive steps. First, table P is updated and table Q is used to provide update values. The roles of tables P and Q are reversed every session.

We will find it convenient to express table entries $P[i]$ as P_i, $P[i \boxminus j]$ as P_{i-j}, and we write P^k_{i-j} for $(P_{i-j} \ggg k)$.

Our analysis is independent of the initialization, so we leave that part out of this description referring to [12].

Probabilistic equalities, equalities that are true with some given minimum probability, are indicated by annotating the equality sign with the corresponding probability.

3.2 A New HC-128 Variant Distinguisher

Table updates in the original HC-128 are performed according to

$$P_i = P_i + (P_{i-3}^{10} \oplus P_{i-511}^{23}) + P_{i-10}^{8}.$$

During the first half session we have 256 table updates with keystream generation

$$s_i = (Q_a + Q_b) \oplus P_i, \tag{3}$$

where $0 \leq i, a \leq 255$ and $256 \leq b \leq 511$. Similarly, the second half session provides 256 table updates with keystream generation

$$s_j = (Q_c + Q_d) \oplus P_j, \tag{4}$$

where $0 \leq c \leq 255$ and $256 \leq j, d \leq 511$. This completes one full update of table P. The subsequent session updates table Q, and for the three first updates we have

$$\begin{aligned} s_k &= (P_l + P_m) \oplus Q_k \\ &= (P_l + P_m) \oplus (Q_e + (Q_f^{10} \oplus Q_g^{23}) + Q_h^{8}), \end{aligned} \tag{5}$$

with $0 \leq l, e, g \leq 255$ and $256 \leq m, f, h \leq 511$. The P's and Q's in Eq's. (3), (4) and (5) denote lookups into the same tables.

Now consider the HC-128 variant for which all $+$ operators are replaced by $\oplus$. Choosing any one equation triplet (3)(4)(5), we have $i = l$ and $j = m$ (and thus $P_i = P_l$ and $P_j = P_m$) each with probability 2^{-8}. We also have $a = e, c = g$ or $a = g, c = e$ with combined probability 2^{-15} (assume $a = e, c = g$ without loss of generality). We similarly have $b = f, d = h$ or $b = h, d = f$ with combined probability 2^{-15} (assume $b = h, d = f$). Using (3) and (4) linearly together with all the equations (5) gives us 3×256 equation triplets for every 512 keystream words. With probability $\frac{3 \times 256}{512} \times 2^{-46} > 2^{-45.42}$ we therefore have

$$s_i \oplus s_j \oplus s_k \overset{2^{-45.42}}{=} \underbrace{(Q_b \oplus Q_b^{8})}_{N_1} \oplus \underbrace{(Q_c \oplus Q_c^{23})}_{N_2} \oplus \underbrace{(Q_d \oplus Q_d^{10})}_{N_3}, \tag{6}$$

where the left-hand side consists of known keystream words only, and N_1, N_2 and N_3 are observations from $E(f_{32,8})$, $E(f_{32,23})$ and $E(f_{32,10})$, respectively. Their combined distribution $E(f_{32,8,23,10})$ has divergence $\gcd(32, 8, 23, 10) = 1$ according to Theorem 2. Eq. (6) shows that we have a $(2^{-45.42}, E(f_{32,8,23,10}))$-biased distribution, which according to Theorem 1 results in a divergence of about $2^{-90.9} \times \gcd(32, 8, 23, 10) = 2^{-90.9}$. This yields a distinguisher requiring roughly $2^{90.9}$ 32-bit keystream words, so it is clear that the $+$ operator plays a vital role in HC-128.

If we use evaluation of the left-hand side of Eq. (6) over all three k-values – four xor operations on 32-bit keystream words – as time unit, we obtain a time complexity of $2^{89.9}$. In absolute terms, this measure is *much* cheaper, a factor of at least 2^{10}, than the cost of an initialization. For comparison, if we were to consider initializations instead, the time complexity of our distinguisher would be less than 2^{80}.

4 Conclusions

We have presented some new and general results on probability distributions related to RX-systems. We have also shown how to apply the new theory to a non-trivial system that uses RX operations in combination with S-boxes. We did this by building a new distinguisher for a partly linearized variant of HC-128. The total time complexity of the new distinguisher is $2^{89.9}$ very simple operations (xor and comparison of 32-bit keystream words) and the distinguisher requires about $2^{90.9}$ 32-bit keystream words.

Acknowledgements. This work was sponsored in part by the Swedish Research Council (Vetenskapsrådet) under grant 621-2006-5249. We also wish to thank Martin Ågren for his helpful comments and discussions.

References

1. Cover, T., Thomas, J.A.: Elements of Information Theory. Wiley series in Telecommunication. Wiley (1991)
2. Dunkelman, O.: Phorum5: ECRYPT forum, post 'A small observation on HC-128', `http://www.ecrypt.eu.org/stream/phorum/read.php?1,1143` (last accessed on January 14, 2011)
3. Khovratovich, D., Nikolić, I.: Rotational Cryptanalysis of ARX. In: Hong, S., Iwata, T. (eds.) FSE 2010. LNCS, vol. 6147, pp. 333–346. Springer, Heidelberg (2010), `http://dx.doi.org/10.1007/978-3-642-13858-4_19`
4. Kircanski, A., Youssef, A.M.: Differential Fault Analysis of HC-128. In: Bernstein, D.J., Lange, T. (eds.) AFRICACRYPT 2010. LNCS, vol. 6055, pp. 261–278. Springer, Heidelberg (2010)
5. Liu, Y., Qin, T.: The key and IV setup of the stream ciphers HC-256 and HC-128. In: International Conference on Networks Security, Wireless Communications and Trusted Computing, pp. 430–433 (2009)
6. Maitra, S., Paul, G., Raizada, S., Sen, S., Sengupta, R.: Some observations on HC-128. In: Designs, Codes and Cryptography, pp. 1–15 (2010)
7. Paul, G., Maitra, S., Raizada, S.: A Combinatorial Analysis of HC-128. Cryptology ePrint Archive: Report 2010/387
8. Paul, S., Preneel, B.: Solving Systems of Differential Equations of Addition (Extended Abstract). In: Boyd, C., González Nieto, J.M. (eds.) ACISP 2005. LNCS, vol. 3574, pp. 75–88. Springer, Heidelberg (2005), `http://dx.doi.org/10.1007/11506157_7`
9. Rivest, R.L.: The invertibility of the XOR of rotations of a binary word. International Journal of Computer Mathematics 88(2), 281–284 (2011); First published on: December 4, 2010
10. Stankovski, P., Ruj, S., Hell, M., Johansson, T.: Improved Distinguishers for HC-128. In: Designs, Codes and Cryptography, pp. 1–16, `http://dx.doi.org/10.1007/s10623-011-9550-9`
11. Thomsen, S.S.: Cryptographic hash functions. PhD thesis, Technical University of Denmark (November 2008)
12. Wu, H.: The Stream Cipher HC-128. In: Robshaw, M., Billet, O. (eds.) New Stream Cipher Designs. LNCS, vol. 4986, pp. 39–47. Springer, Heidelberg (2008)

Private Fingerprint Matching

Siamak F. Shahandashti[1], Reihaneh Safavi-Naini[2], and Philip Ogunbona[3]

[1] University of Paris 8 and École Normale Supérieure, Paris, France
fshahand@di.ens.fr
[2] Institute for Security, Privacy, and Info. Assurance,
University of Calgary, Canada
rei@ucalgary.ca
[3] ICT Research Institute, University of Wollongong, Australia
philipo@uow.edu.au

Abstract. We propose a fully private fingerprint matching protocol that compares two fingerprints based on the most widely-used minutia-based fingerprint matching algorithm. The protocol enables two parties, each holding a private fingerprint, to find out if their fingerprints belong to the same individual. Unlike previous works, we do not make any simplifying assumption on the matching algorithm or use generic multiparty computation protocols in our constructions. We employ a commonly-used algorithm that works by first comparing minutia pairs from the two fingerprints based on their types, locations, and orientations, and then checking if the number of matching minutia pairs is more than a threshold, and we propose a concrete, scalable, and modular protocol. We prove security against honest-but-curious adversaries and discuss how security against malicious adversaries can be achieved using standard cryptographic techniques. Our protocol is realized using common cryptographic primitives and do not require pairing- or lattice-based cryptography.

1 Introduction

Fingerprints authentication systems are increasingly used as means of verifying the identities of individuals in everyday life. This trend calls for the invention and deployment of more efficient privacy enhancing technologies to protect fingerprint privacy. Fundamental to any fingerprint-based system is the fingerprint matching algorithm which decides whether or not two given fingerprints belong to the same individual. Often the two fingerprints in question are held by two separate entities that are not willing to share unnecessary information with each other. Hence, protocols are required that enable the two parties decide whether or not their private fingerprints match without revealing any further information about them. We call such a protocol a *private fingerprint matching* protocol.

A fingerprint is usually represented by a tuple of *minutiae*, where each minutia is in the form of a vector of elements representing e.g. the type, location, and orientation of the minutia. The fingerprint matching algorithms run in two general steps. First minutia pairs from the two fingerprints are compared based on e.g. the Euclidean distance of their locations and the angular difference of their orientations, and matching pairs are identified. Then a decision is made on whether the fingerprints match based on the outcomes of minutia pair matchings, e.g. based on the number and certain patterns of matching minutiae.

Applications of private fingerprint matching protocols in fingerprint-based authentication systems range from the more classic match-on-server applications to the newer match-on-card applications. The former category of applications employ private fingerprint matching as a building block and include e.g. biometric authentication systems using extended private information retrieval [3,4]. The latter category of applications employ private fingerprint matching protocols directly and include e.g. the proposal of [1] in which a protocol similar to private fingerprint matching is used. Further applications of private fingerprint matching protocols are conceivable in any situation where the two fingerprints being matched are held by two separate privacy-conscious parties.

Protocols for private fingerprint matching proposed in the literature fall short of providing full privacy without the presence of a trusted third party. In this paper, we propose a private fingerprint matching protocol which guarantees that *no* information other than the outcome of the matching is revealed to the participating entities.

1.1 Our Contributions

We propose a fully private fingerprint matching protocol that compares two fingerprints based on the most widely-used minutia-based fingerprint matching algorithm. The protocol enables Alice to compare a (private) fingerprint that she holds, with another (private) fingerprint that Bob holds, such that at the end of the protocol, Alice only learns if her fingerprint 'matches' Bob's or not, and Bob does not learn anything.

Unlike (all but one) previous works, we do not make any simplifying assumption on the matching algorithm and base our protocol on a commonly used algorithm that compares minutia pairs of the two fingerprints based on their types, locations, and orientations. A pair of minutiae are defined to match if they have the same type (i.e. they are both ridge endings, bifurcations, etc.), their locations are within a threshold Euclidean distance, and their orientations are within a threshold angular difference. Two fingerprints 'match' if the number of their matching minutia pairs is more than a threshold. The only previous work that is based on the same fingerprint matching protocol uses generic multiparty computation as a building block. Our protocol is concrete and does not rely on any generic building blocks.

One of the main building blocks of our constructions is a primitive that we call *aided computation*. Consider a polynomial $P(x) = \sum a_k x^k$ and a homomorphic encryption algorithm E for which Alice knows the decryption key. It is known that given $\{\mathsf{E}(a_k)\}_{\forall k}$ and x, the homomorphic property of E enables Bob to compute $\mathsf{E}(P(x))$. Aided computation, on the other hand, enables Bob to compute $\mathsf{E}(P(x))$ given $\{a_k\}_{\forall k}$ and $\mathsf{E}(x)$. The underlying idea, which has appeared before in the cryptographic literature, is to simply get Alice to *obliviously* help with the calculation. This can be done in situations like ours where Alice is available for interaction and in effect enables the kinds of computations that only a fully-homomorphic encryption permits, to be performed using only a homomorphic encryption.

A simplified description of our proposed private fingerprint matching protocol is as follows. Let $\{p_i\}_{i=1}^{n}$ and $\{p'_j\}_{j=1}^{m}$ denote Alice and Bob's fingerprint minutiae, respectively. We first use the properties of homomorphic encryption schemes to privately calculate and compare the Euclidean distance and angular difference for each pair of minutiae (p_i, p'_j) to given thresholds. Our protocol enables Bob to calculate $\mathsf{E}(z_{ij})$ for each pair (p_i, p'_j), such that z_{ij} is zero if the two minutiae match and non-zero otherwise. Then, using the aided computation idea, Bob calculates $\mathsf{E}(R(z_{ij}))$, where R is a polynomial that maps zero to one and non-zero values to zero. Let $\sigma = \sum R(z_{ij})$. Now, using the homomorphic property of the encryption scheme $\mathsf{E}(\sigma)$ can be calculated by Bob. Note that σ equals the total number of minutia matchings. Finally, the homomorphic property can be used to finalize the protocol and let Alice find out if σ is greater than or equal to a threshold τ or not.

We provide standard two-party computation security definitions for our protocol and prove its security against honest-but-curious adversaries. Furthermore, we discuss how security against malicious adversaries can be achieved using standard cryptographic techniques. Our protocol is constructed using common cryptographic primitives, such as homomorphic encryption schemes, and do not require pairing- or lattice-based cryptography that imply complex structures and high computational complexity. As an example application, we show how our private fingerprint matching protocol can be integrated into a previously proposed fingerprint-based remote authentication system to enhance the level of privacy it provides for the entities in the system.

1.2 Related Work

There has been proposals in the literature for fingerprint matching protocols in which the two fingerprints are held by two entities. The closest to our work are [1,6,2]. For a comprehensive discussion of related works please refer to the full version of this paper [9].

2 Preliminaries

In the following we discuss the notation and preliminary definitions that we use throughout the paper.

2.1 Notation

We use the notation $(O_{\mathsf{A}}, O_{\mathsf{B}}) \leftarrow \mathsf{P}[\mathsf{Alice}(I_{\mathsf{A}}) \leftrightarrow \mathsf{Bob}(I_{\mathsf{B}})](I_{\mathrm{p}})$ to denote that a protocol P between a party Alice with private input I_{A} and a party Bob with private input I_{B} is run with public protocol input (i.e. input to both parties) I_{p} and at the end of the protocol Alice's output is O_{A} and Bob's output is O_{B}. If a party has no input or output we use the placeholder $-$. We use $\mathsf{E}(\cdot)$ and $\mathsf{D}(\cdot)$ short for $\mathsf{E}_{pk}(\cdot)$ and $\mathsf{D}_{sk}(\cdot)$ to denote encryption and decryption using public key pk and secret key sk, respectively.

2.2 Homomorphic Encryption Schemes

An encryption scheme defined on a plaintext field with operations $(+, \cdot)$, with encryption and decryption schemes (E, D), is homomorphic if there exists a public operation $\oplus$ such that for any plaintexts a and b: $\mathsf{E}(a + b) = \mathsf{E}(a) \oplus \mathsf{E}(b)$. This automatically implies that there exists a second public operation $\odot$ such that for any plaintext a and any scalar c: $\mathsf{E}(c \cdot a) = c \odot \mathsf{E}(a)$. We consider schemes that are semantically secure [5], e.g. . Paillier's encryption scheme [8]. We assume that the resulting ciphertexts are rerandomized when $\odot$ and $\oplus$ operations are carried out.

2.3 Fingerprint Minutiae and Fingerprint Matching

The most common method for fingerprint matching is through extraction of minutiae, comparing them based on their types, locations, and orientations, and deciding based on the number of minutia matchings [7]. A fingerprint minutia is in the form (t, x, y, θ), where t, (x, y), and θ determine the type, location, and orientation of the minutia. Minutia type represents the fact that the minutia is e.g. either a ridge ending, bifurcation point, or of other types. We also assume that the size and orientation of each fingerprint is adjusted after acquisition. To this end, first the orientation needs to be adjusted using widely-used *pre-alignment* techniques (see [7] and the references within.). Then the resulting fingerprint image can be resampled/resized so that the resolution/size of the image conforms to a fixed value.

Two minutiae are considered to match if their types are the same, their locations are closer than a threshold Euclidean distance d_{E}, and their orientations are within an angular difference d_{a} of each other. Let us define the following distance functions for two minutiae $p = (t, x, y, \theta)$ and $p' = (t', x', y', \theta')$: $\mathrm{dis}_{\mathrm{E}}((x, y), (x', y')) = \sqrt{(x - x')^2 + (y - y')^2}$ and $\mathrm{dis}_{\mathrm{a}}(\theta, \theta') = \min(|\theta - \theta'|, 2\pi - |\theta - \theta'|)$. The two minutiae are defined to match if $t = t'$, $\mathrm{dis}_{\mathrm{E}}((x, y), (x', y')) \leq d_{\mathrm{E}}$, and $\mathrm{dis}_{\mathrm{a}}(\theta, \theta') \leq d_{\mathrm{a}}$.

A fingerprint matching can be carried out by checking if the two sets of extracted minutiae have at least a threshold number of matching minutiae in common. In this paper we propose a protocol to carry out the above fingerprint matching *privately*.

2.4 Aided Computation

Assume Alice is in possession of a decryption key for a homomorphic encryption scheme. The basic idea in aided computation is for Bob to employ the ability to interact with Alice to enable him to use the homomorphic encryption scheme as

$(-, \mathsf{E}(P(\beta))) \leftarrow \mathsf{AidComp}[\mathsf{Alice}(sk) \leftrightarrow \mathsf{Bob}(\mathsf{E}(\beta), P)](pk, \eta)$

1. Bob randomizes its input by computing $\mathsf{E}(r\beta) = r \odot \mathsf{E}(\beta)$ for a random r, and sends the result to Alice.
2. Alice decrypts the received ciphertext to get $r\beta$, calculates $\{(r\beta)^k\}_{k=2}^{\eta}$, encrypts the results to ciphertexts $\{\mathsf{E}((r\beta)^k)\}_{k=2}^{\eta}$, and sends them to Bob.
3. Bob derandomizes the results by computing for all $k = 2$ to η: $\mathsf{E}(\beta^k) = r^{-k} \odot \mathsf{E}((r\beta)^k)$ and then computes $\mathsf{E}(P(\beta)) = \bigoplus_{k=0}^{\eta} (a_k \odot \mathsf{E}(\beta^k))$.

Fig. 1. The aided computation protocol

a fully homomorphic one. That is, Alice aids Bob in carrying out operations in the encrypted domain that are not supported by the homomorphic encryption but are possible with the knowledge of the secret key. The aid by Alice must be performed *obliviously* to ensure privacy of the plaintexts corresponding to the involved ciphertexts.

In our protocol, given $\mathsf{E}(\beta)$ Bob often needs to calculate $\mathsf{E}(P(\beta))$ for a polynomial $P(x) = \sum_{k=0}^{\eta} a_k x^k$. Hence we formalize an aided computation protocol $\mathsf{AidComp}$ in which Alice has private input sk, Bob has private input $(\mathsf{E}(\beta), P)$, the public input is a pair (pk, η), and Bob needs to calculate $\mathsf{E}(P(\beta))$. The protocol we propose is based on the idea mentioned above and depicted in Figure 1. We will use this protocol as a building block for our constructions.

An important note is that the protocol requires non-zero β so that $\mathsf{E}(\beta)$ and $\mathsf{E}(r\beta)$ do not leak information about β. Let us call this *the non-zero requirement.* We make sure this condition is met in our constructions.

3 The Private Fingerprint Matching Protocol

We propose the following protocol between Alice and Bob for private fingerprint matching. Let Alice have as private input a set of minutiae $F = \{p_1, \ldots, p_n\}$ where for each $i = 1$ to n: $p_i = (t_i, x_i, y_i, \theta_i)$ and Bob have as private input a set of minutiae $F' = \{p'_1, \ldots, p'_m\}$ where for each $j = 1$ to m: $p'_j = (t'_j, x'_j, y'_j, \theta'_j)$. Alice's private output of the protocol is the binary predicate whether or not there are at least τ number of her minutiae matching those of Bob's, where two minutiae are defined to match if their types are the same, their locations are closer than a threshold Euclidean distance d_{E}, and their orientations are within an angular difference d_{a} of each other.

Let us denote the set of all possible minutia types by T. Consider the set of minutiae $F = \{p_1, \ldots, p_n\}$ where for each $i = 1$ to n: $p_i = (t_i, x_i, y_i, \theta_i)$. For each p_i, let the polynomial Q_i be defined and calculated via the Lagrange interpolation such that we have the following:

$$Q_i(x) = \sum_{k=0}^{|T|-1} b_{ik} x^k = \begin{cases} 0 & \text{if } x = t_i \\ 1 & \text{if } x \in T \setminus \{t_i\} \end{cases} \tag{1}$$

Let us denote the set of all possible Euclidean distances (resp. angular differences) between two arbitrary points (resp. orientations) by D_{E} (resp. D'_{a}). Let

$\mathit{\Delta}(d_\mathrm{E})$ (resp. $\mathit{\Delta}'(d_\mathrm{a})$) denote the subset of D_E (resp. D'_a) that includes the quantities less than or equal to d_E (resp. d_a). Assume $N_\mathrm{E} = |D_\mathrm{E}|$ and $N_\mathrm{a} = |D_\mathrm{a}|$. Let the polynomials Q_E and Q_a be defined and calculated via the Lagrange interpolation such that we have the following:

$$\begin{aligned} Q_\mathrm{E}(x) &= \textstyle\sum_{k=0}^{N_\mathrm{E}-1} e_k x^k = \begin{cases} 0 \text{ if } \sqrt{x-1} \in \mathit{\Delta}(d_\mathrm{E}) \\ 1 \text{ if } \sqrt{x-1} \in D_\mathrm{E} \setminus \mathit{\Delta}(d_\mathrm{E}) \end{cases} \quad \text{and} \\ Q_\mathrm{a}(x) &= \textstyle\sum_{k=0}^{N_\mathrm{a}-1} a_k x^k = \begin{cases} 0 \text{ if } \sqrt{x-1} \in \mathit{\Delta}'(d_\mathrm{a}) \\ 1 \text{ if } \sqrt{x-1} \in D'_\mathrm{a} \setminus \mathit{\Delta}'(d_\mathrm{a}) \end{cases} \end{aligned} \tag{2}$$

Let the polynomials R and S be defined for τ, m and n and written as follows, where $\tau' = \min(m, n) - \tau + 1$.

$$\begin{aligned} R(x) &= \textstyle\sum_{k=0}^{3} R_k x^k = \begin{cases} 1 & \text{if } x-1 = 0 \\ 0 & \text{if } x-1 \in \{1,2,3\} \end{cases} \quad \text{and} \\ S(x) &= \textstyle\sum_{k=0}^{\tau'} s_k x^k \;= \prod_{k=\tau}^{\min(m,n)} (x-1-k) \end{aligned} \tag{3}$$

Let $\|F|F'\|_{d_\mathrm{E},d_\mathrm{a}}$ return the number of minutia pairs that match in the two fingerprints F and F'. We propose the private fingerprint matching protocol PFM depicted in Fig. 2.

Through the initial steps of the protocol, Alice and Bob's interaction enables Bob to compute $\mathsf{E}(Q_i(t'_j))$, $\mathsf{E}(Q_\mathrm{E}(d_{ij}^2 + 1))$, and $\mathsf{E}(Q_\mathrm{a}(\gamma_{ij}^2 + 1))$. Based on Equation 1, $Q_i(t'_j)$ is zero if $t_i = t'_j$ and is one otherwise. Similarly, based on Equation 2, $Q_\mathrm{E}(d_{ij}^2 + 1)$ is zero if $d_{ij} \in \mathit{\Delta}(d_\mathrm{E})$ and is one otherwise; and $Q_\mathrm{a}(\gamma_{ij}^2 + 1)$ is zero if $\gamma_{ij} \in \mathit{\Delta}'(d_\mathrm{a})$ and is one otherwise. Hence, z_{ij} is zero if the two minutiae p_i and p'_j match and is one, two, or three otherwise. Therefore, based on Equation 3, $R(z_{ij} + 1)$ is zero if p_i and p'_j match and is one otherwise. This implies that $\sigma = \sum_{\forall i,j} R(z_{ij} + 1)$ reflects the total number of minutia pair matchings $\|F|F'\|_{d_\mathrm{E},d_\mathrm{a}}$. Now, based on Equation 3, $S(\sigma + 1)$ is zero if $\sigma \geq \tau$ and is non-zero otherwise. Hence, $r'S(\sigma)$ is equal to zero if $\|F|F'\|_{d_\mathrm{E},d_\mathrm{a}} \geq \tau$ and is random otherwise. Hence, the protocol correctly computes the predicate $\|F|F'\|_{d_\mathrm{E},d_\mathrm{a}} \geq \tau$. Note that the the non-zero requirement is met.

Security against honest-but-curious adversaries is guaranteed by Theorem 1. Security against malicious adversaries can be achieved via standard cryptographic techniques. For the definition of security, the proof of the following theorem, and further discussion on how to achieve full security, please refer to the full version of this paper [9].

Theorem 1. *Protocol* PFM *privately computes the predicate* $\|F|F'\|_{d_\mathrm{E},d_\mathrm{a}} \geq \tau$ *for an honest-but-curious Alice and an honest-but-curious Bob.*

In the full version of this paper [9], we discuss some practical considerations of implementing our protocol and an application of it in realizing a remote fingerprint authentication system. We also compare our protocol to previous works in the literature dealing with the same problem, namely [6,1,2].

$\boxed{(\|F|F'\|_{d_{\mathrm{E}},d_{\mathrm{a}}} \geq \tau, -) \leftarrow \mathsf{PFM}[\mathsf{Alice}(F) \leftrightarrow \mathsf{Bob}(F')](n, m, d_{\mathrm{E}}, d_{\mathrm{a}}, \tau)}$

1. Alice generates a key pair for a homomorphic encryption scheme and sends the public key to Bob. Let the polynomials Q_i, Q_{E}, Q_{a}, R, and S be defined as in Equations 1–3. Let $i \in \{1, \dots, n\}$ and $j \in \{1, \dots, m\}$.
2. Alice computes for all i: $\{\mathsf{E}(b_{ik})\}_{k=0}^{|T|-1}$, $\mathsf{E}(x_i^2 + y_i^2)$, $\mathsf{E}(x_i)$, $\mathsf{E}(y_i)$, $\mathsf{E}(\theta_i^2)$, and $\mathsf{E}(\theta_i)$ and sends them to Bob.
3. Let $d_{ij} = \mathrm{dis}_{\mathrm{E}}((x_i, y_i), (x'_j, y'_j))$ and $\gamma_{ij} = |\theta_i - \theta'_j|$. Bob computes for all i and j: $\mathsf{E}(d_{ij}^2+1) = \mathsf{E}(x_i^2+y_i^2) \oplus (-2x'_j \odot \mathsf{E}(x_i)) \oplus (-2y'_j \odot \mathsf{E}(y_j)) \oplus \mathsf{E}({x'}_j^2 + {y'}_j^2) \oplus \mathsf{E}(1)$ and $\mathsf{E}(\gamma_{ij}^2 + 1) = \mathsf{E}(\theta_i^2) \oplus (-2\theta'_j \odot \mathsf{E}(\theta_i)) \oplus \mathsf{E}({\theta'}_j^2) \oplus \mathsf{E}(1)$.
4. Aided computation $\forall i, j$:
   $(-, \mathsf{E}(Q_{\mathrm{E}}(d_{ij}^2 + 1))) \leftarrow \mathsf{AidComp}[\mathsf{Alice}(sk) \leftrightarrow \mathsf{Bob}(\mathsf{E}(d_{ij}^2 + 1), Q_{\mathrm{E}})](pk, N_{\mathrm{E}} - 1)$.
5. Aided computation $\forall i, j$:
   $(-, \mathsf{E}(Q_{\mathrm{a}}(\gamma_{ij}^2 + 1))) \leftarrow \mathsf{AidComp}[\mathsf{Alice}(sk) \leftrightarrow \mathsf{Bob}(\mathsf{E}(\gamma_{ij}^2 + 1), Q_{\mathrm{a}})](pk, N_{\mathrm{a}} - 1)$.
6. Let $z_{ij} = Q_i(t'_j) + Q_{\mathrm{E}}(d_{ij}^2 + 1) + Q_{\mathrm{a}}(\gamma_{ij}^2 + 1)$. Bob calculates for all i and j: $\mathsf{E}(Q_i(t'_j)) = \bigoplus_{k=0}^{|T|-1} ({t'}_j^k \odot \mathsf{E}(b_{ik}))$ and $\mathsf{E}(z_{ij} + 1) = \mathsf{E}(Q_i(t'_j)) \oplus \mathsf{E}(Q_{\mathrm{E}}(d_{ij}^2 + 1)) \oplus \mathsf{E}(Q_{\mathrm{a}}(\gamma_{ij}^2 + 1)) \oplus \mathsf{E}(1)$.
7. Aided computation $\forall i, j$:
   $(-, \{\mathsf{E}(R(z_{ij} + 1))) \leftarrow \mathsf{AidComp}[\mathsf{Alice}(sk) \leftrightarrow \mathsf{Bob}(\mathsf{E}(z_{ij} + 1), R)](pk, 3)$.
8. Let $\sigma = \sum_{\forall i,j} R(z_{ij} + 1)$. Bob computes $\mathsf{E}(\sigma+1) = \bigoplus_{\forall i,j} \mathsf{E}(R(z_{ij} + 1)) \oplus \mathsf{E}(1)$.
9. Aided computation:
   $(-, \mathsf{E}(S(\sigma + 1))) \leftarrow \mathsf{AidComp}[\mathsf{Alice}(sk) \leftrightarrow \mathsf{Bob}(\mathsf{E}(\sigma + 1), S)](pk, \tau')$.
10. Bob calculates $\mathsf{E}(r'S(\sigma + 1)) = r' \odot \mathsf{E}(S(\sigma + 1))$ for a random r' and sends it to Alice.
11. Alice decrypts the received ciphertext and decides that $\|F|F'\|_{d_{\mathrm{E}},d_{\mathrm{a}}} \geq \tau$ if the resulting plaintext is equal to zero and $\|F|F'\|_{d_{\mathrm{E}},d_{\mathrm{a}}} < \tau$ otherwise.

Fig. 2. The private fingerprint matching protocol

4 Concluding Remarks

We have proposed for the first time a protocol for private fingerprint verification without generic protocols. The proposed protocol has a linear computation complexity in the number of minutia matchings and provides the highest possible privacy guarantee, and hence is highly suitable for match-on-server biometric applications. Designing private fingerprint matching protocols with less computation complexity that are more suitable for match-on-card applications remains a challenging problem for future research.

Acknowledgement. Financial support for this research was provided in part by the Australian Research Council and Alberta Innovates - Technology Futures, in the Province of Alberta in Canada. The first author would like to thank Josef Pieprzyk and Macquarie University for hosting him during part of this research. The authors would like to thank the anonymous reviewers of ACISP 2012.

References

1. Barral, C., Coron, J.-S., Naccache, D.: Externalized Fingerprint Matching. In: Zhang and Jain [10], pp. 309–315
2. Blanton, M., Gasti, P.: Secure and Efficient Protocols for Iris and Fingerprint Identification. In: Atluri, V., Diaz, C. (eds.) ESORICS 2011. LNCS, vol. 6879, pp. 190–209. Springer, Heidelberg (2011)
3. Bringer, J., Chabanne, H., Izabachène, M., Pointcheval, D., Tang, Q., Zimmer, S.: An Application of the Goldwasser-Micali Cryptosystem to Biometric Authentication. In: Pieprzyk, J., Ghodosi, H., Dawson, E. (eds.) ACISP 2007. LNCS, vol. 4586, pp. 96–106. Springer, Heidelberg (2007)
4. Bringer, J., Chabanne, H., Pointcheval, D., Tang, Q.: Extended Private Information Retrieval and Its Application in Biometrics Authentications. In: Bao, F., Ling, S., Okamoto, T., Wang, H., Xing, C. (eds.) CANS 2007. LNCS, vol. 4856, pp. 175–193. Springer, Heidelberg (2007)
5. Goldwasser, S., Micali, S.: Probabilistic Encryption. J. Comput. Syst. Sci. 28(2), 270–299 (1984)
6. Kerschbaum, F., Atallah, M.J., M'Raïhi, D., Rice, J.R.: Private Fingerprint Verification without Local Storage. In: Zhang and Jain [10], pp. 387–394
7. Maltoni, D., Maio, D., Jain, A.K., Prabhakar, S.: Handbook of Fingerprint Recognition. Springer (2009)
8. Paillier, P.: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)
9. Shahandashti, S.F., Safavi-Naini, R., Ogunbona, P.: Private Fingerprint Matching. Cryptology ePrint Archive, Report 2012/219 (2012), http://eprint.iacr.org/2012/219
10. Zhang, D., Jain, A.K. (eds.): ICBA 2004. LNCS, vol. 3072. Springer, Heidelberg (2004)

Minimizing Information Leakage of Tree-Based RFID Authentication Protocols Using Alternate Tree-Walking[⋆]

Kaleb Lee, Colin Boyd, and Juan Gonzalez Nieto

Information Security Institute, Queensland University of Technology,
GPO Box 2434, Brisbane, Queensland 4001, Australia
{leekj,c.boyd,j.gonzaleznieto}@qut.edu.au

Abstract. The privacy of efficient tree-based RFID authentication protocols is heavily dependent on the branching factor at the top layer. Indefinitely increasing the branching factor, however, is not a practical option. This paper proposes an alternate tree-walking scheme as well as two protocols to circumvent this problem. The privacy of the resulting protocols is shown to be comparable to that of linear-time protocols, where there is no leakage of information, whilst reducing the computational load of the database by one-third of what is required of tree-based protocols during authentication. We also identify and address a limitation in quantifying privacy in RFID protocols.

1 Introduction

RFID is a wireless technology designed for convenient automatic identification of physical objects, originally intended to replace bar codes. However this convenience also comes with a risk to privacy, as these objects are commonly carried or used in our daily lives. In particular it is possible to track people based on RFID attached objects they carry. Thus it is critical to ensure that only the minimum amount privacy is leaked when using this technology.

The most commonly used RFID networks are low-cost RFID networks. Such networks typically consist of three components: a back-end database, multiple readers and a large number of RFID tags. Whereas the database and reader are usually workstation class devices, tags are severely limited in terms of computational power and storage. A large amount of research has been focused on increasing the privacy of low-cost RFID networks, yet an increase in privacy often comes at the cost of efficiency on the database. Of particular interest in this paper are tree-based protocols which require comparatively little computation but were later shown to be susceptible to information leakage.

This paper presents a scheme to minimize the leakage of privacy in tree-based protocols before presenting two new protocols. In addition, the paper identifies a limitation in, and proposes an extension to, a current method of measuring

⋆ A full version of the paper is available [5].

privacy leakage in RFID protocols. The protocols proposed are shown to leak significantly less information compared to currently proposed protocols whilst only requiring one third of the computation when pre-computation is used. Also discussed is the effort required to attack the protocols, and it is shown that an attacker is required to perform significantly more work to succeed.

Tree-Based Protocols. Tree-based authentication was proposed in 2004 by Molnar et al. [6], referred to as the CR/MW Scheme. They introduced the concept of a hierarchical arrangement of tags that significantly improves the efficiency of the authentication protocol at the reader. Let N be the total number of tags in a RFID system. A tree-based authentication protocol considers a M-ary tree, where tags correspond to the leaves of the tree and each edge of the tree is assigned a random secret. Each tag T_i has associated a tuple of secret values $(k_{i,1}, \ldots, k_{i,l})$, where $l = \log_M N$ is the depth of the tree and $k_{i,j}$ is the secret corresponding to the j^{th} edge in the tree path from the root to the i^{th} leaf. The main motivation behind this approach is to solve the scalability issue associated with linear protocols. This approach has since been considered as one of the most efficient methods of private authentication proposed [2], however it is not without limitations and drawbacks. One of the most significant drawbacks of using a M-ary tree based approach is the leakage of information when secrets of tags are revealed. Where in linear protocols there exists only one unique secret per tag, in tree-based protocols there are multiple secrets per tag, of which most are shared among other tags. Thus revealing the secrets of one or multiple tags will dramatically reduce the privacy of the system as a whole.

2 Limitations of Privacy Leakage Measurement

The amount of privacy leakage has been analyzed and quantified in a number of works [4,2,7,3]. This paper will focus on the work of Avoine *et al.* [2], who measure leakage as the probability that an adversary can distinguish between two tags in a privacy attack experiment. However this defintiion does not take into consideration the ability for an adversary to view transcripts of other successful sessions, a potential major increase of leakage of information. For example, in a library scenario transcripts can be obtained by an adversary eavesdropping communication between a reader (typically close to doors) and tags (attached to books) as they are carried out. Having access to transcripts significantly increases the probability of the adversary winning in the CR/MW scheme as the adversary is able to compare all secrets at the same time. Thus this section proposes an extension to the experiment that also considers the adversary's ability to view successful protocol transcripts. The experiments will subsequently be compared.

In the new experiment, only steps 2 and 3 need to be changed. The experiment is now as follows:

1. The adversary draws one tag, T_m, and obtains its full set of secrets, $(k_{m,1}, \ldots, k_{m,l})$. The tag is put back into circulation.

2. The adversary is then randomly given a tag T_j and is allowed to query the tag and request successful protocol transcripts as much as it wants. However, the adversary is not allowed to reveal the secrets of the tags.
3. The adversary is now given two tags, T_a and T_b such that $T_j \in \{T_a, T_b\}$. and is allowed to query both tags and request successful protocol transcripts. The adversary wins the experiment if it can *definitely* output i, such that $T_i = T_j$.

Remark: In the above definition, following the definition of Avoine *et al.* [2] the adversary is said to *definitely* win if and only if it can obtain from T_m the keys necessary to distinguish between T_a and T_b. In other words, here we are interested in measuring how leakage of the secrets of a tag improves the adversary's advantage in tracing other tags.

2.1 Comparison of Experiments

In this section, the proposed extension of the experiment will be compared with the original using the CR/MW scheme. Before further detailing the results, it should be noted that schemes are considered under a 3-layer tree as opposed to an M-ary tree. In a 3-layer tree each tag T_j is only required to store three keys: $k_{j,1}, k_{j,2}, k_{j,3}$, as opposed to the b keys, where b is the branching factor of the tree. Whereas in a balanced 3-layer tree the branching factor is always $N^{\frac{1}{3}}$, where N is the total number of tags in the system, an M-ary-tree does not set any limits on its branching factor and consequently the number of layers, making direct comparisons difficult.

The probability of success in the extended experiment of the CR/MW scheme would become as follows:

$$\begin{aligned}\Pr(\mathsf{win}) = {} & \Pr((k_{m,1} = k_{a,1}) \wedge (k_{m,1} \neq k_{b,1})) \vee \Pr(((k_{m,1} \neq k_{a,1}) \wedge (k_{m,1} = k_{b,1})) \vee \\ & \Pr((k_{m,2} = k_{a,2}) \wedge (k_{m,2} \neq k_{b,2})) \vee \Pr((k_{m,2} \neq k_{a,2}) \wedge (k_{m,2} = k_{b,2})) \vee \\ & \Pr((k_{m,3} = k_{a,3}) \wedge (k_{m,3} \neq k_{b,3})) \vee \Pr((k_{m,1} \neq k_{a,1}) \wedge (k_{m,1} \neq k_{b,1}))\end{aligned}$$

For a more meaningful comparison, two cases of the CR/MW scheme will be considered in the new experiment, its best case and worst case scenario. In its best case scenario, all $k_{*,3}$ and $k_{*,2}$ secrets are assumed to be unique with its layer. Thus there are N possible values of k_3 and $N^{\frac{2}{3}}$ possible values of k_2. In its worst case scenario, however, all secrets are assumed to be unique only within its branch. As such there are $N^{\frac{1}{3}}$ possible values of k_3 and $N^{\frac{1}{3}}$ possible values of k_2. The possibilities are as follows:

$$\Pr(\mathsf{win} - \mathsf{best}) = \frac{2N^{\frac{1}{3}} - 2}{N^{\frac{2}{3}}} + \frac{2N^{\frac{2}{3}} - 2}{N^{\frac{4}{3}}} + \frac{2N - 2}{N^2}$$

$$\Pr(\mathsf{win} - \mathsf{worst}) = \frac{6N^{\frac{1}{3}} - 6}{N^{\frac{2}{3}}}$$

The results of the comparison are presented in Table 1. It should be emphasized that in the old experiment, the values presented use the same tree-structure

Table 1. Comparison of Experiments

No. of Tags	OldExp	NewExp-Best	NewExp-Worst
100	35%	45%	99%
500	22%	26%	66%
1000	18%	20%	54%
5000	11%	12%	33%

as the worst case scenario in the extended experiment. In the table it is shown throughout the experiments under the same conditions, probability of the adversary winning when given successful protocol transcripts increases by three-fold. Interestingly, it can be observed that, even in the best case scenario, the probability of the adversary with transcripts winning is still higher than worst case of the one without. Nevertheless, it is apparent that this additional power of the adversary does not have any effect on group and linear protocols.

On a side note, by giving the adversary transcripts the alternate tree-walking scheme does not gain any privacy advantage over the CR/MW scheme. For the rest of the paper the best case scenario from the extended experiment will be used as a baseline for the CR/MW scheme.

2.2 Leakage Results of Current Protocols

For the purpose of comparison, three protocols are used as a baseline. In addition to the CR/MW protocol above, the group protocol and linear-time protocol are used. The results shown below can be obtained using the experiment from the previous section. If N is the total number of tags in the system, the results are as follows.

– Group Protocols [1]

$$\Pr(\mathsf{win}) = \frac{2N^{\frac{1}{2}} - 2}{N} + \frac{2N - 2}{N^2}$$

– Linear(-Time) Protocols

$$\Pr(\mathsf{win}) = \frac{2N - 2}{N^2}$$

3 Alternate Tree-Walking (ATW)

The privacy of the CR/MW scheme is heavily dependent on the branching factor on the top layer [3], but it is not feasible to indefinitely increase the branching factor. Thus this paper proposes the alternate-tree walking scheme to circumvent this problem. The resulting scheme significantly reduces the amount of leakage compared to traditional tree-based (and group) protocols whilst maintaining a reasonable amount of computational load on the database. The core concept of alternate-tree walking is to start authentication from a layer in-between the top and bottom layers of a tree, as opposed to authenticating sequentially from the top to bottom. Although the concept can be applied to any tree with more than three layers, for simplicity the rest of the paper will consider a tree with only three layers.

Whereas the traditional CR/MW scheme authenticates sequentially down the tree, the ATW scheme starts from the middle back to the top before working down the tree. In essence, this approach is to achieve benefits of a large branching factor without altering the structure of the tree. In a tree structure it is required that all secrets be unique within a branch, thus in traditional tree-based approach the number of possible unique secrets in the initial branch is limited to its branching factor. However, by starting authentication from a middle layer, the number of possible unique secret values has increased to $N^{\frac{2}{3}}$ from $N^{\frac{1}{3}}$ of the CR/MW Scheme, and $N^{\frac{1}{2}}$ of group protocols.

3.1 The ATW Protocol

Simply by analyzing protocol transcript it is possible for an adversary to gain the same amount of information as the original approach. Thus the remainder of this section will be used to propose protocols designed to take advantage of the reduced information leakage of the ATW scheme.

Table 2. AlternateTree Walking Protocol (ATW Protocol)

Reader		**Tag**
		k_1, k_2, k_3
	$\xrightarrow{N_R}$	
	$\xleftarrow{H(N_T \| N_R \| k_2) \| H(N_R \| k_1) \oplus H(N_T \| k_3), N_T}$	

Table 2 shows the ATW protocol designed to take advantage of the alternate-tree walking scheme. Evidently, transcripts of this protocol does not leak any more information than required. In the protocol, the database first verifies the message by checking if

1. $H(N_T \| N_R \| k_2) = H(N_T \| N_R \| k_{i,2})$ for some $i \in \{1, \ldots N\}$, and
2. $H(N_R \| k_1) \oplus H(N_T \| k_3) = H(N_R \| k_{i,1}) \oplus H(N_T \| k_{i,3})$.

Note that the database only computes the value of $H(N_R \| k_{i,1}) \oplus H(N_T \| k_{i,3})$ for those i that satisfy condition 1.

Due to the use of $H(N_R \| k_1) \oplus H(N_T \| k_3)$ to authenticate layer 1 and 2, it should be noted that the leakage of information using this protocol is less than that given in the previous section. Let $D_i = H(N_R \| k_{i,1}) \oplus H(N_T \| k_{i,3})$, for $i = 1, \ldots, N$. The possible outcomes of winning are:

- C_1: $(k_{m,1} = k_{a,1}) \wedge (k_{m,1} \neq k_{b,1})$
- C_2: $((k_{m,1} \neq k_{a,1}) \wedge (k_{m,1} = k_{b,1})$

- C_3: $(D_m = D_a) \wedge (D_m \neq D_a)$
- C_4: $(D_m \neq D_a) \wedge (D_m = D_b)$

For consistency the best case tree-structure, where there are N possible values of k^3 and $N^{\frac{2}{3}}$ possible values of k^2, will be used. The overall probability of winning is:

$$\begin{aligned}\Pr(\mathsf{win}) &= \Pr(C_1) \vee \Pr(C_1) \vee \Pr(C_2) \vee \Pr(C_3) \vee \Pr(C_4) \\ &= \frac{2N^{\frac{2}{3}} - 2}{N^{\frac{4}{3}}} + \frac{2N^{\frac{4}{3}} - 2}{N^{\frac{8}{3}}}\end{aligned}$$

The resulting leakages are shown in Table 3. Note in particular that the leakage of the ATW protocol has been reduced by more than 50% than when the ATW scheme was compared using the CR/MW protocol. Nevertheless, the protocol still leaks twice as much information as linear protocols.

Table 3. Protocol Leakage Comparison

No. of Tags	CR/MW	Group	Linear	ATW
100	9%	20%	2%	4%
500	3%	8.9%	0.4%	0.08%
1000	2%	6.3%	0.2%	0.04%
5000	0.7%	2.8%	0.04%	0.08%

Further Reduction of Privacy Leakage for Small Networks. This section proposes a modification of the protocol proposed in the previous section that aims to minimize the leakage of information for small networks. The protocol is shown in Table 4.

Table 4. Alternate-Tree Walking Protocol for Small Networks (ATWS-Protocol)

Reader		**Tag**
		k_1, k_2, k_3
	$\xrightarrow{N_R}$	
	$H(N_R \Vert S_2) \oplus H(N_T \Vert S_1) \Vert H(N_T \Vert S_2) \oplus H(N_R \Vert S_3)), N_T$ $\longleftarrow$	

By requiring the database to compute $H(N_R \| S_1) \oplus H(N_T \| S_2)$ first, followed by $H(N_R \| S_2) \oplus H(N_T \| S_3)$ the probability of the adversary winning has decreased to:

$$\Pr(\mathsf{win}) = \frac{4N - 4}{N^2}$$

Nevertheless, the number of required computations has increased to $N^{\frac{2}{3}} + 2N^{\frac{1}{3}}$. However even though this scheme is aimed at small networks, the increased computational requirement is comparatively minimal compared to linear protocols. A comparison is shown in Table 5; evidently the ATWS provides a comparable level of privacy to linear protocols at significantly less computational cost.

Table 5. ATW and ATWS Protocol Results Comparison

No# Tags	ATW	Linear	ATWS
100	9.3%	2%	2.1%
200	5.8%	1%	1%
300	4.5%	0.7%	0.7%
400	3.7%	0.5%	4%

Further Reduction of Computation Using Pre-computation. This section discusses the use of pre-computation by the database. By pre-computing the values of $H(N_R \| k^*)$, it is possible to decrease the amount of time and computation required during authentication. Although the method can be applied to both proposed schemes, it would be most useful when applied to the ATWS-Protocol. By pre-computing the values of $H(N_R \| S_2)$ and $H(N_R \| S_3)$ in $H(N_R \| S_2) \oplus H(N_T \| S_1) \| H(N_T \| S_2) \oplus H(N_R \| S_3))$, it is possible to reduce the number of computations during authentication from $N^{\frac{2}{3}} + 2N^{\frac{1}{3}}$ to $N^{\frac{1}{3}}$, allowing the protocol to complete authentication with less computations than the CR/MW scheme.

3.2 Other Considerations

In key exchange protocols a security parameter governs the length of secrets and determines the effort required by an adversary to mount a brute-force attack. Due to the limited storage it is desirable to make secrets as short as possible for RFID protocols, making brute-force attacks seemingly more attractive than for traditional key-exchange protocols. Assuming that the total amount of memory given to a tag for storing secrets is K bits, the amount of work required is shown in Table 6. It can be observed that linear protocols require the most amount of work to attack followed by the ATW-Scheme. It should be noted that both protocols based on the ATW-Scheme require the same amount of work to attack.

Table 6. Work Comparison

	Tree	Group	Linear	ATree
Work Required	$3(2^{\frac{K}{3}})$	$2(2^{\frac{K}{2}})$	2^K	$2^{\frac{K}{3}} + 2^{\frac{2K}{3}}$

4 Conclusion

This paper analyzed the leakage of information in linear, tree-based and group-based RFID protocols as well addresses a limitation of a current privacy measurement method. The paper also proposed two protocols, the ATW protocol and ATWS protocol, which were showed to leak substantially less privacy compared to analyzed protocols. The increased computational requirement of the proposed protocols can also be offloaded though the use of pre-computation.

The resulting protocol can be completed using less computations than that of tree-based protocols. The resulting P-ATWS protocol was able to match linear protocols in terms of privacy but at the same time only require one-third of what is required of tree-based protocols.

References

1. Avoine, G., Buttyán, L., Holczer, T., Vajda, I.: Group-based private authentication. In: IEEE International Workshop on Trust, Security, and Privacy for Ubiquitous Computing – TSPUC, Helsinki, Finland, pp. 1–6. IEEE Computer Society Press (June 2007)
2. Avoine, G., Dysli, E., Oechslin, P.: Reducing Time Complexity in RFID Systems. In: Preneel, B., Tavares, S. (eds.) SAC 2005. LNCS, vol. 3897, pp. 291–306. Springer, Heidelberg (2006)
3. Buttyán, L., Holczer, T., Vajda, I.: Optimal Key-Trees for Tree-Based Private Authentication. In: Danezis, G., Golle, P. (eds.) PET 2006. LNCS, vol. 4258, pp. 332–350. Springer, Heidelberg (2006)
4. Huang, X.: Quantifying information leakage in RFID systems. In: 10th International Conference on Advanced Communication Technology, vol. 1, pp. 84–89 (February 2008)
5. Lee, K., Nieto, J.G., Boyd, C.: Minimizing information leakage of tree-based RFID authentication protocols using alternate tree-walking (2012), http://eprints.qut.edu.au/49883/
6. Molnar, D., Wagner, D.: Privacy and Security in Library RFID: Issues, Practices, and Architectures. In: CCS 2004: Proceedings of the 11th ACM Conference on Computer and Communications Security, pp. 210–219. ACM, New York (2004)
7. Nohl, K., Evans, D.: Quantifying Information Leakage in Tree-Based Hash Protocols (Short Paper). In: Ning, P., Qing, S., Li, N. (eds.) ICICS 2006. LNCS, vol. 4307, pp. 228–237. Springer, Heidelberg (2006)

ICAF: A Context-Aware Framework for Access Control

A.S.M. Kayes, Jun Han, and Alan Colman

Faculty of Information and Communication Technologies,
Swinburne University of Technology, Melbourne, Australia
{akayes,jhan,acolman}@swin.edu.au

Abstract. Context-aware systems acquire and integrate multi-faceted knowledge about their environments in order to make decisions. A number of attempts to build frameworks for context-aware systems have been made, but these have not provided adequate support for context-aware access control. In this paper, we present a framework for context-aware access control and its prototype implementation. The framework includes a context model for classifying and capturing access control-oriented contextual information, a situation model for identifying and defining contextual conditions of concern, and a policy model for specifying context-aware access control policies.

Keywords: Context-aware access control, context modeling, context reasoning, situation modeling, access control policy.

1 Introduction

Computer systems have shifted from fixed desktop environments to pervasive computing environments in which, as Weiser [1] describes, resources or services should be available for the life of everyday users in an 'anywhere, anytime' fashion even when they are on the move. A key challenge in such pervasive environments is the control over access to these resources or services. Unlike traditional access control, access control decisions in pervasive environments need to take into account the relevant *contextual information* such as time and location that reflect the dynamically changing conditions of the environments [2].

Access control is a mechanism to determine whether a request to access resources in a system should be permitted or denied. The traditional access control models (e.g., [3]) do not provide adequate functionality to adapt to and incorporate *dynamically changing contexts*. More recently, a number of research efforts (e.g., [4-6]) have attempted to design access control models that consider context information, focusing on time and location. In general, there are also other types of environmental factors or context information that need to be considered. When considering whether or not a doctor can access a patient's medical records, for example, the purpose of the request, the health condition of the patient and the relationship between the doctor and the patient (treating physician or not) are all possible additional factors that need to be considered beyond the time and location of the particular request.

In this paper, we introduce a general framework for context-aware access control. It has three major components: an access control-oriented context model for classifying and capturing contextual information, a situation model for identifying and defining relevant environmental or contextual conditions, and a policy model for defining context-aware access control policies. A prototype has been developed to realize this framework.

2 Research Motivation

As an example of the type of situations that a context-aware access control model has to be considered, let us consider the following scenario: *patient Bob has been hospitalized due to a heart attack and is in the emergency room of the hospital. While not being Bob's usual treating physician, Jane, a resident doctor of the hospital, is required to treat Bob and needs to access (read and write) Bob's medical records from the emergency room at 3:00 pm on 18 January 2012.*

Normally, only a patient's treating physician is able to access the patient's medical records. In the above emergency scenario, Jane, while not being the treating physician, needs to access and should be given access to Bob's medical records. The relevant contextual information concerning the above scenario of access control include: Jane's *personal role* is 'Resident Doctor', Jane is *located in* the 'EmergencyRoom' where Bob is, Bob's current *health state* is 'Critical', and Jane's *relationship* to Bob is that of 'NonTreatingPhysician'. In addition, the *purpose* of Jane's request for accessing Bob's medical records is for '*Treatment*'. In making this access control decision, all these factors need to be taken into account. Furthermore, when the situation changes (e.g., Bob has come out of emergency and moved to a hospital ward), decisions on further access requests by Jane to Bob's medical records may change accordingly (e.g., denied).

To support such context-aware access control in a computer application like the medical record management system, we need to consider the 4Ws: *who* (user) wants to access *what* (resources), *when* (current contexts), and *why* (purpose for resource access). In particular, a general framework is required to manage the access to resources in such applications by taking into account the different types of changing environmental factors that impact on the access control decisions. The framework should support (i) different types of context information, (ii) different types of purpose-oriented situations, and (iii) context-aware access control policy rules.

3 Related Work

Wang et al [7] have proposed an OWL encoded CONtext ONtology (CONON) for modeling context information in pervasive environments. The CONON ontology helps to share a common understanding of the structure of contextual information concerning users, places, and devices in order to support semantic interoperability and reuse of domain knowledge. Henricksen et al [8] have developed a Context Modeling Language (CML) and a tool that translates CML-based context models to an OWL representation for the purpose of utilizing the OWL technology. However, these existing general context models do not provide direct support for concepts related to access control such as resource owners, or for relationships between context entities such as that between the resource requester and the resource owner.

The research presented in [7,8] considers the concept of situation as a means to identify high-level context information (*user* activities/states). In making access control decisions, however, in addition to the state of the *user*, the states of the *resource*, its *owner* and *environment* are also important considerations. Furthermore, the *purpose* of resource access is also part of the consideration.

The generalized role based access control model GRBAC [4] is an extension of the traditional role based access control (RBAC) model [3] by introducing the notions of subject, object and environment roles, where the environment roles are used to model the environmental context (day time, night time, etc.). The GEO-RBAC model [5] also extends the RBAC model, where authorizations to access resources are based on the assigned role and location of the user. Toninelli et al [6] have proposed a context-aware access control framework that provides resource access permission based on such factors as resources availability, actor roles/identities and environmental conditions (such as time, other available resources, etc.). These existing research efforts either consider only specific types of context information or do not provide a classification of the relevant contextual factors for access control. As such, none of the existing context-aware access control approaches sufficiently support the requirements identified in Section 2.

4 A Context-Aware Framework for Access Control

Our context-aware framework for access control supports the following three phases of developing context-aware access control applications: modeling *context*, modeling *situation*, and modeling *context-aware access control policy*.

4.1 Representation and Modeling of Context

The most accepted definition of context is given by Dey [2]. However, this definition is not specific enough for access control as it does not identify the access control-specific entities. From the running application scenario, we can see that the access control entities are *user*, *resource*, *resource owner* and their *environments*. We specialize Dey's definition of context to cover access control applications as follows:

'Context information' used in an access control decision is defined as any relevant information about the state of a relevant ***entity*** *(user, resource, resource owner and their environments) or the state of a relevant* ***relationship*** *between the entities.*

Focusing on the context information relevant to making access control decisions, we classify context entities as follows:

- *User* context is any relevant information about the user or resource requester, who makes a request, e.g., the user's *Identity* and *Role*;

- *Resource* context is any relevant information about the resource, which is being requested, e.g., the *PrivacyAttribute* of resource information;
- *Owner* context is any relevant information about the resource owner, e.g., the owner's *Identity* and *Category*;
- *Environment* context is any relevant information about the environments of the resource, owner and user, e.g., the *Status* of the *Emergency Room* at a specific time.

We have further classified the *environment entities* into the following sub-categories:

- *Temporal environment* context – information about time characteristics, e.g., *RequestTime*;
- *Spatial environment* context – information about location characteristics, e.g., *LocationAddress*;
- *Spatio-temporal environment* context – information that is about a specific location at a particular time, e.g., the *Status* of the *Emergency Room* at a specific time;
- *Social environment* context – information about relationships between entities, e.g., the *Relationship* between the user and the resource owner;
- *Other environment* context – information other than the above four types of environment entities, e.g., the *HealthState* of a patient, which is deduced based on the patient's medical data possibly obtained through wearable sensors.

We have created a conceptual context model to represent our definition and classification of access control-oriented context. Ontology has been used to represent context models in several research works (e.g., [7]) due to its expressiveness and capability for formal representation and reasoning. We also adopt an ontology-based representation of our context model. Figure 1 presents a high-level ontology model for our entity-based context model, where the superclass-subclass relationships between context entities are represented using the 'is-a' (is a subclass of) properties, and the other relationships using 'user-defined' properties. The context entities (classes) are organized into a hierarchy. *User*, *Resource*, *Resource Owner* and *Environment* are the four main context entities in the hierarchy. The *Environment* entity is related to the *other* three main entities, and is further specialized into its sub-classes.

For a particular context-aware access control application, the above context model can be instantiated to capture the application-specific context entities and information, i.e., resulting in a domain/application-specific context model. For the motivating scenario, for example, we have some of context entities and information as follows:

- The user's *Identity* and *Role* are part of the context information about the *User* (Jane and ResidentDoctor). Her *LocationAddress* and *RequestTime* are context information concerning the user's *Spatial* and *Temporal environments.*
- The *PrivacyAttribute* of Bob's medical records is part of the context information about the *Resource* (Bob's medical records).
- The resource owner's *Identity* and *Category* are part of the context information about the *Owner* (Bob, Patient).

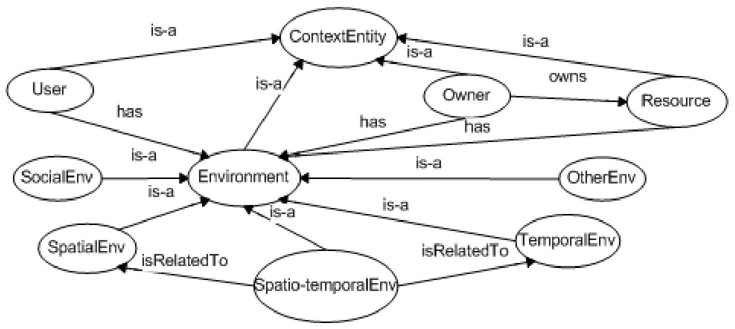


Fig. 1. Context model for access control

- The patient (Bob) is an environment entity relevant to the access request. His *LocationAddress* and *HealthState* are part of the *Spatial* and *Other* context information concerning the patient. Note that Bob is both the resource owner and the patient in this case; in general, however, this does not have to be the case (e.g., the medical records of the patient's mother may need to be consulted in certain situations).
- The *Relationship* (NonTreatingPhysician) between Jane and Bob is context information about the *Social environments* of Jane and Bob.
- The *Status* of the *Emergency Room* at 3:00 pm on 18 January 2012 is part of the context information about the *Spatio-temporal context* entity *EmergencyRoom*.

4.2 Representation of Situation

In the context-aware literature (e.g., [7]), existing *situation* definitions typically describe the state or activity of the *user*. However, these definitions are limited when considering access control in pervasive environments, where a *user* can access a *resource* from a particular *environment* (e.g., a patient is in a *critical* health condition) for a certain *purpose*. In addition to the state of the *user*, the states of the *resource*, its *owner* and their *environments* are also important considerations. Our definition of *purpose-oriented situation* is as follows:

A 'situation' is defined as a specific subset of the complete state of the universe of access control entities that are relevant to a certain goal or purpose of a resource access request.

Here, the universe of access control entities is formed or defined by the context entities from the context model. A situation is a set of values whose types are defined by a domain-specific context model. These values are determined by what the system needs to know given its current state in order to make the access control decision.

Our *purpose-oriented situation model* S is defined using a tuple as follows:

$$S = \{p, c_1, c_2, c_3, \ldots, c_n\}$$

where p is the *purpose*, and c_i (i = 1, 2, 3, …) are the states of relevant entities with each relevant entity attribute taking on a specific value. The situation to capture the access control condition from the motivating scenario can be defined as follows:

```
emergencyMedicalRecordAccess = {
        Purpose = 'Treatment';
        User_Role = 'ResidentDoctor';
        Owner_Category = 'Patient';
        Resource_PrivacyAttribute = 'EmergencyMedicalRecords';
        User_LocationAddress = 'EmergencyRoom';
        Owner_LocationAddress = 'EmergencyRoom';
        User_Owner_Relationship = 'NonTreatingPhysician';
        Patient_HealthState = 'Critical';
        Owner_Identity = Patient_Identity;
      }
```

4.3 Representation of Context-Aware Access Policies

Our *context-aware policy model* uses an identified *situation* as a condition to make context-aware access control decisions. When a user sends a request to access resources, the required *situation* or condition needs to be identified and evaluated based on the relevant context information.

Our access control policies are essentially security policies related to access control decision making and they specify whether a subject is *permitted* or *denied* access to a set of target objects for their specific action or sequence of actions, when a set of constraints (a condition) are satisfied. In particular, our *context-aware policy model* is composed of four basic elements: user or *subject (S)*, resource or *object (O)*, situation or *condition (C)*, and access *action (A)*. This model captures *which* parts of the resources a *user* can *access* and under *what* conditions. The model uses an identified situation as the condition for making context-aware access control decisions. The context-aware access control policy for the motivating scenario from Section 2 can be defined as follows:

```
emergencyMedicalRecordAccessPolicy = {
        S = 'ResidentDoctor';
        O = 'EmergencyMedicalRecords';
        C = 'emergencyMedicalRecordAccess';
        A = 'Write';
      }
```

5 ICAF Prototype

Figure 2 depicts the overall system architecture of our intelligent context-aware framework (ICAF) for access control. The ICAF architecture for access control consists of three layers: *sensor*, *context*, and *service*. It supports the building of context-aware access control (CAAC) services or applications.

As part of the prototype implementation of the framework, we have implemented a number of ICAF components in Java 2 Platform Standard Edition (J2SE) using open source tools. Some of them are Context Providers (CPs), Context Interpreter (CI) and

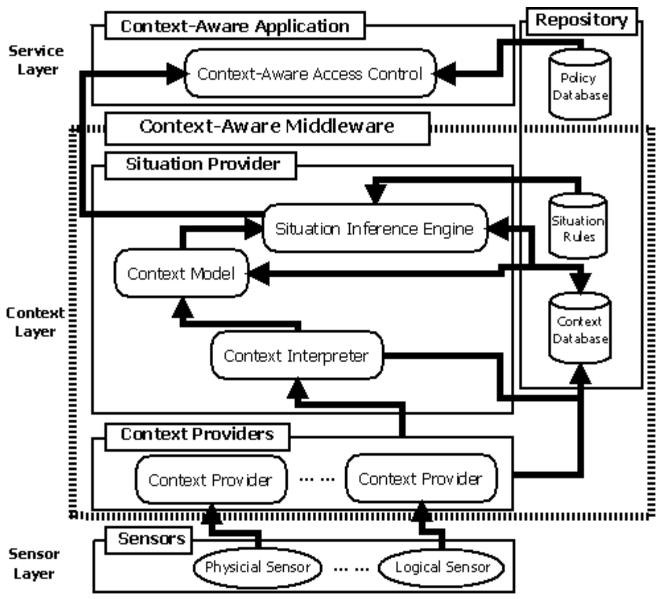


Fig. 2. Overall system architecture of our ICAF framework

Situation Inference Engine (SIE). A CP receives the low-level, raw context data from the physical or logical sensors, and then generates the context information. (Note that in our prototype, Java programs are used to simulate the physical and logical sensors.) We have used the Protégé-OWL graphical ontology API[1] to implement the context model (context ontologies). We have implemented the CI to deduce high-level context information from low-level context information. To define and identify situations, we have used SWRL[2] rules. The Jess Rule Engine[3] is used to implement SIE for executing the SWRL rules. The *Service Layer* uses an identified *situation* from SIE as a condition to evaluate context-aware policies stored in the *Policy Database*. XACML is used as a policy language, and Sun XACML[4] is used as XACML engine.

Referring back to our motivating scenario from Section 2, the sensors provide the raw context data (i.e., the information about Jane, Bob, Emergency Room, etc) for CP to convert to properly formatted context information. Then, the CI infers additional high-level context information such as Bob's health state. The context ontology database receives and records both the low-level and high-level context information. The identified situations are defined in SWRL and stored in the situation rules database. The access control policies for the healthcare application are defined using XACML and are stored in the policy database. When Jane's request for accessing Bob's medical records comes, the SIE is asked to evaluate the access condition (situation) using

[1] Protégé OWL. `http://protege.stanford.edu/download/download.html`
[2] SWRL (Semantic Web Rule Language). `http://www.daml.org/swrl`
[3] Jess Rule Engine. `http://www.jessrules.com/jess/download.shtml`
[4] Sun's XACML Implementation. `http://www.sunxacml.sourceforge.net`

the relevant context information in the context ontology database. As the condition is evaluated to be true, the system grants Jane access to Bob's medical records.

6 Conclusion

The use of context is important in many access control applications, where users can only access certain resources depending on the prevailing contextual conditions. In this paper, we have introduced a framework for context-aware access control, incorporating context into access control policies, so that when a user sends a request to access resources, the access control decision can take into account the relevant situation present. Its key components include an access control oriented context model, a purpose-oriented situation model, and a context-aware access control policy model. As future work, we intend to test our framework in various real-world application domains.

Acknowledgment. Jun Han is supported by the Qatar National Research Fund (QNRF) under Grant No. NPRP 09-069-1-009. The statements made herein are solely the responsibility of the authors.

References

1. Weiser, M.: Some Computer Science Issues in Pervasive Computing. Communications of the ACM 36(7), 75–84 (1993)
2. Dey, A.K.: Understanding and Using Context. Personal and Ubiquitous Computing 5(1), 4–7 (2001)
3. Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Role-based Access Control Models. IEEE Computer 29(2), 38–47 (1996)
4. Covington, M.J., Moyer, M., Ahmad, M.: Generalized Role-based Access Control for Securing Future Applications. In: Proceedings of the 23rd National Information Systems Security Conference, Baltimore, USA (2000)
5. Bertino, E., Catania, B., Damini, M.L., Perlasca, P.: GEO-RBAC: A Spa-tially Aware RBAC. In: Proceedings of the 10th ACM Symposium on Access Control Models and Technologies, pp. 29–37. ACM, New York (2005)
6. Toninelli, A., Montanari, R., Kagal, L., Lassila, O.: A Semantic Context-Aware Access Control Framework for Secure Collaborations in Pervasive Computing Environments. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 473–486. Springer, Heidelberg (2006)
7. Wang, X.H., Zhang, D.Q., Gu, T., Pung, H.K.: Ontology Based Context Modeling and Reasoning Using OWL. In: Proceedings of the PerCom Workshops (2004)
8. Henricksen, K., Indulska, J., Rakotonirainy, A.: Modeling Context Information in Pervasive Computing Systems. In: Mattern, F., Naghshineh, M. (eds.) PERVASIVE 2002. LNCS, vol. 2414, pp. 167–180. Springer, Heidelberg (2002)

Non-malleable Instance-Dependent Commitment in the Standard Model[⋆]

Wenpan Jing, Haixia Xu, and Bao Li

Graduate University of Chinese Academy of Sciences,
No.19A Yuquan Road, 100049 Beijing, China
{wpjing,hxxu,lb}@is.ac.cn

Abstract. An instance-dependent commitment (IDC) scheme takes an instance in a promise problem as public input at each time of committing and separately achieves statistical hiding and statistical binding when the instance is from different subsets of the promise. In this paper, we define a new security property called "instance-non-malleability " for the IDC. It requires the non-malleability of the instances as well as the committed messages. Instance-non-malleability is not only stronger than previous definitions of non-malleability for commitments, but can be achieved in the standard model as well. We also present a general construction of the non-interactive instance-non-malleable IDC.

Keywords: non-malleability, instance-dependent commitment, zero-knowledge proof.

1 Introduction

1.1 Motivation

It has shown that commitment schemes and zero-knowledge protocols have some symmetric properties [7]. The hiding and binding properties of the commitment schemes can be translated to the zero-knowledge and soundness properties of the zero-knowledge protocols respectively. Since it is impossible for a commitment to be both statistically hiding and statistically binding at the same time, there are difficulties in translating SZKP (zero-knowledge protocols for which both zero-knowledge and soundness properties are statistically achieved), to a corresponding commitment scheme. Itoh et. al. proposed a cryptographic primitive called the "instance-dependent commitment" in 1994 [12], which can be applied to solve this problem. An instance-dependent commitment(IDC) scheme takes an instance "x" in a promise problem $\prod$ as a public input to separate the requirements for the hiding and binding properties. A promise problem is composed of two non-intersection sets, with one of the subsets containing the "yes-instances" and the other containing the "no-instances"(see section 2.2 for the formal definition). For a statistical zero-knowledge proof protocol, we require the zero-knowledge property while "$x \in \prod_{Yes}$" to protect the prover, and we require soundness

[⋆] Supported by the National Natural Science Foundation of China (No.61070171) and the National Basic Research Program of China (973 project) (No.2007CB311201).

while "$x \in \prod_{No}$" to ensure that the receiver cannot be cheated. Correspondingly, the instance-dependent commitment achieves statistical hiding while "$x \in \prod_{Yes}$" and statistical binding while "$x \in \prod_{No}$". For a promise problem, if there exists a statistical hiding and statistical binding instance-dependent commitment, an SZKP can be constructed [7].

With these special properties, instance-dependent commitment scheme can replace some ordinary commitment schemes in constructing zero-knowledge protocols. Therefore, studying special security properties of an instance-dependent commitment such as non-malleability can help us to build upper-level protocols that are more secure.

We observe that for a promise problem with "hard relations" (see definition 1 and remark 1), an SZKP sometimes can be generally constructed with an IDC using a pattern called the "commit-challenge-answer" pattern[13]. However, the non-malleable commitments according to previous definition[11] cannot assure the non-malleability of the ZKP. In previous definition of the non-malleable commitment, all public parameters are required to remain unchanged and only the committed message is taken into consideration while measuring non-malleability. Since the instance x is necessary according to upper-level protocols, it is different from the usual common reference string (CRS) for an IDC. Therefore, even the commitment is non-malleable according to previous definitions, given a commitment to m, there might be an adversary who is able to generate a commitment to a related message m' using a different instance x' which is related with the x. The ZKP constructed with the "commit-challenge-answer" pattern then will be malleable. Therefore, a stronger definition of non-malleability for the commitments is inspired. We try to let the new definition of non-malleability of the commitments be consistent with the non-malleability of the ZKP and thus provide stronger security for the upper-level protocols.

1.2 Contribution

In this paper, we first study the non-malleability of the IDC and make a new definition called instance-non-malleability.

We hope that by executing the scheme, the ability of an adversary to commit under a certain type of instance does not increase. Moreover, it is reasonable to assume that the adversary has the ability to impersonate the third party and generate its own instances. Hence, it is necessary to create a new definition of non-malleability that takes the instance into consideration.

We propose a definition of non-malleability for the IDC called *instance-non-malleability*. We consider a binary relationship $R((x, m), (x_{\mathcal{A}}, m_{\mathcal{A}}))$ in our definition instead of only $R(m, m_{\mathcal{A}})$ considered in previous definitions of non-malleability.

We also present a general non-interactive construction of instance-non-malleable IDC.

2 Preliminaries

2.1 Notations

We write $y \leftarrow A(x)$ to denote that algorithm A takes x as input and outputs y, and $y \leftarrow_R A(x)$ means the algorithm A is a randomized algorithm. We write $y \leftarrow < A(x), B(z) > (w)$

to denote an interactive machine. A and B take w as public inputs and x and z as private input respectively.

2.2 Definitions

Definition 1 (Promise Problems[6])
A promise problem $\prod$ *is a pair of non-intersecting sets, denoted* $(\prod_{Yes}, \prod_{No})$*; that is,* $\prod_{Yes}, \prod_{No} \subseteq \{0,1\}^*$ *and* $\prod_{Yes} \cap \prod_{No} = \emptyset$*. The set* $\prod_{Yes} \cup \prod_{No}$ *is called the promise.*

Remark 1. In the following context, we only discuss "*hard promise problems*" (promise problem with "hard relations"[3]), for which the following conditions are satisfied.

(1) It is easy to recognize an instance in the promise: there is a polynomial time algorithm to decide if "$x \in \prod$" with high success probabilities.
(2) For each "$x \in \prod_{Yes}$", there is a witness w and a relation $R_{\prod_{Yes}}$ such that "$(x, w) \in R_{\prod_{Yes}}$" and there is a probabilistic polynomial time algorithm that on input 1^k outputs a pair $(x, w) \in R_{\prod_{Yes}}$. Additionally, given a pair (x, w), it is easy to establish whether $(x, w) \in R_{\prod_{Yes}}$. Moreover, for any $x \in \prod_{No}$, the probability of finding a witness w such that $(x, w) \in R_{\prod_{Yes}}$ is negligible.
(3) Given a instance $x \in \prod$, the probability of any probabilistic polynomial time machine outputting a witness w such that $(x, w) \in R_{\prod_{Yes}}$ is negligible.

Definition 2 (Instance-Dependent Commitment). *Let* $\prod = \prod_{Yes} \cup \prod_{No}$ *be a promise problem. An* instance-dependent commitment scheme *with the third trusted party is a protocol* $< \mathcal{T}, C, \mathcal{R} >$ *involves a third trusted party* $\mathcal{T}$*, a committer* C *and a receiver* $\mathcal{R}$*. Let "crs" denotes all the common inputs other than* $x \in \prod$*. "k" is the security parameter. The protocol has three parts:*

– *The setup phase.* $\mathcal{T}$ *generates all common inputs including crs and x and makes them public.*
– *The commit phase. If* C *and* $\mathcal{R}$ *follow the protocol, after the commit phase,* $\mathcal{R}$ *obtains a commitment* $com \leftarrow< C(m), \mathcal{R} > (crs, x)$*, which includes all the messages exchanged between C and R in this phase.*
– *The reveal phase. In this phase, C sends the message m being committed to, and all the randomness (denoted by dec) being used, to* $\mathcal{R}$*. The receiver outputs either "accept" or "reject" (which is denoted by "1" and "0" in the following context).*

The protocol satisfies hiding and binding properties on varying degrees according to $x \in \prod_{Yes}$ *or* $x \in \prod_{No}$*:*

– *Hiding property:*

$$Pr\begin{bmatrix} (crs, x) \leftarrow \mathcal{T}(1^k), where\ x \in \prod; \\ (m_0, m_1) \leftarrow_R \mathcal{M}(1^k); \\ com \leftarrow< C(m_b), \mathcal{R} > (crs, x); \\ b' \leftarrow \mathcal{D}(crs, x, com_b); \\ dec_b, m_b \leftarrow C(crs, x, com_b, m_b); \\ \text{accept} \leftarrow \mathcal{R}(crs, x, com_b, dec_b, m_b) : \\ b' = b \end{bmatrix} \leq 1/2 + neg(k)$$

where $\mathcal{M}$ is a random sampling algorithm on the message space, $\mathcal{D}$ is a distinguish algorithm and $b, b' \in \{0,1\}$.

– *Binding property:*

$$Pr\left[\begin{array}{l} (crs, x) \leftarrow \mathcal{T}(1^k), where\ x \in \prod; \\ com \leftarrow < C, \mathcal{R} > (crs, x); \\ \{(m, dec)(m', dec')\} \leftarrow C(crs, x, com) : \\ \text{accept} \leftarrow \mathcal{R}(prs, com, dec, m) \cap \text{accept} \leftarrow \mathcal{R}(prs, com, dec', m'), \\ m \neq m' \end{array}\right] \leq neg(k)$$

The commitment is an instance-dependent commitment scheme on the promise problem, if the following conditions are satisfied:

(1) While $x \in \prod_{Yes}$, for any probabilistic polynomial time algorithm C, and any computationally unbounded algorithms $\mathcal{R}$ and $\mathcal{D}$, the above properties hold.
(2) While $x \in \prod_{No}$, for any computationally unbounded algorithm C, and any probabilistic polynomial time algorithms $\mathcal{R}$ and $\mathcal{D}$, the above properties hold.

Note that there are some differences between our definition of IDC and previous definitions [10]: (1) We require computational binding and hiding for yes-instances and no-instances respectively, which are not required according to previous definitions. Since most of the existing IDCs already satisfy these requirements, this restriction does not make constructing an IDC less efficient, but makes the definition of instance-non-malleability possible. (2) Our definition considers both the interactive situation and non-interactive situation.

3 Instance-Non-malleable IDC

Before we give a formal definition for the instance-non-malleability of an IDC , we clarify the model of man-in-the-middle attacks for a protocol. The description of the model is mainly after [11].

Man-in-the-Middle Attack. During a man-in-the-middle attack, the adversary simultaneously participates in two executions, which are called the left and right interactions. In the left interaction, the adversary acts as a verifier and obtains messages from a real sender (which is the committer in a commitment scheme). In the right interaction, the adversary acts as a sender who composes messages that are related to the real sender's messages. An honest receiver would not be able to tell the difference between interactions with a real sender and with the adversary. For simplicity and without loss of generality, we only consider the cases of the protocol being fully executed when calculating the success probability.

Stand-Alone Execution. In this execution, the adversary $\mathcal{A}'$ only obtains the public inputs as in the man-in-the-middle execution. It then interacts with a real receiver until the protocol is fully executed.

For an IDC scheme $< \mathcal{T}, C, \mathcal{R} >$, in the man-in-the-middle execution, $\mathcal{T}$ generates an instance x, $\mathcal{A}$ receives x and outputs $x_{\mathcal{A}}$ to the receiver, and the left and right interactions then proceed. Let m denote the message being decommitted by the real committer in the left interaction, $m_{\mathcal{A}}$ denote the message being decommitted by the adversary $\mathcal{A}$ and z denote the auxiliary input of $\mathcal{A}$. Let $mim^{\mathcal{A}}_{\mathcal{R}}(R, x, m, z)$ denote the random variable describing the final outputs of $< \mathcal{A}, \mathcal{R} >$ in the man-in-the-middle execution. We have $mim^{\mathcal{A}}_{\mathcal{R}}(R, x, m, z) = 1$ if $\mathcal{A}$'s commitment in the right interaction is decommitted to $m_{\mathcal{A}}$ that an honest verifier accepts and there is a non-trivial binary relation R s.t. $R((x, m), (x_{\mathcal{A}}, m_{\mathcal{A}})) = 1$.

In the stand-alone execution, $\mathcal{T}$ generates an instance x, and a stand-alone adversary $\mathcal{A}'$ receives x and outputs x' to the receiver. m is chosen prior to the interactions between $\mathcal{A}'$ and the verifier but is only passed on to $\mathcal{A}'$ after the commitment phase. Let $sta^{\mathcal{A}'}_{\mathcal{R}}(R, x, m, z)$ denote the random variable describing the final outputs of $< \mathcal{A}, \mathcal{R} >$ in stand-alone execution. We have $sta^{\mathcal{A}'}_{\mathcal{R}}(R, x, m, z) = 1$ if $\mathcal{A}'$'s commitment is decommitted to $m_{\mathcal{A}'}$ that the honest verifier accepts, and for the same non-trivial binary relation R, we have $R((x, m), (x_{\mathcal{A}'}, m_{\mathcal{A}'})) = 1$.

Definition 3 (Instance-Non-malleable IDC). *Let $< \mathcal{T}, C, \mathcal{R} >$ be an instance-dependent commitment on a promise problem $\prod = \{\prod_{Yes}, \prod_{No}\}$ and k be the security parameter. We say that $< \mathcal{T}, C, \mathcal{R} >$ is instance-non-malleable if for any probabilistic polynomial time man-in-the-middle adversary $\mathcal{A}$, there exists a polynomial time stand-alone simulator $\mathcal{A}'$, such that for every non-reflexive polynomial-time computable relation R:*

$$Pr[mim^{\mathcal{A}}_{\mathcal{R}}(R, x, m, z) = 1] - Pr[sta^{\mathcal{A}'}_{\mathcal{R}}(R, x, m, z) = 1] \leq neg(k)$$

.

Theorem 1. *For an IDC, instance-non-malleability is stronger than non-malleability according to previous definitions.*[1]

4 Construction

In this section, we give a general construction of instance-non-malleable IDC from bit-IDCs.

We first briefly show the basic building blocks to construct our scheme. Let k be the security parameter. A collision-resistant one-way hash function $H(r) : \{0, 1\}^* \to \{0, 1\}^k$ is used in our construction. In addition, a hash function $G(x) : \{0, 1\}^* \to \{0, 1\}^{2k}$ which satisfies the pair-wise independent property is also used (Note that there are many well-known and very simple construction of pair-wise independent hash families [1]. Informally speaking, pair-wise independent requires that for $x \neq x'$, $G(x)$ and $G(x')$ are uniformly independent distributed). We also apply a common statistical binding commitment scheme **{com, dec}** (which is not instance-dependent) and a strong one-time signature scheme **{KeyGen,Sig,Ver}** in our construction. Finally, we construct our scheme with the non-interactive bit-IDC **{Com, Dec}**, which can be constructed from a variation of Σ-protocol for the hard promise problems[3].

[1] We omit all the theorem proofs in this paper due to the length limitation. The readers could contact the authors for a full version of this work including the proofs if interested.

The IDC scheme is described as following.

With an instance $x \in \prod$ as a previous input to both the committer and the verifier, and the description of the hash functions $H(\cdot) : \{0,1\}^* \to \{0,1\}^k$ and $G(\cdot) : \{0,1\}^* \to \{0,1\}^{2k}$ as the common reference; to commit to a message m, the committer generates its commitment as following.

1. Chooses random strings: $r; r_1, r_2, ...r_k \in \{0,1\}^*$;
2. Uses the algorithm **KeyGen** to generate the signature key and the verification key (SK, VK) for the strong one-time signature scheme.
3. Computes $H(m) = u_1u_2...u_k$ (where u_i represents the ith bit of $H(m)$) and calculates the bit-IDCs for each bits of $u_1u_2...u_k$: $\mathbf{Com}_{x,r_1}(u_1), \mathbf{Com}_{x,r_2}(u_1), ...,$ $\mathbf{Com}_{x,r_k}(u_k)$.
4. Computes $G(r \parallel VK) = v_1v_2, ..., v_{2k}$ (where v_i represents the ith bit of $G(r \parallel VK)$) and generates a commitment composed of $2k$ bit-IDCs $Com = Com_1 \circ Com_2 \circ Com_3...Com_{2k}$ as following:

   If the number of "0" is larger than that of "1" in the string "$v_1v_2, ..., v_{2k}$", assume the jth "0" is v_i $(j \leq k)$, we let $Com_i = \mathbf{Com}_{x,r_j}(u_j)$ for all of the j such that $1 \leq j \leq k$. For the rest of $v_i = 0$ and all of the $v_i = 1$ $(1 \leq i \leq 2k)$, the committer randomly choose $Com_i \in \mathcal{COM}$ as the redundant padding of Com, where $\mathcal{COM}$ is the range of the bit-IDC.

   Similarly, if the number of "0" is not larger than that of "1" in the string "$v_1v_2, ..., v_{2k}$", assume the jth "1" is v_i $(j \leq k)$, we let $Com_i = \mathbf{Com}_{x,r_j}(u_j)$ for all of the j such that $1 \leq j \leq k$. For all the $v_i = 0$ and the rest of $v_i = 1$ $(1 \leq i \leq 2k)$, the committer just randomly choose $Com_i \in \mathcal{COM}$.

   In addition, using the common commitment algorithm **com** to commit to the randomness r and let c denote the commitment.
5. The committer calculates the signature $\sigma = \mathbf{Sig}_{SK}(x, Com, c)$.

Afterwards, the committer sends (Com, VK, c, σ) to the verifier as its commitment. The verifier uses the algorithm **Ver** to check the validity of the signature.

To decommitment:

1. The committer reveals the message m, and the randomness $(r, r_1, r_2, ...r_k)$ to the verifier.
2. The verifier checks if the deccommitment of c is r. If true, computes $G(r \parallel VK)$ and picks the effective bit-IDCs out of Com accordingly, which are denoted as $s_1, s_2, ..., s_k$.
3. The verifier computes $H(m) = u_1u_2, ..., u_k$, and checks if $s_i = \mathbf{Com}_{x,r_i}(u_i)$ for all of the i such that $1 \leq i \leq k$.

If true, the decommitment is accepted; and if false, the verifier rejects.

Theorem 2. *The above construction is an IDC according to definition 2.*

Theorem 3. *The above construction is instance-non-malleable according to definition 3.*

Intuitively, the strong one-time signature scheme in our construction prevents the man-in-the-middle adversary from choosing its own instance and commitment which is different from the real committer's. Similar technique has been widely used in constructing non-malleable encryption schemes and zero-knowledge protocols[4,8]. What is new and interesting in our construction is that we choose the redundant padding in the same range with real bit-IDCs and introduce a randomness r to hide the value of the output of the hash function G, so that it is hard for a probabilistic polynomial time adversary to tell which ones of the bit-IDCs are meaningful (being the commitments of the bits of $H(m)$). Therefore, the man-in-the-middle adversary has no advantage in picking the meaningful commitments and making its own bit-IDCs related to them.

Just for clarification, the strong one-time signature is necessary here. Especially, just use $G(r \parallel x)$ cannot even achieve non-malleability in the standard model. Because in this situation, an adversary can choose $x_{\mathcal{A}} = x$, $c_{\mathcal{A}} = c$ and make each of its bit-IDCs related with the real committer's bit-IDCs in the same position. Since the effective bit-IDCs of the adversary's and the real committer's are on the same positions, in the opening phase, $h_{\mathcal{A}}$ and h can be correlated. Moreover, the hash function H does not provide non-malleability.

It is worth noting that the common commitment which is applied to hide r has to be statistical binding and computational hiding in order to achieve the statistical binding property of the construction.

5 Conclusions

In this paper, we explained the necessariness of instance-non-malleability for an IDC and proposed a formal definition. We also give a general construction for the non-interactive instance-non-malleable IDC scheme according to our definition.

We hope more usages of instance-non-malleable IDC can be found in constructing secure protocols and more efficient instance-non-malleable IDCs can be developed in the future.

References

1. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack, pp. 13–25. Springer (1998)
2. Di Crescenzo, G., Ishai, Y., Ostrovsky, R.: Non-interactive and non-malleable commitment, pp. 141–150. ACM (1998)
3. Damgard, I.: On Σ-protocols (2010), http://www.daimi.au.dk/ivan/Sigma.pdf
4. Dolev, D., Dwork, C., Naor, M.: Non-malleable cryptography, pp. 542–552. ACM (1991)
5. Fischlin, M.: Completely non-malleable schemes. Automata, Languages and Programming, 779–790 (2005)
6. Goldreich, O.: On promise problems (a survey in memory of Shimon Even (1935-2004)). ECCC, TR05-018 127, 128 (2005)
7. Ong, S., Vadhan, S.: An equivalence between zero knowledge and commitments. Theory of Cryptography, 482–500 (2008)
8. Sahai, A.: Non-Malleable Non-Interactive Zero Knowledge and Adaptive Chosen-Ciphertext Security. In: 40th FOCS, pp. 543–553 (1999)

9. Shoup, V.: Using Hash Functions as a Hedge against Chosen Ciphertext Attack. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 275–288. Springer, Heidelberg (2000)
10. Vadhan, S.: An unconditional study of computational zero knowledge. SIAM Journal on Computing 36, 1160–1214 (2007)
11. Pass, R., Rosen, A.: New and improved constructions of non-malleable cryptographic protocols. In: Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing, pp. 542–552 (2005)
12. Itoh, T., Ohta, Y., Shizuya, H.: A language-dependent cryptographic primitive. Journal of Cryptology 10, 37–49 (1997)
13. Kapron, B., Malka, L., Srinivasan, V.: A Characterization of Non-interactive Instance-Dependent Commitment-Schemes (NIC). In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 328–339. Springer, Heidelberg (2007)

Author Index