Colin Boyd
Leonie Simpson (Eds.)

LNCS 7959

# Information Security and Privacy

18th Australasian Conference, ACISP 2013
Brisbane, QLD, Australia, July 2013
Proceedings

Springer

Colin Boyd   Leonie Simpson (Eds.)

# Information Security and Privacy

18th Australasian Conference, ACISP 2013
Brisbane, QLD, Australia, July 1-3, 2013
Proceedings

Springer

Volume Editors

Colin Boyd
Leonie Simpson
Queensland University of Technology
GPO Box 2434, Brisbane, QLD 4001, Australia
E-mail: {c.boyd; lr.simpson}@qut.edu.au

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-39058-6 e-ISBN 978-3-642-39059-3
DOI 10.1007/978-3-642-39059-3
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2013940582

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

This volume contains the papers presented at ACISP 2013: the 18th Australasian Conference on Information Security and Privacy held during July 1–3, 2013, in Brisbane, Australia. The conference was hosted by the Institute for Future Environments at Queensland University of Technology, who provided first-class facilities and material support. The excellent local Organizing Committee was led by the ACISP 2013 General Chair, Ed Dawson, with administration led by Amanda Dunne. We made use of the EasyChair submission and reviewing software.

There were 78 submissions. Each submission was allocated to three Program Committee members and each paper received on the average 3.1 reviews. The committee decided to accept 28 papers. Accepted papers came from 11 countries, with the largest proportions coming from Japan (9), China (5), Australia (5), and Estonia (2). Other authors are from France, Hong Kong, India, Korea, Norway, UK, and USA. We would like to extend our sincere thanks to all authors who submitted papers to ACISP 2013.

The program also included four excellent and informative invited talks. Two of these were from eminent cryptography researchers, two were from highly experienced security practitioners. Paul Ashley (IBM Security Systems) talked on "Trends in Advanced Threat Protection"; Xavier Boyen (QUT) talked on "Expressive Cryptography: Lattice Perspectives"; Bradley Schatz (Schatz Forensic) talked on "Current and Future Challenges in Digital Forensics"; and Yuliang Zheng (UNC Charlotte) talked on "Public Key Cryptography for Mobile Cloud."

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

April 2013

Colin Boyd
Leonie Simpson

# Organization

## General Chair

| | |
|---|---|
| Ed Dawson | Queensland University of Technology, Australia |

## Program Chairs

| | |
|---|---|
| Colin Boyd | Queensland University of Technology, Australia |
| Leonie Simpson | Queensland University of Technology, Australia |

## Program Committee

| | |
|---|---|
| Michel Abdalla | École Normale Supérieure, France |
| Joon Sang Baek | Khalifa University, UAE |
| Paulo Barreto | University of São Paulo, Brazil |
| Lynn Batten | Deakin University, Australia |
| Colin Boyd | Queensland University of Technology, Australia |
| Serdar Boztas | RMIT University, Australia |
| Raymond Choo | University of South Australia |
| Orr Dunkelman | University of Haifa, Israel |
| Bao Feng | Institute for Infocomm Research, Singapore |
| Ernest Foo | Queensland University of Technology |
| Steven Galbraith | University of Auckland, New Zealand |
| Praveen Gauravaram | Tata Consultancy Services, India |
| Matt Henricksen | Institute for Infocomm Research, Singapore |
| Jiankun Hu | University of New South Wales at ADFA, Australia |
| Wang Huaxiong | Nanyang Technological University, Singapore |
| Keith Martin | Royal Holloway, University of London, UK |
| Atsuko Miyaji | Japan Advanced Institute of Science and Technology |
| Udaya Parampalli | University of Melbourne, Australia |
| Josef Pieprzyk | Macquarie University, Australia |
| Axel Poschmann | Nanyang Technological University, Singapore |
| Rei Safavi-Naini | University of Calgary, Canada |
| Palash Sarkar | Indian Statistical Institute |
| Leonie Simpson | Queensland University of Technology, Australia |
| Douglas Stebila | Queensland University of Technology, Australia |
| Ron Steinfeld | Monash University, Australia |
| Willy Susilo | University of Wollongong, Australia |

| | |
|---|---|
| Tsuyoshi Takagi | Kyushu University, Japan |
| Vijay Varadharajan | Macquarie University, Australia |
| Kapaleeswaran Viswanathan | HP Labs, India |
| Guilin Wang | University of Wollongong, Australia |
| Duncan Wong | City University of Hong Kong, SAR China |
| Chuankun Wu | Chinese Academy of Sciences |
| Kan Yasuda | NTT Corporation, Japan |

## Additional Reviewers

Abdelraheem, Mohamed Ahmed
Aono, Yoshinori
Aranha, Diego
Asghar, Hassan
Au, Man Ho
Barua, Rana
Brzuska, Christina
Chalamala, Srinivasa Rao
Chang, Donghoon
Chen, Chien-Ning
Chen, Jiageng
Chen, Jie
Cho, Joo Yeon
Chow, Sherman
Dela Cruz, Romar
Deng, Yi
Futa, Yuichi
Gao, Wei
Ghodosi, Hossein
Granger, Robert
Hinek, Jason
Hirose, Shoichi
Huang, Qiong
Huang, Xinyi
Jiang, Shaoquan
Knudsen, Lars
Kohlweiss, Markulf
Kojima, Tetsuya
Kucuk, Ozgul
Kunihiro, Noboru
Laverdière, Marc-André
Lee, Peter
Liang, Kaitai
Lim, Hoon Wei
Liu, Zhen
Lu, Yao
Lyubashevsky, Vadim
Martini, Ben
May, Alexander
McDonald, Cameron
Moonsamy, Veelasha
Morozov, Kirill
Nguyen, Khoa
Nguyen, Phuong Ha
Omote, Kazumasa
Parno, Bryan
Paterson, Maura
Paul, Goutam
Pereira, Geovandro
Persichetti, Edoardo
Phuong Ha, Nguyen
Rangasamy, Jothi
Rodriguez-Henriquez, Francisco
Ruan, Chun
Saarinen, Markku-Juhani
Salim, Farzad
Sanadhya, Somitra
Sasaki, Yu
Sato, Hisayoshi
Stehlé, Damien
Sun, Donald
Teo, Sui Guan
Todo, Yosuke
Tupakula, Udaya
Wang, Pengwei
Wang, Yilei
Warinschi, Bogdan
Weng, Jian
Xu, Haixia
Xu, Jing
Yang, Guomin
Yoneyama, Kazuki
Zhang, Hui
Zhang, Liangfeng
Zhang, Rui
Zhang, Wei
Zhang, Yun
Zhang, Zongyang

# Table of Contents

# Analysing the IOBC Authenticated Encryption Mode

Chris J. Mitchell

Information Security Group, Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK
c.mitchell@rhul.ac.uk

**Abstract.** The idea of combining a very simple form of added plaintext redundancy with a special mode of data encryption to provide data integrity is an old one; however, despite its wide deployment in protocols such as Kerberos, it has largely been superseded by provably secure authenticated encryption techniques. In this paper we cryptanalyse a block cipher mode of operation called IOBC, possibly the only remaining encryption mode designed for such use that has not previously been analyzed. We show that IOBC is subject to known-plaintext-based forgery attacks with a complexity of around $2^{n/3}$, where $n$ is the block cipher block length.

## 1 Introduction

This is perhaps the last chapter in a long and rather unfortunate story[1], namely that of 'special' modes of operation for block ciphers, designed to offer 'low cost' combined integrity and confidentiality protection by combining encryption with the addition of simple (or fixed) redundancy to the plaintext. The underlying idea is to design the mode so that modifying the ciphertext without invalidating the added redundancy is impossible without knowledge of the encryption key. It is a long story since the idea dates back over 30 years, and a sad story because one by one these special modes of operation have been shown to fail to meet their objectives.

Such modes are the theme of section 9.6.5 of Menezes, van Oorschot and Vanstone's landmark book [1]. As they point out, the starting point for the development of such modes is the observation that encryption alone does not guarantee integrity. This, combined with the observation that the 'obvious' approach of encrypting the data and then separately computing a MAC involves twice the work, leads to the alternative notion of adding detectable redundancy before encrypting so that it can be detected after decryption. Two main methods for adding redundancy have been proposed:

– add a fixed block at the end of the plaintext, which may be public or secret (in the latter case the block acts as an auxiliary key);

[1] Since ACISP has played its own part in this developing tale, it seems fitting to present this final chapter at ACISP 2013.

– append to the plaintext some easily computed and simple (public) function of the plaintext.

In either case we refer to the block added to the end of the plaintext as a *Manipulation Detection Code (MDC)*. Whichever approach is adopted, the method for computing the MDC needs to be simple, or it offers no advantage over the more conventional ‘encrypt then MAC’ approach.

Note that there is a third, related, approach which remains viable and is, indeed, increasingly used; this involves computing a keyed function of the plaintext (a sort of ‘weak MAC’) which is then encrypted to make it secure. The plaintext itself may or may not be encrypted. Indeed, one example of such an approach, namely GCM/GMAC [2], has a proof of security and has been standardized [3,4]. The main differences between GCM (and other related techniques) and the approaches we are concerned with here is that GCM uses a ‘standard’ encryption method and, of course, GCM has a proof of security.

At this point we observe that the general approaches described above possess an intrinsic weakness if the method of adding redundancy is public[2]. Suppose an attacker persuades the legitimate originator of protected messages to encrypt a message containing a correct MDC somewhere in the middle (where the MDC is correct in the sense that it is a correct MDC on the data that precedes it). The attacker will then be able to delete all ciphertext blocks following the encrypted MDC, and such a change will not be detectable. Despite this weakness, using an appropriate encryption mode combined with a public method for adding verifiable redundancy to a message has been widely used for message integrity protection — e.g. in Kerberos (see, for example, [5]). We thus restrict our attention in the remainder of the paper to the case where the MDC is a secret value (this is the approach proposed for use with the IOBC mode, which forms the main subject of this paper).

Regardless of the nature of the MDC, the method of encryption needs to be chosen with great care. Using a stream cipher or CBC mode encryption is clearly totally insecure, since a ciphertext produced with such a technique can readily be manipulated in such a way that large parts of the recovered plaintext, including the end of the plaintext, remains unchanged. A simple modified version of CBC called Plaintext Cipher Block Chaining (PCBC) [6,7], in which the feedback chaining variable is the exclusive-or of the immediately preceding ciphertext and plaintext blocks, was proposed back in the 1980s to avoid this problem. This scheme was used in Kerberos version 4 [7] to provide combined encryption and integrity protection. The weakness of PCBC for use as an integrity-protection mode was first pointed out by Kohl [6]. As is simple to verify, Kohl showed that interchanging two of the ciphertext blocks of a PCBC-encrypted message does not affect the decryption of the final block, i.e. it is extremely simple to make undetectable changes to messages. Note that this is actually a stronger attack than is implied by Menezes, van Oorschot and Vanstone [1] who refer only to the danger of known-plaintext attacks.

[2] This problem appears to be part of the cryptographic folklore — it was pointed out to the author by Bart Preneel in the late 1990s.

Menezes, van Oorschot and Vanstone [1] describe a slightly different mode, confusingly also referred to as PCBC (this time for *Plaintext-Ciphertext Block Chaining*), in which the feedback chaining variable is the modulo $2^n$ sum of the immediately preceding ciphertext and plaintext blocks. This technique was also described as long ago as 1982 by Meyer and Matyas [8]. Unfortunately, despite its long history, the latter version of PCBC was shown to possess major weaknesses when used with an MDC for integrity protection in a 2005 ACISP paper [9]. M-PCBC, a further variant of PCBC due to Sierra et al. [10], was also shown to fail to offer appropriate MDC protection in the same ACISP 2005 paper [9]. Another variant of PCBC was proposed by Gligor and Donescu [11]; however, this latter scheme, known as iaPCBC, was shown to possess serious vulnerabilities by Ferguson et al. [12]. Yet another scheme, called PES-PCBC, was proposed by Zuquete and Guedes [13] but, as observed by Recacha [14] as well the authors themselves [15], PES-PCBC is subject to known-plaintext attacks.

Indeed, to the author's knowledge, almost all the proposals for such special modes have been cryptanalyzed, with one exception — a scheme proposed in 1996 called *Input and Output Block Chaining (IOBC)* [14]. One possible reason why IOBC has escaped attention is that until recently the only available description was in Spanish. However, recently an English language translation of the original 1996 paper[3] has been made available by the author, and it is this translation that has been used as the basis of this paper. It is interesting to note that IOBC has, nevertheless, had an impact on the cryptographic literature. A modified version of IOBC, called EPBC, was published in 1997 [15] and was subsequently cryptanalyzed in 2007 [16].

Finally we observe that, although the history of the area sketched above may give the impression of an ordered and coherent narrative, the truth is somewhat different. Ideas, and sometimes attacks, appear to have been put forward independently of one another, and one role of this introduction is to try to pull together the main developments in the area.

The remainder of the paper is structured as follows. Section 2 provides details of the operation of IOBC and its use for integrity protection. This is followed in section 3 by certain observations on the properties of IOBC and its component functions. This sets the stage for section 4, in which a known plaintext forgery attack against IOBC when used for integrity protection is described. A brief discussion of possible fixes to IOBC is given in section 5, before a certificational chosen-plaintext attack on IOBC is outlined in section 6. Concluding remarks are provided in section 7.

## 2 The Recacha IOBC Mode

We next describe the operation of the IOBC block cipher mode of operation[4]. We base our description on Recacha's 1996 paper [14], although we use the notation of Mitchell [16].

[3] `http://inputoutputblockchaining.blogspot.com.es/`

[4] PES-PCBC (briefly mentioned in section 1) is the same as IOBC with the exception that in PES-PCBC the function $g$ is the identity function.

## 2.1 Initial Assumptions

We suppose that IOBC is to be used with an $n$-bit block cipher, i.e. a block cipher operating on plaintext and ciphertext blocks of $n$ bits. We further suppose that $n$ is a power of two, and put $n = 2^m$ (as is the case for all standardized block ciphers [17]). We write $e_K(P)$ for the result of block cipher encrypting the $n$-bit block $P$ using the secret key $K$, and $d_K(C)$ for the result of block cipher decrypting the $n$-bit block $C$ using the key $K$. Finally we suppose the plaintext to be protected is divided into a sequence of $n$-bit blocks (if necessary, having first been padded): $P_1, P_2, \ldots, P_t$.

## 2.2 Initialization Vectors

The scheme uses two secret $n$-bit Initialization Vectors (IVs), denoted by $F_0$ and $G_0$. The nature of the intended restrictions on their use is not altogether clear. However, one suggestion in the original Recacha paper [14] is that they should be generated as follows.

Suppose $K'$ is an auxiliary key used solely for generating the IVs. Suppose also that $S$ is a sequence number, managed so that different values are used for every message. Then $F_0 = e_{K'}(S)$ and $G_0 = e_{K'}(F_0)$. For the purposes of this paper we assume that $F_0$ and $G_0$ are always generated this way, and thus the scheme can be thought of as employing a pair of block cipher keys and a non-secret, non-repeating, sequence number (which must be carefully managed to prevent accidental re-use of sequence number values). Note that special measures will need to be taken if the same key is to be used to encrypt communications in both directions between a pair of parties. Avoiding sequence number re-use in such a case could be achieved by requiring one party to start the sequence number they use for encryption at a large value, perhaps halfway through the range.

## 2.3 Operation

The IOBC encryption of the plaintext $P_1, P_2, \ldots, P_t$ operates as follows:

$$\begin{aligned} G_i &= P_i \oplus F_{i-1}, \quad (1 \leq i \leq t), \\ F_i &= e_K(G_i), \quad (1 \leq i \leq t), \\ C_i &= F_i \oplus g(G_{i-1}), \quad (2 \leq i \leq t), \end{aligned}$$

where $C_1 = F_1 \oplus G_0$, $\oplus$ denotes bit-wise exclusive-or, and $g$ is a function that maps an $n$-bit block to an $n$-bit block, defined below. The operation of the mode (when used for encryption) is shown in Figure 1. Note that we refer to the values $F_i$ and $G_i$ as 'internal' values, as they are computed during encryption, but they do not constitute part of the ciphertext.

The function $g$ is defined as follows. Suppose $X$ is an $n$-bit block, where $n = 2^m$. Suppose also that $X = L||R$ where $L$ is the leftmost $2^{m-1} - 1$ bits of $X$ and $R$ is the rightmost $2^{m-1} + 1$ bits of $X$ (and, as throughout, $||$ denotes concatenation). Then

$$g(X) = (>_1 (L))||(>_1 (R))$$

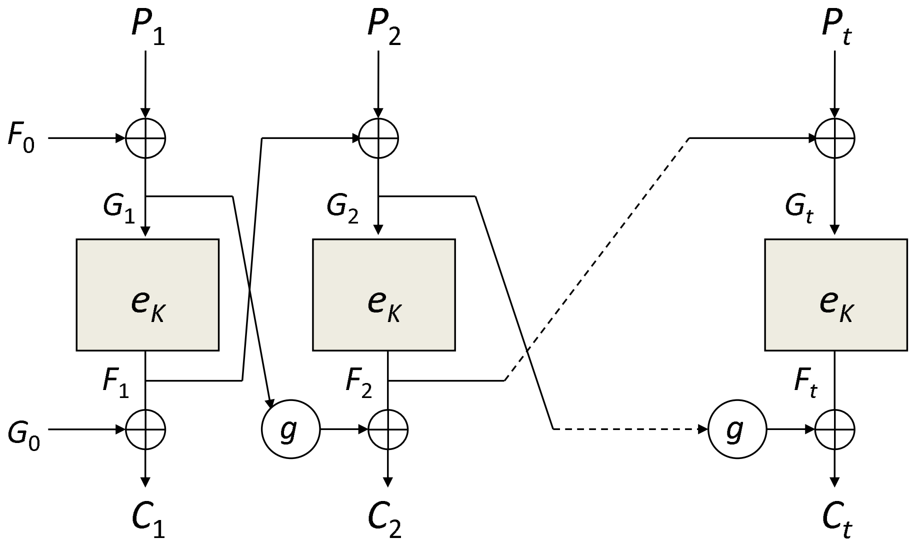


**Fig. 1.** IOBC encryption

where $>_i$ denotes a rightwards (cyclic) shift by $i$ bit positions.

Decryption operates similarly. We have:

$$F_i = C_i \oplus g(G_{i-1}), \quad (2 \leq i \leq t),$$
$$G_i = d_K(F_i), \quad (1 \leq i \leq t),$$
$$P_i = G_i \oplus F_{i-1}, \quad (1 \leq i \leq t).$$

and $F_1 = C_1 \oplus G_0$, where $d$ denotes block cipher decryption.

## 2.4 Additional Remarks

As described above, we assume throughout that the IVs $F_0$ and $G_0$ are derived by ECB-mode-encrypting a sequence number using a secondary key. Thus the ciphertext blocks will be a function of this serial number (as well as the pair of keys used). We thus write $[S], C_1, C_2, \ldots, C_t$ for a sequence of ciphertext blocks, meaning that $C_1, C_2, \ldots, C_t$ were encrypted using the sequence number $S$. This is logical, since the sequence number will need to be sent or stored with the ciphertext to enable correct decryption.

Finally observe that IOBC should only be used with relatively short messages. That is, as specified by Recacha [14] (and for reasons which become clear below), a message to be encrypted using IOBC shall contain at most $n^2/2 - 1$ plaintext blocks, where $n$ is the plaintext block length. Thus for $n = 64$ and $n = 128$, the two most commonly used block lengths, a message shall contain at most 2047 and 8191 blocks, respectively.

### 2.5 Using IOBC for Integrity Protection

As already implied, IOBC is designed for combined confidentiality and integrity protection. Confidentiality comes simply by encrypting the data using IOBC mode. Integrity is achieved by adding an MDC to the end of the plaintext — what Recacha [14] refers to as an *Integrity Check Vector (ICV)*. After decryption of an IOBC-protected message, the receiver must check that the ICV is correct, and must reject the message if it is not.

Recacha recommends use of a secret ICV of length $n/2$. This ICV must clearly be known to the intended recipient, and should therefore be regarded as forming part of the key (along with the key $K$ used in IOBC computations and the key $K'$ used to derive the IVs).

## 3 Preliminary Observations

We first establish some simple results on the operation of the IOBC scheme. The first Lemma derives directly from a discussion in section 6 of [16]. It is also implicit in the discussions of Recacha [14].

**Lemma 1.** *Suppose* $[S], C_1, C_2, \ldots, C_t$ *and* $[S'], C'_1, C'_2, \ldots, C'_{t'}$ *are IOBC encrypted versions of the plaintext sequences* $P_1, P_2, \ldots, P_t$ *and* $P'_1, P'_2, \ldots, P'_{t'}$*, respectively. If the ciphertext:*

$$[S'], C^*_1, C^*_2, \ldots, C^*_{t-v+u} =$$
$$[S'], C'_1, C'_2, \ldots, C'_{u-1}, C_v \oplus g(G'_{u-1}) \oplus g(G_{v-1}), C_{v+1}, \ldots, C_t$$

*is submitted for IOBC decryption (where* $1 < u$ *and* $1 < v < t$*, and* $G_{v-1}$ *and* $G'_{u-1}$ *are values computed during the encryption of the respective sequences of blocks), then the resulting sequence of plaintext blocks* $P^*_1, P^*_2, \ldots, P^*_{t-v+u}$ *will be equal to*

$$P'_1, P'_2, \ldots, P'_{u-1}, P_v \oplus F'_{u-1} \oplus F_{v-1}, P_{v+1}, P_{v+2}, \ldots, P_t.$$

*Proof.* We first note that it follows immediately from the definitions that $F^*_i = F'_i$ and $G^*_i = G'_i$ $(1 \leq i \leq u-1)$, where $F'_i$ and $G'_i$ are the internal values generated during the encryption process that yielded the ciphertext message $C'_1, C'_2, \ldots, C'_{t'}$. Hence $P^*_i = P'_i$ $(1 \leq i \leq u-1)$.
We now consider the decryption of $C^*_u$. We have:

$$\begin{aligned} F^*_u &= C^*_u \oplus g(G^*_{u-1}) \quad \text{(from section 2.3)} \\ &= C_v \oplus g(G'_{u-1}) \oplus g(G_{v-1}) \oplus g(G^*_{u-1}) \quad \text{(by defn. of } C^*_u\text{)} \\ &= C_v \oplus g(G_{v-1}) \quad \text{(since } G^*_{u-1} = G'_{u-1}\text{)} \\ &= F_v \quad \text{(from section 2.3).} \end{aligned}$$

Hence $G^*_u = G_v$. Finally we have:

$$\begin{aligned} P^*_u &= G^*_u \oplus F^*_{u-1} \quad \text{(from section 2.3)} \\ &= G_v \oplus F'_{u-1} \quad \text{(from above)} \\ &= P_v \oplus F'_{u-1} \oplus F_{v-1} \quad \text{(from section 2.3).} \end{aligned}$$

We now consider the decryption of $C^*_{u+1}$. We have

$$\begin{aligned} F^*_{u+1} &= C^*_{u+1} \oplus g(G^*_u) \quad \text{(from section 2.3)} \\ &= C_{v+1} \oplus g(G_v) \quad \text{(by defn. of } C^*_{u+1} \text{ and from above)} \\ &= F_{v+1} \quad \text{(from section 2.3).} \end{aligned}$$

Hence $G^*_{u+1} = G_{v+1}$. Finally, we have

$$\begin{aligned} P^*_{u+1} &= G^*_{u+1} \oplus F^*_u \quad \text{(from section 2.3)} \\ &= G_{v+1} \oplus F_v \quad \text{(from above)} \\ &= P_{v+1} \quad \text{(from section 2.3).} \end{aligned}$$

The same argument shows that $P^*_{u+i} = P_{v+i}$ for every $i \geq 1$, and the desired result follows. □

*Remark 1.* Lemma 1 suggests a way in which it may be possible to forge an IOBC-encrypted message so that the final block will contain the correct ICV. However, the problem remains of discovering $g(G'_{u-1}) \oplus g(G_{v-1})$ (as used in constructing the message in the statement of the lemma). Recacha [14] discusses this very point, and explains that making this difficult motivates the inclusion of the function $g$ in the design of IOBC — that is, if $g$ was not included (as is the case for PES-PCBC), then simple forgeries could be achieved by manipulating a single encrypted message for which part of the plaintext was known. We revisit this point later, and show that $g$ is not as effective in achieving the goal as intended.

We next give some elementary observations on the operation of IOBC.

**Lemma 2.** *Suppose* $[S], C_1, C_2, \ldots, C_t$ *is the encryption of* $P_1, P_2, \ldots, P_t$ *using IOBC, and that* $F_i$ *and* $G_i$ *are as defined in section 2.3. Then:*

*(i)* $C_{j+1} \oplus P_{j+2} = g(G_j) + G_{j+2}$, $1 \leq j \leq t-2$;

*(ii)* $\bigoplus_{i=1}^{k} g^{k-i}(C_{j+2i-1} \oplus P_{j+2i}) = g^k(G_j) \oplus G_{j+2k}$, $1 \leq j \leq t-2$, $1 \leq k \leq (t-j)/2$.

*Proof.* (i) follows immediately from the definition of the operation of IOBC. (ii) follows by inductively applying (i), observing that $g$ is a bit permutation, and hence a linear function, and so it distributes across the bitwise exclusive-or operation ($\oplus$). □

*Remark 2.* It is not hard to see that if $g^k(G_j) = G_j$ for some $k$, then Lemma 2(ii) could be combined with Lemma 1 to yield a forgery attack (given a ciphertext message with corresponding known plaintext). This point is made by Recacha [14], who explains that the bit permutation $g$ has been chosen so that the smallest integer $i > 1$ such that $g^i$ is the identity permutation is $n^2/4-1$. The restriction on the maximum length of messages that can be encrypted using IOBC, as defined in section 2.4, prevents this problem arising in practice. However, as we show next, in some cases $g^k$ is 'close' to the identity permutation for significantly smaller values of $k$.

We conclude this section by giving certain properties of the function $g$. We examine two special cases of particular practical importance, i.e. where $n$ is either 64 or 128. We first consider the case $n = 64$.

**Lemma 3.** *If $X$ is a randomly selected 64-bit block then:*

$$\Pr(X = g^{341}(X)) = 2^{-22}.$$

*Proof.* As in section 2.3, put $X = L_X||R_X$, where $L_X$ and $R_X$ are 31-bit and 33-bit blocks, respectively. Let $Y = g^{341}(X)$, and, analogously, let $Y = L_Y||R_Y$.

We first observe that $L_X = L_Y$. This follows immediately from the definition of $g$ and the observation that $341 = 31 \times 11$, i.e. it is a multiple of 31.

Secondly, we show that $\Pr(R_X = R_Y) = 2^{-22}$. To establish this, first observe that $341 = 10 \times 33 + 11$, i.e. $R_Y =>_{11} (R_X)$. Since $33 = 3 \times 11$, it follows that $R_Y = R_X$ if and only if $R_X = Z||Z||Z$, where $Z$ is an arbitrary 11-bit string. There are clearly $2^{11}$ such strings $Z$, and hence the probability that $R_Y = R_X$ is $2^{11}/2^{33}$, and the claim follows. This establishes the desired result. $\square$

An analogous result holds for $n = 128$, as follows.

**Lemma 4.** *If $X$ is a randomly selected 128-bit block then:*

$$\Pr(X = g^{1365}(X)) = 2^{-42}.$$

*Proof.* As previously, let $X = L_X||R_X$, where $L_X$ and $R_X$ are 63-bit and 65-bit blocks, respectively. Put $Y = g^{1365}(X)$, and define $L_Y$ and $R_Y$ analogously to the proof of the previous lemma.

Since $1365 = 21 \times 65$ it follows that $R_Y = R_X$. Also, since $1365 = 21 \times 63 + 42$, and since $21|42$ and $21|63$, we have $L_Y = L_X$ if and only if $L_X = Z||Z||Z$, where $Z$ is an arbitrary 21-bit string. The result now follows. $\square$

*Remark 3.* Similar results can be achieved for any $n = 2^m$ since, for every $m$, either $2^m - 1$ or $2^m + 1$ is a multiple of 3.

## 4 A Known-Plaintext Forgery Attack on IOBC

The main elements of the attack are now in place. We suppose that the attacker has access to a number of ciphertext messages all encrypted using the same key, and that the attacker also knows large parts of the plaintext for these messages. The precise number of messages required for the attack will depend on the message lengths and the value of $n$. We look at two special cases of particular importance.

### 4.1 The Case $n = 64$

We start by considering the case $n = 64$, as applies for standardized block ciphers such as 3DES, MISTY1 and CAST-128 [17]. In this case the definition of IOBC requires that messages encrypted using IOBC contain at most 2047 blocks.

Suppose the attacker has obtained a ciphertext message $[S], C_1, C_2, \ldots, C_t$ where $t \geq 685$. Suppose also that the attacker knows the corresponding plaintext blocks $P_1, P_2, \ldots, P_t$ (in fact, the attack we describe does not require the attacker to know all the plaintext blocks, as will become clear). Using Lemma 2(ii) with $j = 1$ and $k = 341$, the attacker can use knowledge of $C_2, C_4, \ldots, C_{682}$ and $P_3, P_5, \ldots, P_{683}$ to compute $g^{341}(G_1) \oplus G_{683}$.

The attacker now constructs a new ciphertext message $[S], C_1^*, C_2^*, \ldots, C_{t-682}^*$ equal to the following sequence of blocks:

$$[S], C_1, C_{684} \oplus g^{342}(G_1) \oplus g(G_{683}), C_{685}, \ldots, C_t.$$

Note that $g^{342}(G_1) \oplus g(G_{683})$ can be obtained simply by applying $g$ to $g^{341}(G_1) \oplus G_{683}$.

By Lemma 3, the probability that $g^{342}(G_1) \oplus g(G_{683}) = g(G_1) \oplus g(G_{683})$ is $2^{-22}$, assuming that the encryption algorithm generates randomly distributed ciphertext blocks. If this event occurs, then, by Lemma 1, the result of IOBC decrypting $[S], C_1^*, C_2^*, \ldots, C_{t-682}^*$ will be equal to:

$$P_1, P_{684} \oplus F_1 \oplus F_{683}, P_{685}, P_{686}, \ldots, P_t.$$

That is, since $t \geq 685$, the final plaintext block will contain the correct ICV, i.e. $[S], C_1^*, C_2^*, \ldots, C_{t-682}^*$ will be a successful forgery.

The above attack, which essentially involves cutting out 682 consecutive ciphertext blocks from a valid message and modifying the ciphertext block immediately after the removed portion, will yield a successful forgery with probability $2^{-22}$. In the example above, the removed ciphertext blocks were $C_2, C_3, \ldots, C_{683}$, but essentially the same attack will work by removing any sequence of 682 consecutive ciphertext blocks as long as it does not include the first block or the final two blocks. Thus, for example, a message containing 1808 blocks (well short of the maximum of 2047) could be used to construct $1024 = 2^{10}$ different possible forgeries, each of which would have a probability of $2^{-22}$ of being accepted as legitimate. A simple argument shows that 1000–2000 encrypted messages of this length could therefore yield $2^{21}$–$2^{22}$ forgeries, at least one of which is likely to be accepted.

We therefore conclude that the IOBC integrity protection mechanism can, in this case, be defeated with potentially as few as 1000–2000 known plaintexts and $2^{21}$–$2^{22}$ queries to a decrypting party.

## 4.2 The Case $n = 128$

We next consider the case $n = 128$, as applies for standardized block ciphers such as AES, Camellia and SEED [17]. In this case the definition of IOBC requires that messages encrypted using IOBC contain at most 8191 blocks.

An exactly analogous approach will clearly work here as for the $n = 64$ case, except that in this case we need to omit 2730 consecutive ciphertext blocks from a valid message, and make appropriate modifications to the ciphertext block immediately following the omitted sequence. In this case, the probability of the forged message being accepted will be $2^{-42}$ (from Lemma 4). As in the 64-bit case, a single message could yield a number of possible forgeries. For example, a 6829-block message could be used to generate $2^{12}$ different possible forgeries. A total of $2^{41}$–$2^{42}$ forgery attempts will be required to have a good chance of having at least one forgery accepted, potentially requiring $2^{29}$–$2^{30}$ known plaintexts (if they have an average length of around 7000 blocks).

This is a rather large number, but significantly less than the $2^{64}$ which is the design goal.

### 4.3 Other Values of $n$

The same general approach will work for any value of $n$ (see Remark 3), yielding a known-plaintext-based forgery attack with complexity approximately $2^{n/3}$ decryptions and somewhat less than $2^{n/3}$ known plaintexts.

## 5 Can IOBC be Fixed?

It is not hard to see that the attacks in section 4 could be prevented by further limiting the maximum length of message that can be encrypted using IOBC mode. However, unless the limit is made very small, less effective versions of the attack described in section 4 will still apply, where the exact results will depend on the factorisation of $2^m - 1$ and $2^m + 1$.

For example, for the $n = 64$ case (i.e. $m = 6$) we know that, for randomly chosen blocks $(X, Y)$ of 33 and 31 bits respectively, $\Pr(g^{93}(X) = X) = 2^{-30}$ and $g^{93}(Y) = Y$. That is a forgery attack with a success probability of $2^{-30}$ could be launched using a ciphertext (with known plaintext) of length only 100 blocks.

Of course, it may be possible to devise significantly more secure schemes by choosing $g$ to be more complex, but this would reduce the attractiveness of the scheme. After all, the only reason to adopt this approach instead of ‘encrypt then MAC’ (which is provably secure) is to reduce the complexity of protecting the message to that of encryption plus a small delta.

## 6 A Chosen-Plaintext Forgery Attack

All the attacks we have considered so far can be avoided if only relatively short messages are encrypted. Moreover, these attacks take advantage of special properties of the function $g$. As a result, it is of at least theoretical interest to know the level of security provided by IOBC mode regardless of the length of plaintext messages and of the choice of $g$.

We thus conclude the main part of the paper by sketching a certificational chosen-plaintext-based forgery attack which serves to limit the security of IOBC regardless of length limits for plaintexts (and the choice of $g$). Suppose that $[S], C_1, C_2, \ldots, C_t$ and $[S'], C'_1, C'_2, \ldots, C'_{t'}$ are IOBC encrypted versions of the plaintext sequences $P_1, P_2, \ldots, P_t$ and $P'_1, P'_2, \ldots, P'_{t'}$, respectively. Suppose also that $P'_i = P_j$ and $P'_{i+1} = P_{j+1}$.

It is not hard to see that if $C'_i = C_j$ and $C'_{i+1} = C_{j+1}$ then, with very high probability, we have $F'_{i-1} = F_{j-1}$, and hence $G'_{i-1} = G_{j-1}$ and $G'_{i+1} = G_{j+1}$. If such an event occurs, then, by Lemma 1, the constructed ciphertext message $[S'], C'_1, C'_2, \ldots, C'_{i-1}, C_j, C_{j+1}, \ldots, C_t$ will very conveniently decrypt to $P'_1, P'_2, \ldots, P'_{i-1}, P_j, P_{j+1}, \ldots, P_t$, i.e. a MAC forgery has been constructed. By the usual 'birthday paradox' probabilistic arguments, to find such an event simply requires around $2^{n/2}$ chosen plaintexts to be encrypted, each containing the same consecutive pair of plaintext blocks. In fact, the number of required chosen plaintext encryptions can be reduced to significantly less than $2^{n/2}$ by including many occurrences of the fixed pair of plaintext blocks in each chosen plaintext.

That is, regardless of the lengths of plaintext messages and the choice of $g$, forgery attacks on IOBC are possible if of the order of $2^{n/2}$ messages are encrypted using the same key.

## 7 Summary and Conclusions

The analysis in this paper suggests that IOBC does not offer an adequate level of security for routine use as the basis of a combined integrity and confidentiality technique. In fact, use of the 'add redundancy and then encrypt using a special mode' approach to provide combined integrity and confidentiality protection is no longer 'state of the art', and so this is arguably not a major development. The main significance is that, as mentioned in section 1, IOBC was the only remaining proposed block cipher mode for simultaneous confidentiality and integrity protection known to the author which had not already been shown to suffer from forgery attack issues. Hence this paper serves to bring a cryptographic chapter to a tidy close.

As discussed in many other places, if both confidentiality and integrity protection are required, then either encryption and a MAC should be combined in an appropriate way, or a dedicated 'authenticated encryption' mode should be used — see, for example, ISO/IEC 19772 [3]. Indeed, a wide variety of provably secure schemes are available.

**Acknowledgements.** The author would like to thank Francisco Recacha for his very helpful explanations and corrections, and also the anonymous referees for comments which have helped to improve the presentation of this paper.

## References

1. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1997)
2. McGrew, D.A., Viega, J.: The Galois/Counter mode of operation (GCM) (May 2005), `http://www.mindspring.com/~dmcgrew/gcm-nist-6.pdf`
3. International Organization for Standardization Genève, Switzerland: ISO/IEC 19772:2009, Information technology — Security techniques — Authenticated encryption mechanisms (February 2009)

4. International Organization for Standardization Genève, Switzerland: ISO/IEC 9797-3:2011, Information technology — Security techniques — Message Authentication Codes (MACs) — Part 3: Mechanisms using a universal hash-function (2011)
5. Dent, A.W., Mitchell, C.J.: User's Guide to Cryptography and Standards. Artech House (2005)
6. Kohl, J.T.: The use of encryption in kerberos for network authentication. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 35–43. Springer, Heidelberg (1990)
7. Steiner, J., Neuman, C., Schiller, J.: Kerberos: an authentication service for open network systems. In: Proceedings: Usenix Association, Winter Conference, Dallas 1988, pp. 191–202. USENIX Association, Berkeley (1988)
8. Meyer, C.H., Matyas, S.M.: Cryptography: A new dimension in computer data security. John Wiley and Sons, New York (1982)
9. Mitchell, C.J.: Cryptanalysis of two variants of PCBC mode when used for message integrity. In: Boyd, C., González Nieto, J.M. (eds.) ACISP 2005. LNCS, vol. 3574, pp. 560–571. Springer, Heidelberg (2005)
10. Sierra, J.M., Hernandez, J.C., Jayaram, N., Ribagorda, A.: Low computational cost integrity for block ciphers. Future Generation Computer Systems 20, 857–863 (2004)
11. Gligor, V.D., Donescu, P.: Integrity-aware PCBC encryption schemes. In: Malcolm, J.A., Christianson, B., Crispo, B., Roe, M. (eds.) Security Protocols. LNCS, vol. 1796, pp. 153–171. Springer, Heidelberg (2000)
12. Ferguson, N., Whiting, D., Kelsey, J., Wagner, D.: Critical weaknesses of iaPCBC (November 1999)
13. Zuquete, A., Guedes, P.: Transparent authentication and confidentiality for stream sockets. IEEE Micro 16(3), 34–41 (1996)
14. Recacha, F.: IOBC: Un nuevo modo de encadenamiento para cifrado en bloque. In: Proceedings: IV Reunion Espanola de Criptologia, Valladolid, pp. 85–92 (September 1996)
15. Zuquete, A., Guedes, P.: Efficient error-propagating block chaining. In: Darnell, M. (ed.) Cryptography and Coding 1997. LNCS, vol. 1355, pp. 323–334. Springer, Heidelberg (1997)
16. Mitchell, C.J.: Cryptanalysis of the EPBC authenticated encryption mode. In: Galbraith, S.D. (ed.) Cryptography and Coding 2007. LNCS, vol. 4887, pp. 118–128. Springer, Heidelberg (2007)
17. International Organization for Standardization Genève, Switzerland: ISO/IEC 18033-3:2010, Information technology — Security techniques — Encryption algorithms — Part 3: Block ciphers, 2nd edn. (2010)

# A Chosen IV Related Key Attack on Grain-128a

Subhadeep Banik[1], Subhamoy Maitra[1],
Santanu Sarkar[2], and Turan Meltem Sönmez[2]

[1] Applied Statistics Unit, Indian Statistical Institute Kolkata, 203, B.T. Road, Kolkata-108
`s.banik_r@isical.ac.in, subho@isical.ac.in`

[2] National Institute of Standards and Technology, 100 Bureau Drive, Stop 8930 Gaithersburg, MD 20899-8930, USA
`santanu.sarkar@nist.gov, meltem.turan@nist.gov`

**Abstract.** Due to the symmetric padding used in the stream cipher Grain v1 and Grain-128, it is possible to find Key-IV pairs that generate shifted keystreams efficiently. Based on this observation, Lee et al. presented a chosen IV related Key attack on Grain v1 and Grain-128 at ACISP 2008. Later, the designers introduced Grain-128a having an asymmetric padding. As a result, the existing idea of chosen IV related Key attack does not work on this new design. In this paper, we present a Key recovery attack on Grain-128a, in a chosen IV related Key setting. We show that using around $\gamma \cdot 2^{32}$ ($\gamma$ is a experimentally determined constant and it is sufficient to estimate it as $2^8$) related Keys and $\gamma \cdot 2^{64}$ chosen IVs, it is possible to obtain $32 \cdot \gamma$ simple nonlinear equations and solve them to recover the Secret Key in Grain-128a.

**Keywords:** Cryptanalysis, eStream, Grain-128a, Related Keys, Stream Cipher.

## 1 Introduction

The Grain family of stream ciphers, proposed by Martin Hell, Thomas Johansson and Willi Meier in 2005, is designed for constrained devices. Grain v1 [16] is included in the final hardware portfolio of the eStream project [1]. To meet increased security requirements, the designers proposed a 128-bit version called Grain-128 in ISIT 2006 [17]. In both ciphers, the symmetric padding of all ones is used during the initialization of the internal state of the cipher, before the Key-IV mixing. Due to this symmetric padding, slide attacks based on the observation that one could obtain Key-IV pairs that produce $\epsilon$-bit shifted keystream with probability $2^{-2\epsilon}$, were reported in [9]. This probability was improved to $2^{-\epsilon}$ in [6]. In the SKEW conference of 2011, the designers proposed the Grain-128a cipher that accommodated both functionalities of message encryption and authentication [2,3]. In order to protect against the previous attacks, the designers used an asymmetric padding in the design of Grain-128a. For detailed cryptanalytic results related to this family, the reader may refer to [4,7–9,11–14,18–20,22,24,25].

The symmetric padding used in the initialization of Grain v1 and Grain-128 was also exploited in [20] to mount a chosen IV related Key attack. Their main idea is to use related Keys and chosen IVs to obtain shifted keystream and then to carefully study the scenario to obtain the Secret Key bits. The same attack fails against Grain-128a, for the following reasons:

1. The padding used in Grain-128a is a string of 31 ones followed by a zero. Because of this asymmetric nature of the padding it is not possible to obtain related Key-IV pairs that produce shifted keystream bits for less than 32 bit shifts by using the idea of [9,20]. Following their idea, getting related Key-IV pairs for 32-bit shifted keystream should require an expected $2^{64}$ trials.
2. In Grain-128a, the first 64 keystream bits and thereafter every alternate keystream bit are used for computation of a MAC and hence are not directly available to the attacker. This also ensures that Grain-128a is resistant against the attack proposed in [20].

Thus one can argue that an attack against Grain-128a in the chosen IV related Key setting is much more difficult and hence requires much higher computational effort compared to [20].

In this paper, first, we present a novel approach to obtain related Key-IV pairs that produce 32-bit shifted keystream with an expected number of $2^{32}$ random trials. Using these Key-IV pairs, we present a Key recovery attack on Grain-128a, in a chosen IV related Key setting. We show that using around $\gamma \cdot 2^{32}$ ($\gamma$ is an experimentally determined constant and it is sufficient to estimate it as $2^8$) related Keys and $\gamma \cdot 2^{64}$ chosen IVs, it is possible to obtain $32 \cdot \gamma$ simple nonlinear equations and solve them to recover the Secret Key in Grain-128a. We experimentally verified that solving these equations are practical, due to the simplicity of the equations.

The paper is organized as follows. In the next section, a brief explanation of chosen IV attacks and the structure of the Grain family of stream ciphers are presented. In Section 3, the Key-IV pairs that produce shifted keystreams in Grain-128a are discussed. In Section 4, the details of the chosen IV related Key attack are presented, along with the experimental results and a discussion of a few possible countermeasures to prevent attacks of this nature. Finally, in Section 5 the conclusions of the paper are given.

## 2 Preliminaries

### 2.1 Chosen IV Attacks

The model used in chosen IV attacks is as follows. The adversary is given access to an Oracle which is in possession of an unknown quantity (typically the Secret Key). The adversary can choose a public parameter of his choice (typically the IV) and ask the Oracle to encrypt a message of his choice. In the context of stream ciphers, this implies that the adversary is able to obtain keystream bits by querying the Oracle possessing the Secret Key with any IV of his choice

(See Fig. 1). The above process can be repeated with different IVs of the adversary's choice. The task of the adversary could be either (i) to compute the Secret Key efficiently or, (ii) to distinguish the keystream output from a random stream.

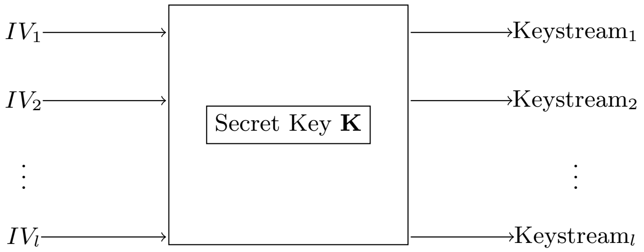


**Fig. 1.** Chosen IV Attack

The first model has been successfully employed in cube attacks on stream ciphers [12] whereas the second model has been used in distinguishing attacks on reduced round variants of stream and block ciphers [13, 14, 19, 22].

**Chosen IV Related Key Attack.** This attack model relaxes the requirements of the chosen IV attack slightly. It is assumed that the adversary can somehow obtain keystream bits corresponding to the Key-IV pair $[f_i(\mathbf{K}), IV_{i,j}]$, $i, j = 0, 1, 2, \ldots$, where $f_i : \mathcal{K} \rightarrow \mathcal{K}$ is a function from the Key-space $\mathcal{K}$ on to itself (See Fig. 2) and **K** is the Secret Key. As before the adversary attempts to recover the value of **K**. Chosen IV related Key attacks were successfully reported against Grain v1 and Grain-128 [20].

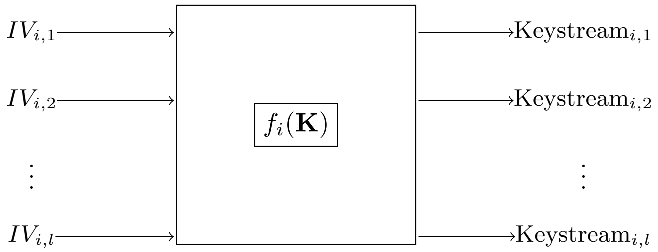


**Fig. 2.** Chosen IV Related Key Attack

## 2.2 Grain Family of Stream Ciphers

The Grain family of stream ciphers consists of two shift registers; an $n$-bit LFSR and an $n$-bit NFSR. Certain bits of both the registers are taken as inputs to a combining Boolean function, whence the keystream is produced. The structure of the Grain family is explained in Fig. 3. The update function of the LFSR is given by the equation $y_{t+n} = f(Y_t)$, where $Y_t = [y_t, y_{t+1}, \ldots, y_{t+n-1}]$ is an $n$-bit vector that denotes the LFSR state at time $t$ and $f$ is a linear function on the LFSR state bits obtained from a primitive polynomial in $GF(2)$ of degree $n$. The NFSR state is updated as $x_{t+n} = y_t + g(X_t)$. Here, $X_t = [x_t, x_{t+1}, \ldots, x_{t+n-1}]$ is an $n$-bit vector that denotes the NFSR state at time $t$ and $g$ is a nonlinear function of the NFSR state bits.

The output keystream is produced by combining the LFSR and NFSR bits as $z_t = h'(X_t, Y_t) = \bigoplus_{a \in A} x_{t+a} + h(X_t, Y_t)$, where $A$ is a subset of $\{0, 1, 2, \ldots, n-1\}$ fixed by the specification of each Grain variant.

**Key Loading Algorithm (KLA).** The Grain family uses an $n$-bit Key $K$, and an $m$-bit initialization vector $IV$, with $m < n$. The Key is loaded in the NFSR and the IV is loaded in the first $m$ bits of the LFSR. The remaining $n-m$ bits of the LFSR are loaded with some fixed padding $P \in \{0,1\}^{n-m}$. Hence at this stage, the $2n$ bit initial state is of the form $K||IV||P$.

**Key Scheduling Algorithm (KSA).** After the KLA, for the first $2n$ clocks, the output of the function $h'$ is XOR-ed to both the LFSR and NFSR update functions, i.e., during the first $2n$ clock intervals, the LFSR and the NFSR bits are updated as $y_{t+n} = z_t + f(Y_t)$, $x_{t+n} = y_t + z_t + g(X_t)$.

**Pseudo-Random keystream Generation Algorithm (PRGA).** After the of KSA, $z_t$ is no longer XOR-ed to the LFSR and NFSR update functions but it is used as the output keystream bit. Therefore during this phase, the LFSR and NFSR are updated as $y_{t+n} = f(Y_t), x_{t+n} = y_t + g(X_t)$.

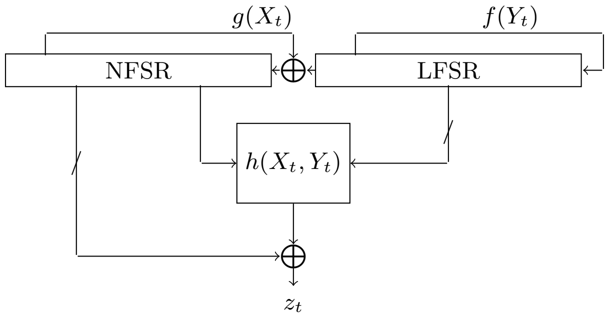


**Fig. 3.** Structure of Stream Cipher in Grain Family

### 2.3 Description of Grain-128a

Grain-128a authenticated encryption scheme consists of a 128 bit LFSR and a 128 bit NFSR. The size of the Key and IV is $n = 128$ and $m = 96$ bits, respectively. The value of the padding is $P = 0\texttt{xffff fffe}$. The LFSR update function is given by

$$y_{t+128} \overset{\Delta}{=} f(Y_t) = y_{t+96} + y_{t+81} + y_{t+70} + y_{t+38} + y_{t+7} + y_t.$$

The NFSR state is updated as follows

$$\begin{aligned} x_{t+128} = \ & y_t + g(x_{t+96}, x_{t+95}, x_{t+93}, x_{t+92}, x_{t+91}, x_{t+88}, x_{t+84}, x_{t+82}, x_{t+78}, \\ & x_{t+70}, x_{t+68}, x_{t+67}, x_{t+65}, x_{t+61}, x_{t+59}, x_{t+48}, x_{t+40}, x_{t+27}, \\ & x_{t+26}, x_{t+25}, x_{t+24}, x_{t+22}, x_{t+13}, x_{t+11}, x_{t+3}, x_t), \end{aligned}$$

where $g(x_{t+96}, x_{t+95}, \dots, x_t) \overset{\Delta}{=} g(X_t) =$

$$\begin{aligned} & x_t + x_{t+26} + x_{t+56} + x_{t+91} + x_{t+96} + x_{t+3}x_{t+67} + x_{t+11}x_{t+13} + \\ & x_{t+17}x_{t+18} + x_{t+27}x_{t+59} + x_{t+40}x_{t+48} + x_{t+61}x_{t+65} + x_{t+68}x_{t+84} + \\ & x_{t+88}x_{t+92}x_{t+93}x_{t+95} + x_{t+22}x_{t+24}x_{t+25} + x_{t+70}x_{t+78}x_{t+82}. \end{aligned}$$

The pre-output function $z_t$ is defined as

$$\sum_{j \in A} x_{t+j} + y_{t+93} + h(x_{t+12}, y_{t+8}, y_{t+13}, y_{t+20}, x_{t+95}, y_{t+42}, y_{t+60}, y_{t+79}, y_{t+94})$$

where $A = \{2, 15, 36, 45, 64, 73, 89\}$ and $h(s_0, \dots, s_8) = s_0s_1 + s_2s_3 + s_4s_5 + s_6s_7 + s_0s_4s_8$. The output function is defined as $y_t = z_{64+2t}$.

**Authentication.** We use the description as explained in [3]. Assume that we have a message of length $L$ defined by the bits $m_0, \dots, m_{L-1}$. Set $m_L = 1$. To provide authentication, two registers, called accumulator and shift register of size 32 bits each, are used. The content of accumulator and shift register at time $t$ are denoted by $a_t^0, \dots, a_t^{31}$ and $r_t, \dots, r_{t+31}$, respectively. The accumulator is initialized through $a_0^t = z_t, 0 \le t \le 31$ and the shift register is initialized through $r_t = z_{32+t}, 0 \le t \le 31$. The shift register is updated as $r_{t+32} = z_{64+2t+1}$. The accumulator is updated as $a_{t+1}^j = a_t^j + m_t r_{t+j}$ for $0 \le j \le 31$ and $0 \le t \le L$. The final content of accumulator, $a_{L+1}^0, \dots, a_{L+1}^{31}$ is used for authentication.

## 3 Key-IV Pairs That Produce Shifted Keystream in Grain-128a

In [9], a method to obtain Key-IV pairs $K, IV$ and $K', IV'$ in Grain v1 and Grain-128, that produce $\epsilon$-bit shifted keystream bits by performing a random experiment $2^{2\epsilon}$ many times was presented. The complexity was improved to $2^{\epsilon}$

in [6]. Both these techniques utilized the fact that the padding $P$ used in Grain v1 and Grain-128 was symmetric, i.e. a string of all ones. And in both [6,9], it was suggested that the method would fail if an asymmetric padding was used. This is precisely the strategy employed in Grain-128a, where the padding is $P =$`0x ffff fffe` is a set of 31 ones followed by a single zero.

In this section, we explain how despite of the asymmetric nature of $P$, one can obtain related Key-IV pairs $K, IV$ and $K', IV'$ in Grain-128a such that they produce exactly 32-bit shifted keystream by running a random experiment $2^{32}$ times. We begin by noting that the state update functions in both the KSA and PRGA in the Grain family are one-to-one and invertible. This is because the state update functions of the NFSR and the LFSR can be written in the form

$$g(x_0, x_1, \dots, x_{127}) = x_0 + g'(x_1, \dots, x_{127})$$

$$f(y_0, y_1, \dots, y_{127}) = y_0 + f'(y_1, \dots, y_{127}).$$

This implies that one can construct the KSA$^{-1}$ routine that takes a $2n$ bit vector $S_i$ denoting the internal state of the cipher at any $i^{th}$ round of the KSA, returns the $2n$ bit vector $S_{i-1}$ denoting the internal state of the cipher at the previous round of the KSA. The same is true for the PRGA. A detailed description of the KSA$^{-1}$ routine are given in Algorithm 1.

**Algorithm 1.** KSA$^{-1}$ routine for Grain-128a

```
Input: State $S_i = (x_0, \dots, x_{127}, y_0, \dots, y_{127})$
Output: The preceding State $S_{i-1} = (x_0, \dots, x_{127}, y_0, \dots, y_{127})$
$l = y_{127}$ and $n = x_{127}$
for $t = 127$ to $1$ do
    $y_t = y_{t-1}$ and $x_t = x_{t-1}$
end
$z = \bigoplus_{a \in A} x_a + y_{93} + h(x_{12}, y_8, y_{13}, y_{20}, x_{95}, y_{42}, y_{60}, y_{79}, y_{94})$
$y_0 = z + l + f'(y_1, \dots, y_{127})$
$x_0 = z + n + y_0 + g'(x_1, \dots, x_{127})$
```

Given this information, our strategy to find related Key-IV pairs in Grain-128a will be as follows. Let $K = (k_0.k_1, k_2, \dots, k_{127})$ be the Key. We choose a 96-bit IV of the form

$$IV = (v_0, v_1, \dots, v_{63}, \underbrace{1, 1, \dots, 1, 0}_{32})$$

Therefore the initial state

$$\begin{aligned} S = K||IV||P =& (s_0, s_1, \dots, s_{255}) \\ =& (k_0, \dots, k_{127},\ v_0, \dots, v_{63}, \underbrace{1, 1, \dots, 1, 0}_{32}, \underbrace{1, 1, \dots, 1, 0}_{32}). \end{aligned}$$

If we apply the KSA$^{-1}$ to $S$, 32 times, then we get the following internal state;

$$S' = (a_0, a_1, \ldots, a_{31}, k_0, k_1, \ldots, k_{95},\ b_0, b_1, \ldots, b_{31}, v_0, v_1, v_{63}, 1, \ldots, 1, 0).$$

where the values of $a_i, b_i$ for $0 \le i \le 31$ are given by polynomial functions in $k_0, \ldots, k_{127}$, $v_0, \ldots, v_{63}$. The exact form of these functions can be found out by executing the KSA$^{-1}$ routine 32 times.

Note that $S'$ is a valid initial state for Grain-128a, since it is of the form $K'||IV'||P$, where the value of $K' = (a_0, a_1, \ldots, a_{31}, k_0, k_1, \ldots, k_{95})$ and that of $IV' = (b_0, b_1, \ldots, b_{31}, v_0, v_1, v_{63})$. Therefore if one were to initialize Grain-128a with $K', IV'$ then the internal state of the cipher after the KSA round $32+t$ will be the same as the internal state after $t$ rounds of initialization with $K, IV$. This would be true for all $t \le 224$. After this, the cipher initialized with $K', IV'$ would enter the PRGA phase while the one initialized with $K, IV$ would still be in the KSA phase. As we have already seen, in the Grain family of ciphers, the output bit feedback to the internal state, is discontinued after the KSA. Therefore the state updates in the next 32 rounds are not guaranteed to be identical. The situation has been explained pictorially in Fig. 4.

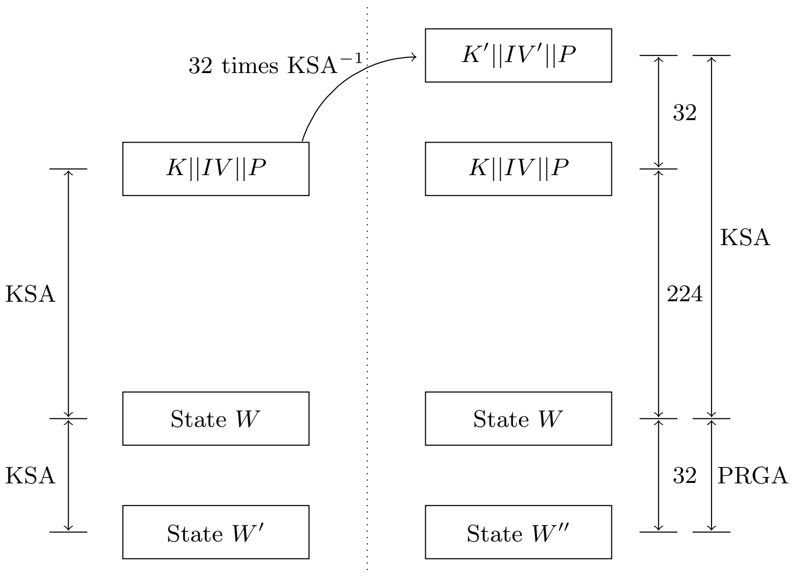


**Fig. 4.** Construction of Related Key-IV pairs in Grain Family

For the state updates to be identical in the next 32 rounds, it is necessary and sufficient that the cipher initialized with $K', IV'$ produces zero keystream bits for each of these 32 rounds. After this, both systems run in PRGA mode and so if the internal state of the cipher with $K, IV$ just after the KSA is equal to the internal state of the cipher with $K', IV'$ after 32 PRGA rounds, then they will remain the same forever thereafter. In such a situation the $(32+t)^{th}$ PRGA state produced by $K', IV'$ will be equal to the $t^{th}$ PRGA state produced by $K, IV$ for all $t > 0$. In such a situation it is natural that $K', IV'$ and $K, IV$ will produce 32 bit shifted keystream bits.

Now if we choose random values of $K \in \{0,1\}^{128}$ and $IV = V||P$ with $V \in \{0,1\}^{64}$, then it is expected that in one out of $2^{32}$ trials we will obtain a $K', IV'$ which produces an all zero output stream in the first 32 PRGA rounds. If so, $K', IV'$ and $K, IV$ will produce 32 bit shifted keystream bits. The arguments are formalized in Algorithm 2.

**Algorithm 2.** Constructing Key-IV pairs that generate 32 bit shifted keystream

```
Output: Key-IV pairs $K', IV'$ and $K, IV$ that generate 32 bit shifted
        keystream
$s \leftarrow 0$;
while $s = 0$ do
    Choose $K \in_R \{0,1\}^{128}$, $V \in_R \{0,1\}^{64}$;
    $IV \leftarrow V||P$;
    Run KSA$^{-1}$($K||IV||P$) routine for 32 clocks and produce state
    $S' = (K'||IV'||P)$;
    if $K', IV'$ produces all zero keystream bits in the first 32 PRGA rounds
    then
        $s \leftarrow 1$;
        Return $(K, IV)$ and $(K', IV')$;
    end
end
```

*Example 1.* In the following table, we present two Key-IV pairs that generate 32-bit shifted keystreams for Grain-128a. It can be seen that the second Key-IV pair has been obtained by the right shifting the first Key-IV pair by 32 bits. The pairs were found in around $2^{32}$ random trials using Algorithm 2. It should be noted that output bits given in the table includes the bits used for authentication and encryption.

| Pair | Key | IV | Output bits |
|---|---|---|---|
| 1 | 9bbe 7e2b b99d 1477 | 5a7c 21e9 3a77 | 41d5c1f0387c |
|  | 0317 9f3b a1aa 8c70 | 52ce ffff fffe | 3bf64e031725 |
| 2 | f32a 7bd3 9bbe 7e2b | 032d 0fee 5a7c | 0000000041d5c1f0387c |
|  | b99d 1477 0317 9f3b | 21e9 3a77 52ce | 3bf64e031725 |

*Remark 1.* It is also possible to obtain two Key-IV pairs $K_1, IV_1$ and $K_2, IV_2$ that produce $r$-bit shifted keystream bits (where $1 \leq r \leq 31$) by using a slight modification of the ideas presented in [6]. But in that case $K_1, IV_1$ and $K_2, IV_2$ would be structurally unrelated i.e. no meaningful similarity exists between these pairs. Such pairs cannot be used to mount a chosen IV related Key attack of the nature that we are about to describe.

## 4 A Chosen IV Related Key Attack on Grain-128a

We will now present a technique to cryptanalyze Grain-128a in the related Key and chosen IV setting i.e. in accordance to the model presented in Section 2.1. It is worth noting that Algorithm 2 cannot be directly applied in this problem due to two reasons. First, the Key is assumed to be secret in this model and so executing the KSA$^{-1}$ routine 32 times has to be done over the Key variables $k_0, k_1, \ldots, k_{127}$ rather than bits. A second reason is that in Grain-128a the first 64 keystream bits and every alternate keystream bit thereof goes towards the computation of MAC and is unavailable to the attacker directly. Hence it is not possible to check if the first 32 keystream bits produced by any Key-IV pair is all zero or not.

Let $K = (k_0, k_1, k_2, \ldots, k_{127})$ be the 128-bit Secret Key. We will write $K = \alpha_0||\alpha_1||\alpha_2||\alpha_3$ where each $\alpha_i$ is a 32-bit word given by the equation $\alpha_i = (k_{32i}, k_{32i+1}, \ldots, k_{32i+31})$ etc. Let the initial vector $IV = \beta_0||\beta_1||P$, where $\beta_i$s are 32 bit words. If we initialize the cipher with $K, IV$ we get the initial state

$$S = \alpha_0||\alpha_1||\alpha_2||\alpha_3 \ || \ \beta_0||\beta_1||P||P.$$

Now let us fix $\beta_0$ and $\beta_1$ to some fixed 32-bit values and let the $\alpha_i$s be unknowns, and apply the KSA$^{-1}$ routine over $S$, 32 times. We will get a new state $S'$ of the form

$$S' = \chi||\alpha_0||\alpha_1||\alpha_2 \ || \ \Upsilon||\beta_0||\beta_1||P.$$

where each bit in $\chi$ can be expressed as polynomial functions over the Secret Key variables $k_0, k_1, \ldots, k_{127}$. The form of these polynomials will of course depend on the exact values of $\beta_0, \beta_1$. So, we can write

$$\chi = f_{\beta_0||\beta_1}(\alpha_0, \alpha_1, \alpha_2, \alpha_3),$$

where $f_{\beta_0||\beta_1} : \{0,1\}^{128} \to \{0,1\}^{32}$ denotes a set of 32 Boolean functions. Similarly one can write

$$\Upsilon = g_{\beta_0||\beta_1}(\alpha_0, \alpha_1, \alpha_2, \alpha_3),$$

where $g_{\beta_0||\beta_1} : \{0,1\}^{128} \to \{0,1\}^{32}$ denotes another set of 32 Boolean functions. The exact forms of the functions $f_{\beta_0||\beta_1}, g_{\beta_0||\beta_1}$ for any value of $\beta_0, \beta_1$ can be computed efficiently by implementing the KSA$^{-1}$ routine in any computer algebra system like Sage 5.4.1 [23]. Note that for $S$ and $S'$ to produce 32-bit shifted keystream we need that the first 32 output bits produced by $S'$ be all 0s. Again this cannot be checked as the first 64 keystream bits are not directly available. Therefore our strategy to proceed will be as follows

1. Fix some value of $\beta_0$ and $\beta_1$.
2. Calculate the polynomials $f_{\beta_0||\beta_1}, g_{\beta_0||\beta_1}$.
3. Query the oracle for keystream bits produced by $K = \alpha_0||\alpha_1||\alpha_2||\alpha_3$, $IV = \beta_0||\beta_1||P$ and $K' = f_{\beta_0||\beta_1}(\alpha_0, \alpha_1, \alpha_2, \alpha_3)||\alpha_0||\alpha_1||\alpha_2$, $IV' = \eta||\beta_0||\beta_1$, where $\eta$ varies over all possible 32 bit words.

4. We check if, for any value of $\eta$, we get 32-bit shifted keystream bits. This can be done by checking the keystream bits after round 64 that Grain-128a makes directly available.
5. We will get 32-bit shifted keystream if and only if the following two occur simultaneously
   **A.** $\eta = g_{\beta_0||\beta_1}(\alpha_0, \alpha_1, \alpha_2, \alpha_3)$ AND
   **B.** The first 32 keystream bits produced by $K', IV'$ are all zeros.
6. We know that **A.** will be satisfied for exactly one value of $\eta$ and for that value of $\eta$, the condition **B.** may not hold and so for this value of $\beta_0, \beta_1$, none of the $2^{32}$ values of $\eta$ yields shifted keystream bits. In such an event we take a new value of $\beta_0$ and $\beta_1$ and repeat the process.

Now we know that on expectation, by trying out $2^{32}$ random values of $\beta_0$ and $\beta_1$, we are likely to land up with a related Key-IV pair $K', IV'$ that produces all zeroes in the first 32 output rounds. Therefore by running the above experiment $2^{32}$ times we are likely to obtain some values of $\beta_0, \beta_1, \eta$ such that $K' = f_{\beta_0||\beta_1}(\alpha_0, \alpha_1, \alpha_2, \alpha_3)||\alpha_0||\alpha_1||\alpha_2$, $IV' = \eta||\beta_0||\beta_1$ such that $K', IV'$ and $K, IV$ produce 32-bit shifted keystream bits. When this happens, we obtain the following set of 32 nonlinear equations in the Secret Key bits;

$$\eta = g_{\beta_0||\beta_1}(\alpha_0, \alpha_1, \alpha_2, \alpha_3).$$

Hence, by repeating the above process for $\gamma_1 \cdot 2^{32}$ different values of $\beta_0, \beta_1$ for any fixed value of the Secret Key we will on expectation be able to obtain $32 \cdot \gamma_1$ equations.

Next, we can start the above process for the single bit left rotated version of the Secret Key $K$ i.e. $K \lll 1 = k_1, k_2, \ldots, k_{127}, k_0$. Then by expending the same computational effort we would be able to obtain another $32 \cdot \gamma_1$ nonlinear equations in the Key bits. In general by starting the routine with the $i$-bit cyclically left rotated Key $K \lll i$, for $i = 0, 1, 2, \ldots, \gamma_2 - 1$, we would in total get $32 \cdot \gamma_1 \cdot \gamma_2 = 32 \cdot \gamma$ equations in the Key bits which can be solved together to recover the Secret Key for some suitable value of $\gamma$.

## 4.1 Complexity of the Attack

For each phase of the attack that yields $32 \cdot \gamma_1$ equations we need to try out on an average $\gamma_1 \cdot 2^{32}$ different values of $\beta_0, \beta_1$, and for each these values of $\beta_0, \beta_1$ we need to try out in the worst case $2^{32}$ different values of $\eta$. This leads to the use of $\gamma_1 \cdot 2^{32}$ related Keys. Since we use $2^{32}$ IVs for each related Key this leads to a total of $\gamma_1 \cdot 2^{64}$ chosen IVs for each phase of the attack. We need to check at least the first 128 keystream bits output from each related Key-IV pair to determine if the keystreams are 32-bit shifts or not and so that requires $\gamma.2^{64} \cdot 128 = \gamma_1 \cdot 2^{71}$ keystream bits. Repeating each phase $\gamma_2$ number of times for each rotated version of the Secret Key, increases the number of related Keys to $\gamma_1 \cdot \gamma_2 \cdot 2^{32} = \gamma \cdot 2^{32}$, the number of chosen IVs to $\gamma \cdot 2^{64}$ and the number of keystream bits to $\gamma \cdot 2^{71}$. The computational effort is therefore proportional to $\gamma \cdot 2^{64}$.

### 4.2 Experimental Results

After obtaining $32 \cdot \gamma$ nonlinear equations, the attacker needs to solve these equations to obtain the Secret Key. For the attack to be meaningful, the nonlinear equations should be as simple as possible so that the attacker can solve the system efficiently. However, to get $32 \cdot \gamma$ equations, the attacker needs a computational effort of the order of $\gamma \cdot 2^{64}$, which is infeasible with the processing resources at our disposal. So, in order to prove that after obtaining the required number of equations, the attacker can recover the Secret Key efficiently we make the following assumptions.

1. We assume that the attacker has succeeded in obtaining $32 \cdot \gamma_1$ equations by using $\gamma_1$ random tuples of $[\beta_0, \beta_1, \eta]$. Using each such tuple we construct the set of 32 equations
$$\eta = g_{\beta_0 || \beta_1}(\alpha_0, \alpha_1, \alpha_2, \alpha_3).$$
We have simulated this situation as the actual values of $[\beta_0, \beta_1, \eta]$ are difficult to obtain in practical time using the computational resources that we have.
2. We have observed that for each tuple $[\beta_0, \beta_1, \eta]$, only a few of the equations are very complex, therefore they were not used in our system of equations. Out of each set of 32 equations around $20 - 22$ equations are of low degree and are used for solving the system.
3. We repeat the above process for $\gamma_2$ cyclically left rotated versions of the Secret Key and thus obtain $32 \cdot \gamma_1 \cdot \gamma_2 = 32 \cdot \gamma$ equations in the Secret Key bits.

**Table 1.** Experimental Results

| $\gamma_1$ | $\gamma_2$ | $\gamma = \gamma_1 \cdot \gamma_2$ | Total # Polynomials | Time (in seconds) |
|---|---|---|---|---|
| 20 | 13 | 260 | $20\gamma$ | 112.80 |
| 20 | 11 | 220 | $22\gamma$ | 3240.62 |

After this we attempt to solve the equations so obtained, by using the SAT solver Cryptominisat-2.9.5 [21] available in the computer algebra system Sage 5.4.1 [23]. Table 1 lists the total number of polynomials and the required time to solve these equations to obtain the Secret Key. As seen from the table, it is possible to solve these equations in less than 1 hour using a Dual Core PC, with a CPU speed of 1.83 GHz and 2 GB RAM. These results show that once the attacker can obtain enough equations, then he can solve them efficiently to recover the Secret Key.

### 4.3 Possible Countermeasures

The computational complexity required to mount our attack is given as $\gamma \cdot 2^{64} = \gamma \cdot 2^{2|P|}$ where $|P| = 32$ is the length of the padding used in the initialization of Grain-128a. Thus to make the attack worse than brute force $|P|$ needed to be more than or equal to half the length in bits of the Secret Key i.e. $\frac{128}{2} = 64$. So prevention of such an attack on Grain-like ciphers requires that the bit-length of the padding be atleast half the bit-length of the Secret Key. For Trivium like ciphers where there is no difference in the operations performed during the KSA and the PRGA, the length of padding must be atleast equal to the length of the Secret Key (this is indeed the case for Trivium whose Keylength is 80 and where the length of the padding is 128).

Another popular approach used to prevent slide attacks altogether is the ones used in KATAN [10] and Quark [5], where update of two shift registers would be controlled by a third register which is usually initialized to a fixed constant at the start of operations. Performing a slide attack on then would require a simultaneous synchronization of the third register for the related Key-IV pair, which is not possible as it always starts with a fixed constant. This of course requires extra hardware and hence increases the area and power consumption of the device implementing the cipher.

## 5 Conclusion

In this paper we present a chosen IV related Key attack against the stream cipher Grain-128a. A similar attack against Grain v1 and Grain-128 were proposed in [20]. The attack worked due to the symmetric padding used in both Grain v1 and Grain-128. The new design, Grain-128a, uses an asymmetric padding and consequently the attack of Lee et. al. [20] does not work in this scenario. We show that by using around $\gamma \cdot 2^{32}$ related Keys and $\gamma \cdot 2^{64}$ chosen IVs the attacker can obtain $32 \cdot \gamma$ nonlinear equations in the Secret Key bits which he can then solve to recover the Secret Key in Grain-128a. Our attack on Grain-128a requires higher complexities than that of [20] on Grain v1 and Grain-128. However obtaining attacks against Grain-128a with lesser complexities is elusive due to the asymmetric padding.

**Acknowledgments.** The authors like to thank the anonymous reviewers for their helpful suggestions. The first three authors like to acknowledge the Centre of Excellence in Cryptology, Indian Statistical Institute for supporting this research.

## References

1. The ECRYPT Stream Cipher Project. eSTREAM Portfolio of Stream Ciphers (Revised on September 8, 2008)
2. Ågren, M., Hell, M., Johansson, T., Meier, W.: A New Version of Grain-128 with Authentication. In: Symmetric Key Encryption Workshop 2011, DTU, Denmark (February 2011)

3. Ågren, M., Hell, M., Johansson, T., Meier, W.: Grain-128a: A New Version of Grain-128 with Optional Authentication. IJWMC 5(1), 48–59 (2011); This is the journal version of [2]
4. Aumasson, J.P., Dinur, I., Henzen, L., Meier, W., Shamir, A.: Efficient FPGA Implementations of High-Dimensional Cube Testers on the Stream Cipher Grain-128. In: SHARCS - Special-purpose Hardware for Attacking Cryptographic Systems (2009)
5. Aumasson, J.P., Henzen, L., Meier, W., Naya-Plasencia, M.: Quark: A Lightweight Hash. Journal of Cryptology 26(2), 313–339 (2013)
6. Banik, S., Maitra, S., Sarkar, S.: Some Results on Related Key-IV Pairs of Grain. In: Bogdanov, A., Sanadhya, S. (eds.) SPACE 2012. LNCS, vol. 7644, pp. 94–110. Springer, Heidelberg (2012)
7. Berbain, C., Gilbert, H., Maximov, A.: Cryptanalysis of Grain. In: Robshaw, M. (ed.) FSE 2006. LNCS, vol. 4047, pp. 15–29. Springer, Heidelberg (2006)
8. Bjørstad, T.E.: Cryptanalysis of Grain using Time/Memory/Data tradeoffs (v1.0 / February 25, 2008), `http://www.ecrypt.eu.org/stream`
9. De Cannière, C., Küçük, Ö., Preneel, B.: Analysis of Grain's Initialization Algorithm. In: Vaudenay, S. (ed.) AFRICACRYPT 2008. LNCS, vol. 5023, pp. 276–289. Springer, Heidelberg (2008)
10. 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)
11. Dinur, I., Güneysu, T., Paar, C., Shamir, A., Zimmermann, R.: An Experimentally Verified Attack on Full Grain-128 Using Dedicated Reconfigurable Hardware. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 327–343. Springer, Heidelberg (2011)
12. Dinur, I., Shamir, A.: Breaking Grain-128 with Dynamic Cube Attacks. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 167–187. Springer, Heidelberg (2011)
13. Englund, H., Johansson, T., Sönmez Turan, M.: A Framework for Chosen IV Statistical Analysis of Stream Ciphers. In: Srinathan, K., Rangan, C.P., Yung, M. (eds.) INDOCRYPT 2007. LNCS, vol. 4859, pp. 268–281. Springer, Heidelberg (2007)
14. Fischer, S., Khazaei, S., Meier, W.: Chosen IV Statistical Analysis for Key Recovery Attacks on Stream Ciphers. In: Vaudenay, S. (ed.) AFRICACRYPT 2008. LNCS, vol. 5023, pp. 236–245. Springer, Heidelberg (2008)
15. Fredricksen, H.: A Survey of Full Length Nonlinear Shift Register Cycle Algorithms. SIAM Rev. 24, 195–221 (1982)
16. Hell, M., Johansson, T., Meier, W.: Grain - A Stream Cipher for Constrained Environments. ECRYPT Stream Cipher Project Report 2005/001 (2005), `http://www.ecrypt.eu.org/stream`
17. Hell, M., Johansson, T., Maximov, A., Meier, W.: A Stream Cipher Proposal: Grain-128. In: IEEE International Symposium on Information Theory, ISIT 2006 (2006)
18. Khazaei, S., Hassanzadeh, M., Kiaei, M.: Distinguishing Attack on Grain. ECRYPT Stream Cipher Project Report 2005/071 (2005), `http://www.ecrypt.eu.org/stream`
19. Knellwolf, S., Meier, W., Naya-Plasencia, M.: Conditional Differential Cryptanalysis of NLFSR-based Cryptosystems. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 130–145. Springer, Heidelberg (2010)
20. Lee, Y., Jeong, K., Sung, J., Hong, S.: Related-Key Chosen IV Attacks on Grain-v1 and Grain-128. In: Mu, Y., Susilo, W., Seberry, J. (eds.) ACISP 2008. LNCS, vol. 5107, pp. 321–335. Springer, Heidelberg (2008)

21. Soos, M.: CryptoMiniSat-2.9.5, `http://www.msoos.org/cryptominisat2/`
22. Stankovski, P.: Greedy Distinguishers and Nonrandomness Detectors. In: Gong, G., Gupta, K.C. (eds.) INDOCRYPT 2010. LNCS, vol. 6498, pp. 210–226. Springer, Heidelberg (2010)
23. Stein, W.: Sage Mathematics Software. Free Software Foundation, Inc. (2009), `http://www.sagemath.org` (Open source project initiated by W. Stein and contributed by many)
24. Zhang, B., Li, Z.: Near Collision Attack on the Grain v1 Stream Cipher. To appear in FSE 2013 (2013)
25. Zhang, H., Wang, X.: Cryptanalysis of Stream Cipher Grain Family. IACR Cryptology ePrint Archive 2009: 109 (2009), `http://eprint.iacr.org/2009/109`

# Cryptanalysis of Helix and Phelix Revisited*

Zhenqing Shi[1], Bin Zhang[2], and Dengguo Feng[1]

[1] Institute of Software, Chinese Academy of Sciences, Beijing, 100190, China
[2] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, 100195, China
zhenqingshi@gmail.com, {zhangbin,feng}@is.iscas.ac.cn

**Abstract.** Helix, designed by Ferguson et al., is a high-speed asynchronous stream cipher with a built-in MAC functionality. At FSE 2004, Muller presented two attacks on Helix. Motivated by these attacks, Phelix was proposed and selected as a Phase 2 focus cipher for both Profile 1 and Profile 2 by the eSTREAM project, but was not advanced to Phase 3 mainly due to a key recovery attack by Wu and Preneel when the prohibition against reusing a nonce is violated.

In this paper, we study the security of Helix and Phelix in the more realistic chosen nonce model. We first point out a flaw in Muller's second attack, which results in the failure of his attack. Then we propose our distinguishing attack on Helix with a data complexity of $2^{132}$, faster than exhaustive search when the key length is larger than 132 bits. Furthermore, when the maximal length of output keystream is extended, the data complexity can be reduced to $2^{127}$ and we also can construct a key recovery attack with a data complexity of $2^{163}$. Since this flaw is overlooked by the designers of Phelix, we can extend the distinguishing attack to Phelix with the same complexity, which shows that Phelix fails to strengthen Helix against internal state collision attacks. Our results provide new insights on the design of such dedicated ciphers with built-in authentication.

**Keywords:** Authenticated encryption, Stream ciphers, ARX, Helix, Phelix, eSTREAM.

## 1 Introduction

In the last 20 years, a huge number of symmetric primitives using modular additions, interword rotations, and exclusive ors (ARX) have appeared, e.g., the MD-family hash functions (MD4, MD5) and their descendants SHA-x, the stream ciphers Helix [1], Phelix [2] and Salsa20 [3]. These primitives usually have

* This work was supported by the National Grand Fundamental Research 973 Program of China(Grant No. 2013CB338002), the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDA06010701), IIE's Research Project on Cryptography (Grant No. Y3Z0016102) and the programs of the National Natural Science Foundation of China (Grant No. 60833008, 60603018, 61173134, 91118006, 61272476).

a high-speed and a low price of implementation in both software and hardware. However, the security of these primitives is not well understood.

Helix is a stream cipher proposed by Ferguson et al. [1] at FSE 2003. It is a high-speed asynchronous stream cipher with a built-in MAC functionality. At FSE 2004, Muller published two attacks on Helix [4]. The first has a complexity of $2^{88}$ and requires $2^{12}$ adaptive chosen-plaintext words, but it requires nonces to be reused. In [5], Paul and Preneel developed an optimal algorithm to solve differential equations of addition and reduced the number of adaptive chosen-plaintext words by a factor of 3 in the worst case, and a factor of 46.5 in the best case. Later, they showed that Muller's attack can be even launched with chosen plaintexts (CP) rather than adaptive chosen plaintexts (ACP) with data complexity $2^{35.64}$ CP's [6]. The second attack on Helix is a distinguishing attack by utilizing internal state collisions, which requires $2^{114}$ words of chosen plaintext in the chosen nonce model, i.e., it does not require nonces to be reused. Until our work, it is commonly believed that Helix was broken in theory by this attack.

As a result, a strengthened version, Phelix, was released by adding an output function instead of extracting one word of the internal state directly. In 2004, Phelix was submitted to the eSTREAM contest and was selected as a Phase 2 Focus Candidate for both Profile 1 and Profile 2, but was not advanced to Phase 3 mainly due to Wu and Preneel's key recovery attack [7]. Their attack shows that if the cipher is used incorrectly (nonces reused), the key of Phelix can be recovered with about $2^{37}$ operations, $2^{34}$ chosen nonces and $2^{38.2}$ chosen plaintext words. However, there is some debate on the validity of this attack model, notably by Bernstein [8]. A counter example is that all additive stream ciphers can be broken under the above security definition if the adversary is not nonce-respecting. Despite of the above disputes, Helix and Phelix provide an innovative and interesting design approach according to the final eSTREAM portfolio report [9], especially in terms of the increasing demand of the authenticated encryption [10].

In this paper, we study the security of Helix and Phelix in the more realistic chosen nonce model. We first point out a flaw in Muller's second attack on Helix, which results in the failure of this attack. Precisely, there is an implicit independence assumption in Muller's attack that each collection of eight consecutive keystream words is independent to each other. Unfortunately, this assumption does not always hold. We find that under some conditions, a state collision can propagate for more than 8 rounds, thus the independent assumption is violated. This results in many invalid pairs in Muller' attack. Based on this finding, we propose our distinguishing attack and key recovery attack on Helix without nonce reused. Since this flaw is overlooked by the designers of Phelix, we extend the distinguishing attack to Phelix and show that Phelix fails to strengthen Helix against internal state collision attacks. To the best of our knowledge, this is the first distinguishing attack on Phelix without nonces reused. Furthermore, we prove that any attempt to Helix by adding an output function, has the same security level as Helix with respect to internal state collision attacks, if the round interval of this output function is much smaller than $2^{31}$. For Phelix, the round interval of its output function is only 5, which is far less than $2^{31}$. This is a new insight on the design of such dedicated ciphers with built-in authentication.

This paper is organized as follows. In Section 2, we briefly describe the Helix and Phelix stream ciphers, and in Section 3 we recall Muller's second attack. In Section 4 we first point out the flaw in Muller's second attack and then present all the details of our attacks on Helix and Phelix, respectively. Finally, we conclude the paper in Section 5.

## 2 Description of Helix and Phelix

In this section we recall the Helix and Phelix briefly and all the details can be found in [1,2]. The only difference between Helix and Phelix is the output of the keystream words, so here we mainly introduce Helix.

### 2.1 General Structure of Helix

The Helix encryption function takes as input a variable length key $U$ of up to 32 bytes (produces the working key of 8 words $K_0, K_1, ..., K_7$ by key mixing), a 16-byte nonce $N$ (interpreted by 4 words $N_0, N_1, ..., N_3$), and a plaintext $P$. It produces a ciphertext message and a tag that provides authentication. The decryption function takes as input the key, nonce, ciphertext $C$, and tag, and produces either the plaintext message or an error if the authentication failed. The plaintext $P$ and ciphertext $C$ are both sequences of bytes of the same length, with the sequence length shorter than $2^{64}$. The only operations used are addition modulo $2^{32}(\boxplus)$, exclusive or$(\oplus)$, and rotation by fixed numbers of bits$(\lll)$.

Helix is based on an iterated block function applied to an internal state of 160 bits. The internal state before encryption of the $i$-th word of plaintext is represented as 5 words $(Z_0^{(i)}, ..., Z_4^{(i)})$ which are initialized for $i = -8$ using $K$ and $N$. It basically uses a block function $F$ to update the internal state in function of the plaintext $P$, the working key $K$ and nonce $N$. More precisely, during the $i$-th round, the internal state is updated with $F$, using the $i$-th word of plaintext $P_i$ and two words derived from $K$, $N$ and $i$, denoted as $X_{i,0}$ and $X_{i,1}$ (called key words). Hence,

$$(Z_0^{(i+1)}, ..., Z_4^{(i+1)}) = F(Z_0^{(i)}, ..., Z_4^{(i)}, P_i, X_{i,0}, X_{i,1}).$$

### 2.2 The Block Function

The block function $F$ of Helix mixes three types of basic operations on words: addition modulo $2^{32}$, exclusive or, and rotation by fixed numbers of bits. $F$ relies on two consecutive applications of a single "helix" function denoted as G (see in Fig.1).

$G$ uses two auxiliary inputs $(A$,$B)$. In the first half of the block function, $(A$,$B) = (0$,$X_{i,0})$ and in the second half, $(A$,$B) = (P_i$,$X_{i,1})$. Thus, the block

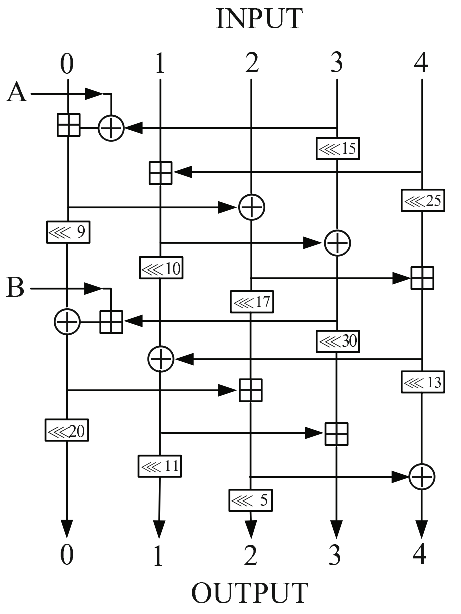


**Fig. 1.** The half-round "helix" function G

function can be described by the following relations

$$\begin{aligned}(Y_0^{(i)}, ..., Y_4^{(i)}) &= G(Z_0^{(i)}, ..., Z_4^{(i)}, 0, X_{i,0}) \\ (Z_0^{(i+1)}, ..., Z_4^{(i+1)}) &= G(Y_0^{(i)}, ..., Y_4^{(i)}, P_i, X_{i,1})\end{aligned}$$

where $(Y_0^{(i)}, ..., Y_4^{(i)})$ is the internal state in the middle of the computation.

### 2.3 The Key Words

The expanded key words are derived from the working key $K_0, ..., K_7$, the nonce $N_0, ..., N_3$, the input key length $\ell(U)$, and the block number $i$. We first extend the nonce to 8 words by defining $N_k := (k \bmod 4) - N_{k-4} (\bmod 2^{32})$ for $k = 4, ..., 7$. The key words for block $i$ are then defined by

$$\begin{aligned}X_{i,0} &:= K_{i \bmod 8} \\ X_{i,1} &:= K_{(i+4) \bmod 8} + N_{i \bmod 8} + X'_i + i + 8 \\ X'_i &:= \begin{cases} \lfloor (i+8)/2^{31} \rfloor & \text{if } i \bmod 4 = 3 \\ 4 \cdot \ell(U) & \text{if } i \bmod 4 = 1 \\ 0 & \text{otherwise} \end{cases}\end{aligned}$$

where all additions are taken modulo $2^{32}$.

### 2.4 Initialization

A Helix encryption is started by setting

$$\begin{aligned} Z_j^{(-8)} &:= K_{j+3} \oplus N_j \quad \text{for} \quad j = 0, ..., 3 \\ Z_4^{(-8)} &:= K_7. \end{aligned}$$

Eight blocks are then applied, using block number $i = -8$ to $-1$. For these blocks, the plaintext word $P_i$ is defined to be zero, and the generated key stream words are discarded.

### 2.5 The Keystream Words

After the initialization, the keystream word is output. For Helix, the keystream word $s_i := Z_0^{(i)}$, while for Phelix $s_i := Y_4^{(i)} \boxplus Z_4^{(i-4)}$, where $Z^{(-8-j)} = 0$ for $j = 1, ..., 4$.

## 3 Muller's Distinguishing Attack on Helix

This section describes Muller's second attack on Helix. As Muller said:" this attack is faster than exhaustive search, processes less than $2^{128}$ blocks of plaintext and respects the security requirements proposed in [1], since no pair (key,nonce) is ever reused to encrypt different messages. Therefore, this attack constitutes a theoretical break of Helix."

### 3.1 Influence of Each Nonce Word

Suppose that the same plaintext $P$ is encrypted twice with the same secret key, but two distinct nonces $N$ and $N'$ such that $N = (N_0, N_1, N_2, N_3)$, $N' = (N_0, N_1, N_2, N_3 + \Delta)$. It appears from Section 2.3 that the two key words introduced at each round do not depend on the full nonce. Actually, the key words at round $i$ depend only on $N_{i \bmod 4}$. Therefore, if we consider two distinct nonces $N$ and $N'$, the round function will essentially apply the same mapping on the internal state for 3 rounds out of 4, i.e., for any round $i$ such that $i \bmod 4 \neq 3$. If a state collision occurs on the input of such a round, it will also propagate to a state collision for the input of the next round. Thus state collisions on inputs of rounds $i$ such that $i \bmod 4 = 0$ imply collisions on 4 consecutive blocks of keystream.

### 3.2 Forcing the Collisions

The general idea of Muller's distinguishing attack on Helix is to work on a large set of nonces that will preserve collisions for a few rounds. Then these collisions can be detected by observing the corresponding keystream blocks. More precisely, suppose an attacker can build a message $P$ of the maximal authorized length

$2^{62}$ words by repeating $2^{62}$ times the same word $P_0$. Then, $P$ is encrypted under a fixed unknown secret key $K$ using different nonces of the form

$$N^{(\delta,\Delta)} = (N_0 + \delta, N_1 + \delta, N_2 + \delta, N_3 + \Delta)$$

with four fixed constants $(N_0, ..., N_3)$. $\delta$ is of the form $8 \times x$ where $x$ spans all values from 0 to $2^{20}$ and $\Delta$ spans all $2^{32}$ possible words. Therefore, the number of blocks encrypted is $2^{62} \times 2^{20} \times 2^{32} = 2^{114}$. Consider any state collision that occurs between two different nonces $N^{(\delta_1,\Delta_1)}$ and $N^{(\delta_2,\Delta_2)}$, at two different positions $i_1$ and $i_2$, respectively. This state collision should be preserved for several rounds, in order to detect some properties on the keystream, as in the previous section. It's sure that the plaintext word introduced in any round is always $P_0$. Furthermore, the round key words for both encryptions should be the same. Hence, these positions should satisfy

$$i_1 \bmod 8 = i_2 \bmod 8 = 0$$

in order to have $X_{i_1+j,0} = X_{i_2+j,0}$ for all $j$. Besides, if

$$\delta_1 + i_1 = \delta_2 + i_2 \bmod 2^{32} \tag{1}$$

then $X_{i_1+j,1} = X_{i_2+j,1}$ when $j \bmod 4 \neq 3$. In this case, the state collision is preserved for at least 3 rounds. Concerning rounds $i_1 + 3$ and $i_2 + 3$, the state collision is also expected to be preserved, thus it requires $X_{i_1+3,1} = X_{i_2+3,1}$ or

$$\Delta_1 + i_1 + X'_{i_1+3} = \Delta_2 + i_2 + X'_{i_2+3} \bmod 2^{32}. \tag{2}$$

With these three assumptions, the state collision is preserved at least until the rounds $i_1 + 7$ and $i_2 + 7$ which results in collisions on 8 consecutive keystream words.

To mount an attack, first store sequences of 8 consecutive keystream words, for each message and for each position $i$ such that $i \bmod 8 = 0$. Then, look for a collision among the $\frac{2^{114}}{8} = 2^{111}$ entries in this table. This can be achieved by sorting the table, with complexity of $2^{111} \times 111 \simeq 2^{118}$ basic comparisons. Then, since the considered objects are of 256 bits, the number of "fortuitous" collisions in the table is

$$\frac{2^{111} \times 2^{111}}{2} \times 2^{-256} \simeq 0.$$

Besides, when a state collision occurs, a collision is also observed on the entries of the table, provided the additional assumptions (1) and (2) hold, and denote such collision in the table "a true collision" in contrast to "a fortuitous collision". Furthermore, (1) holds with probability $2^{-29}$, since all terms are multiples of 8, and (2) holds with probability $2^{-32}$. Therefore, the number of true collisions observed in the table is in average

$$\frac{2^{111} \times 2^{111}}{2} \times 2^{-160} \times 2^{-29} \times 2^{-32} = 1.$$

Thus for a true Helix output an attacker expects to find at least one collision in the previous table, while it is not the case for a random output.

## 4 Our Attacks on Helix and Phelix

In this section we first calculate the success probability of Muller's collision attack on Helix. Then we point out the flaw in Muller's attack. Finally, based on our analysis, we construct our distinguishing attack and key recovery attack on Helix, and extend the distinguishing attack to Phelix.

### 4.1 The Success Probability and the Advantage of the Attack

Muller's collision attack would succeed if at least one collision is found in the table. However, the success probability is not given in [4]. Now we estimate this probability.

For a pair of 8 consecutive keystream words in Muller's collision attack, it is a collision with probability $p_0 = 2^{-221}$, then given $M$ independent pairs, the success probability of the attack is

$$Pr_0 = 1 - (1 - p_0)^M \approx 1 - e^{-Mp_0}.$$

While given $M$ independent pairs of random 256-bits words, the probability that there is at least one collision is

$$Pr = 1 - (1 - p)^M \approx 1 - e^{-Mp},$$

where $p = 2^{-256}$.

Thus, the advantage of the attack is

$$A = Pr_0 - Pr \approx e^{-Mp} - e^{-Mp_0}.$$

### 4.2 A Flaw in Muller's Attack on Helix

In section 3.2, $\delta$ is of the form $8 \times x$ where $x$ spans all values from 0 to $2^{20}$ and $\Delta$ spans all $2^{32}$ possible words. Therefore, the number of blocks encrypted is $2^{62} \times 2^{32} \times 2^{20} = 2^{114}$, i.e., $\frac{2^{114}}{8} = 2^{111}$ sequences of 8 consecutive keystream words. Then an attacker compares every two sequences to look for a collision. This process can be regarded as the following 4 steps:

1. Choose distinct $(\delta_1, \Delta_1)$ and $(\delta_2, \Delta_2)$, encrypt $P$ with the fixed $K$ and two distinct nonces $N^{(\delta_1, \Delta_1)}$ and $N^{(\delta_2, \Delta_2)}$, and get two keystreams, denoted by $S_1$ and $S_2$ respectively;
2. For $h = 1, 2$, divide $S_h$ into $2^{31}$ blocks, denoted by $S_{h,r} (r = 0, 1, ..., 2^{31} - 1)$, and each $S_{h,r}$ contains consecutive keystream words from round $2^{31}r - 8$ to round $2^{31}(r + 1) - 9$;
3. For $h = 1, 2$, $r = 0, 1, ..., 2^{31} - 1$, divide $S_{h,r}$ into $2^{28}$ blocks, denoted by $S_{h,r,t} (t = 0, 1, ..., 2^{28} - 1)$, and each $S_{h,r,t}$ contains 8 consecutive keystream words from round $2^{31}r + 8t - 8$ to round $2^{31}r + 8t - 1$;
4. For $0 \le r_1, r_2 \le 2^{31} - 1$, $0 \le t_1, t_2 \le 2^{28} - 1$, check whether $S_{1,r_1,t_1} = S_{2,r_2,t_2}$. If there exists a pair $(S_{1,r_1,t_1}, S_{2,r_2,t_2})$ satisfying this condition, output the successful information; otherwise, return to Step 1.

**Remark 1** *At Step 2, for $r = 0$, $S_{h,0}$ does not contain consecutive keystream words from round $-8$ to round $-1$ because there are no outputs in those rounds. And we also omit the last 8 consecutive keystream words for each $S_h$. These operations have negligible impact.*

The number of pairs used in Muller's attack can be estimated as follow:

1. At Step 1, there are $\frac{2^{52}\times 2^{52}}{2}$ pairs of $(S_1, S_2)$, because $\delta$ is of the form $8 \times x$ where $x$ spans all values from 0 to $2^{20}$ and $\Delta$ spans all $2^{32}$ possible words;
2. At Step 2, for each $(S_1, S_2)$, there are $2^{31} \times 2^{31}$ pairs of $(S_{1,r_1}, S_{2,r_2})$, because $0 \leq r_1, r_2 \leq 2^{31} - 1$;
3. At Step 3, for each $(S_{1,r_1}, S_{2,r_2})$, there are $2^{28} \times 2^{28}$ pairs of $(S_{1,r_1,t_1}, S_{2,r_2,t_2})$, because $0 \leq t_1, t_2 \leq 2^{28} - 1$, and all these pairs of $(S_{1,r_1,t_1}, S_{2,r_2,t_2})$ will be checked at Step 4.

Thus there are $\frac{2^{52}\times 2^{52}}{2} \times 2^{31} \times 2^{31} \times 2^{28} \times 2^{28}$ pairs. If those pairs are all independent, Muller's attack would succeed with probability $Pr_0 \approx 0.632$ and advantage $A \approx 0.632$. However, many pairs are not independent in Muller's attack.

**Lemma 1.** *For $0 \leq r_1, r_2 \leq 2^{31} - 1$, $0 \leq t_1 \leq t_2 \leq 2^{28} - 1$, $S_{1,r_1,t_1}$ and $S_{2,r_2,t_2}$ is a true collision if and only if $S_{1,r_1,0}$ and $S_{2,r_2,t_2-t_1}$ is a true collision.*

*Proof.*
$(\Longleftarrow)$

Denote $i_h(r,t) = 2^{31}r + 8t - 8$, $h = 1, 2$. Suppose $S_{1,r_1,0}$ and $S_{2,r_2,t_2-t_1}$ is a true collision, that means a state collision occurs at round $i_1(r_1, 0)$ and round $i_2(r_2, t_2 - t_1)$, and the additional assumptions (1) and (2) hold, i.e.,

$$\delta_1 + i_1(r_1, 0) = \delta_2 + i_2(r_2, t_2 - t_1) \bmod 2^{32} \tag{3}$$

$$\Delta_1 + i_1(r_1, 0) + X'_{i_1(r_1,0)+3} = \Delta_2 + i_2(r_2, t_2 - t_1) + X'_{i_2(r_2,t_2-t_1)+3} \bmod 2^{32}. \tag{4}$$

By (3) and the definition of $i_h(r,t)$, we can deduce

$$\delta_1 - \delta_2 = i_2(r_2, t_2 - t_1) - i_1(r_1, 0) = i_2(r_2, t_2 - t_1 + j) - i_1(r_1, j) \bmod 2^{32}$$

where $0 \leq j \leq 2^{28} - 1 + t_1 - t_2$, i.e.,

$$\delta_1 + i_1(r_1, j) = \delta_2 + i_2(r_2, t_2 - t_1 + j) \bmod 2^{32}. \tag{5}$$

From Section 2.3, we deduce

$$\begin{aligned} X'_{i_1(r_1,0)+3} = r_1 &= X'_{i_1(r_1,j)+3} \\ X'_{i_2(r_2,t_2-t_1)+3} = r_2 &= X'_{i_1(r_1,t_2-t_1+j)+3}. \end{aligned}$$

Then by (4) and the definition of $i_h(r,t)$, we obtain

$$\Delta_1 + i_1(r_1, j) + X'_{i_1(r_1,j)+3} = \Delta_2 + i_2(r_2, t_2 - t_1 + j) + X'_{i_2(r_2,t_2-t_1+j)+3} \bmod 2^{32}. \tag{6}$$

Thus the state collision on inputs of round $i_{1,r_1,0}$ and $i_{2,r_2,t_2-t_1}$ can be preserved for $8(2^{28}+t_1-t_2)$ rounds. So $S_{1,r_1,t_1}$ and $S_{2,r_2,t_2}$ is a true collision.
$(\Longrightarrow)$

Because the round function of Helix is invertible, the proof of necessary condition is the same as that of sufficient condition. $\square$

Thus for a fixed pair $(S_{1,r_1}, S_{2,r_2})$, the pair $(S_{1,r_1,t_1}, S_{2,r_2,t_2})$ can be classified into the equivalence classes

$$\{(S_{1,r_1,j}, S_{2,r_2,t_2-t_1+j})|0 \le j \le 2^{28}-1+t_1-t_2\}, \quad \text{when} \quad t_1 \le t_2$$

or

$$\{(S_{1,r_1,t_1-t_2+j}, S_{2,r_2,j})|0 \le j \le 2^{28}-1+t_2-t_1\}, \quad \text{when} \quad t_2 \le t_1.$$

By the proof of Lemma 1, we deduce that any pair in the equivalence class is a true collision if and only if the remaining pairs in the same equivalence class are true collisions. Therefore, we only need to check the representative element of each equivalence class to look for a true collision.

For each $(S_{1,r_1}, S_{2,r_2})$, there are $2^{28}+2^{28}$ equivalence classes. Thus the number of independent pairs in Muller's attack is $M = \frac{2^{52}\times 2^{52}}{2} \times 2^{31} \times 2^{31} \times (2^{28}+2^{28})$, and we can obtain the success probability $Pr_0 \approx 0$ and advantage $A \approx 0$.

## 4.3 Distinguishing Attack on Modified Helix

In this section, we present the improved distinguishing attack on Helix, provided that the maximal length of output keystream can be extended.

Suppose the maximal length of output keystream for a pair $(U,N)$ is $2^n$. First we build a message $P$ of the maximal length $2^n$ words by repeating $2^n$ times the same word $P_0$. Then, $P$ is encrypted under a fixed unknown secret key $K$ and two different nonces $N^{(\delta_1,\Delta_1)}$ and $N^{(\delta_2,\Delta_2)}$, where $\delta_1$ and $\delta_2$ are of the form $8\times x$ and $x$ spans all values from 0 to $2^{29}-1$, $\Delta_1$ and $\Delta_2$ span all $2^{32}$ possible words. Therefore, the number of blocks encrypted is $2^n \times 2^{29} \times 2^{32} = 2^{n+61}$. To mount an attack, we first store sequences of 8 consecutive keystream words, for each message and for each position $i$ such that $i \bmod 8 = 0$. Then, we look for a collision among the $\frac{2^{n+61}}{8} = 2^{n+58}$ entries in this table. This can be achieved by sorting the table, with complexity of $2^{n+58}\times(n+58)$ basic instructions. From Section 4.2, the number of independent pairs in our attack is

$$M = \frac{2^{61}\times 2^{61}}{2} \times 2^{n-31} \times 2^{n-31} \times (2^{28}+2^{28}) = 2^{2n+88}$$

And from Section 4.1, we obtain the success probability $Pr_0 \approx 1-e^{-2^{2n-133}}$, and the advantage $A \approx e^{-2^{2n-168}} - e^{-2^{2n-133}}$.

Our attack needs $2^{n+61}$ Helix block function operations and $2^{n+58}\times(n+58)$ basic comparisons. In order to provide a better estimate of the complexity, we test the time complexities of a Helix block function operation and a basic comparison respectively. The result shows that a basic comparison costs $1/42 (\simeq 2^{-5.4})$ time of a Helix block function operation in average. Thus the time complexity of our attack is $2^{n+52.6} \times (n + 396)$ Helix block function evaluations.

For different modified versions of Helix, the complexities of the attacks are different, and three of them are summarized in table 1.

**Table 1.** Our attacks on different modified versions of Helix

| $n$ | Data | Time | $Pr_0$ | $A$ |
|---|---|---|---|---|
| 66 | $2^{127}$ | $2^{127.45}$ | 0.393 | 0.393 |
| 66.5 | $2^{127.5}$ | $2^{127.95}$ | 0.632 | 0.632 |
| 67.5 | $2^{128.5}$ | $2^{128.95}$ | 0.982 | 0.982 |

## 4.4 Distinguishing Attack on Helix

If we have $n = 62$, we obtain $M = 2^{212}$ and $Pr_0 \approx 1 - e^{-2^{-9}}$. Now we consider how many times of the forcing process we can do, i.e., how many pairs of $(N_0, N_1, N_2, N_3)$ we can use in the no-nonce-reuse model. In fact, for the nonces

$$N^{(\delta, \Delta)} = (N_0 + \delta, N_1 + \delta, N_2 + \delta, N_3 + \Delta)$$

because $\Delta$ spans all $2^{32}$ possible words, we can only have $N_3$ fixed; $\delta$ is of the form $8 \times x$ where $x$ spans all values from 0 to $2^{29} - 1$, so each $N_i (i = 0, 1, 2)$ can take values from the set {0,1,...,7}. Thus we can do $2^9$ times of the forcing process. In this case, the data complexity is $2^9 \times 2^{123} = 2^{132}$, the time complexity is $2^9 \times 2^{n+52.6} \times (n+396) \approx 2^{132.4}$, the success probability $Pr_0 \approx 1 - (e^{-2^{-9}})^{2^9} \approx 0.632$, and the advantage is $A \approx 0.632$.

## 4.5 Key Recovery Attack on Modified Helix

By lemma 1, if we find a collision of the form $S_{1,0,1} = S_{2,r_2,1+j}$, we can obtain a collision of the form $S_{1,0,0} = S_{2,r_2,j}$, where $0 \le r_2 \le 2^{31} - 1, 0 \le j \le 2^{28} - 2$. Furthermore, from Section 2.5 we obtain

$$S_{1,0,0}^{<1>} = Z_0^{(-8)}$$
$$S_{1,0,0}^{<2>} = Z_0^{(-7)}$$
$$\cdots$$
$$S_{1,0,0}^{<8>} = Z_0^{(-1)}$$

where $S_{h,r,t}^{<i>}$ represents the $i$-th word of $S_{h,r,t}$. Then from Section 2.4, we obtain $Z_0^{(-8)} = K_3 \oplus (N_0 + \delta_1)$, and deduce $K_3 = S_{2,r_2,j}^{<1>} \oplus (N_0 + \delta_1)$.

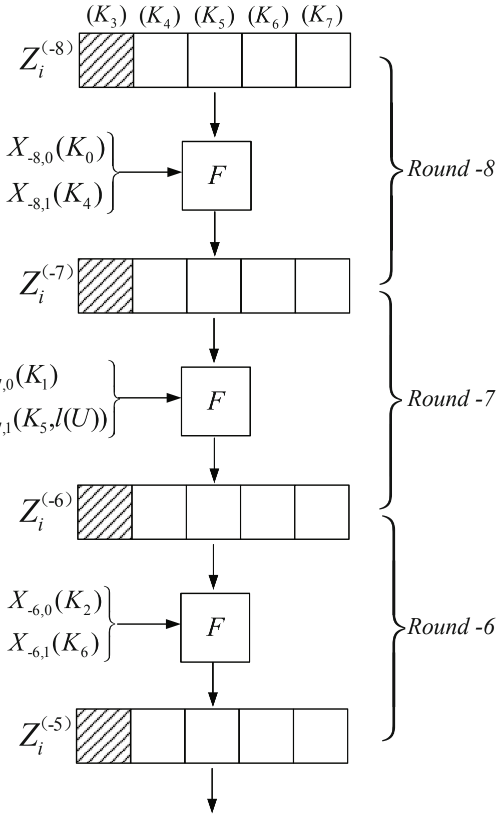


**Fig. 2.** The first three rounds of Helix

We suppose an attacker has access to the keystream. Then we can obtain $S_{2,r_2,j}$, and deduce $Z_0^{(-i)}(i = 1, 2, ..., 8)$(the dashed boxes represented in Fig.2). Now let us consider the round $-8$ of Helix encryption. Because $Z_1^{(-8)}$ and $X_{-8,1}$ have the same working key word $K_4$, there are only 9 unknown state words at round $-8$: $K_4$, $K_5$, $K_6$, $K_7$, $K_0$ and $Z_i^{(-7)}, 1 \leq i \leq 4$. Hence if we guess 4 working key words $K_4$, $K_5$, $K_6$ and $K_7$, we can deduce the remaining 5 unknown words using 5 state equations. Then we consider the round $-7$. There are 6 unknown state words: $K_1$, $l(U)$ and $Z_i^{(-6)}, 1 \leq i \leq 4$, and if $l(U)$ guessed we can deduce the remaining 5 unknown words using 5 state equations. After we obtain the internal state of round $-7$, we can deduce $K_2$ using the 5 state equations of round $-6$. Thus we have recovered all the working key words $K_i$ and the key length $l(U)$. We check the correctness of the guessed $K_4$, $K_5$, $K_6$, $K_7$ and $l(U)$ using the state equations of round -5 to round 0. After we obtain a correct pair of $K_i$ and $l(U)$, we can deduce the input key $U$ by reversing the key mixing of Helix.

The complexity of recovering the input key $U$ by the method above is $(2^{32})^4 \times 256 \times 8 = 2^{139}$ Helix block function operations. In the next work, we will force a collision of the form $S_{1,0,1} = S_{2,r_2,1+j}$ for some $r_2$ and $j$.

We first build a message $P$ of the maximal length $2^n$ words by repeating $2^n$ times the same word $P_0$. Then, $P$ is encrypted under a fixed unknown secret key $K$ and two different nonces $N^{(\delta_1,\Delta_1)}$ and $N^{(\delta_2,\Delta_2)}$, where $\delta_1$ and $\delta_2$ are of the form $8 \times x$ and $x$ spans all values from 0 to $2^{29}-1$, $\Delta_1$ and $\Delta_2$ span all $2^{32}$ possible words. Therefore, the number of blocks encrypted is $2^n \times 2^{29} \times 2^{32} = 2^{n+61}$. To mount an attack, for each position $i$ such that $i \bmod 8 = 0$, we store sequences of 8 consecutive keystream words as a prefix of each entry, followed by the round number $i$, $\delta$ and $\Delta$. Then, we look for a collision among the $\frac{2^{n+61}}{8} = 2^{n+58}$ entries in this table. This can be achieved by sorting the table by prefix of each entry, with a complexity of $2^{n+58} \times (n+58)$ basic comparisons. When we find a collision, we need to check whether the collision is of the form $S_{1,0,1} = S_{2,r_2,1+j}$. This can be achieved by checking the round numbers in the two entries: if there is a round number 0, this collision is of the form $S_{1,0,1} = S_{2,r_2,1+j}$, otherwise, it is not. The number of independent pairs of the form $S_{1,0,1} = S_{2,r_2,1+j}$ in the table is

$$M = \frac{2^{61} \times 2^{61}}{2} \times (2^{n-3} + 2^{n-3}) = 2^{n+119},$$

the success probability is $Pr_0 \approx 1 - e^{-2^{n-102}}$, and the advantage is $A \approx e^{-2^{n-137}} - e^{-2^{n-102}}$. After we find a collision of the form $S_{1,0,1} = S_{2,r_2,j}$, we can recover the key $U$ with the method above. This attack needs $2^{n+61}$ data, and the time complexity is $2^{n+61} + 2^{n+58} \times (n+58) \times 2^{-5.4} + 2^{139} \simeq 2^{n+52.6} \times (n+396)$ Helix block function evaluations. If we have $n = 102$, our attack would success with probability $Pr_0 \approx 0.632$ and needs a data of $2^{163}$.

The following problem remains to be solved: how to find a true collision of Helix and a special true collision of the form $S_{1,0,1} = S_{2,r_2,j}$ with plaintext restricted by the maximal authorized length of $2^{62}$ and "fewer than $2^{128}$ Helix block function evaluations to be carried out".

## 4.6 Distinguishing Attack on Phelix

In this section, we extend the distinguishing attack to Phelix, which reveals that Phelix has the same security level as Helix with respect to the state collision attacks.

The only difference between Helix and Phelix is the output of the keystream word (see Section 2.5). As the authors of Phelix declared, to avoid Muller's second attack, Phelix added the 4 "old" state words.

We first build a message $P$ by repeating $2^n$ times the same word $P_0$. Then, $P$ is encrypted by Helix and Phelix cipher respectively, under a fixed unknown secret key $K$ using two different nonces $N^{(\delta_1,\Delta_1)}$ and $N^{(\delta_2,\Delta_2)}$, and denote the four keystream by $S_1, S_2, S'_1, S'_2$.

**Lemma 2.** *For $0 \le r_1, r_2 \le 2^{31}-1$ and $0 \le j \le 2^{28}-2$, if $S_{1,r_1,0}$ and $S_{2,r_2,j}$ is a true collision of Helix, $S'_{1,r_1,1}$ and $S'_{2,r_2,j+1}$ is a true collision of Phelix.*

*Proof.* Suppose $S_{1,r_1,0}$ and $S_{2,r_2,j}$ is a true collision of Helix, that means there is a state collision at positions $i_1(r_1,0)$ and $i_2(r_2,j)$, and this state collision can be preserved for $2^{31}-8j$ rounds by the proof of Lemma 1. Thus for Phelix there is also a state collision at positions $i_1(r_1,0)$ and $i_2(r_2,j)$, and this state collision can be preserved for $2^{31}-8j$ rounds. From Section 2.5, the output of Phelix is $s_i := Y_4^{(i)} + Z_4^{(i-4)}$, so $S'_{1,r_1,1}$ is determined only by the states from round $i_1(r_1,1)-4$ to $i_1(r_1,1)+7$, and $S'_{2,r_2,j+1}$ is determined only by the states from round $i_2(r_2,j+1)-4$ to $i_2(r_2,j+1)+7$. So $S'_{1,r_1,1}$ and $S'_{2,r_2,j+1}$ is a true collision of Phelix. □

Thus we can easily extend the distinguishing attack to Phelix, with the same time and data complexities. However, Phelix can resist our key recover attack on Helix. Because even if we find a collision of the form $S'_{1,0,1}=S'_{2,r_2,1+j}$, we can't obtain a collision of the form $S'_{1,0,0}=S'_{2,r_2,j}$ due to $Z_4^{(i)}=0$ for $i=-12,...,-9$.

### 4.7 Cryptanalysis of a General Type of Enhanced Version of Helix

Now we consider a general type of enhanced version of Helix by adding an output function. If the output function at round $i$ takes as input some states from round $i-n$ to round $i$, we denote this type of enhanced version $n$-Helix. For example, the output function of Phelix at round $i$ takes as input states from round $i-4$ to round $i$, so Phelix is an instance of 4-Helix.

**Lemma 3.** *For* $0 \le r_1, r_2 \le 2^{31}-1$ *and* $0 \le j \le 2^{28}-1-m$*, if* $S_{1,r_1,0}$ *and* $S_{2,r_2,j}$ *is a true collision of Helix,* $S'_{1,r_1,m}$ *and* $S'_{2,r_2,j+m}$ *is a true collision of any instance of n-Helix, where* $m=\lceil \frac{n}{8} \rceil$*.*

*Proof.* Suppose $S_{1,r_1,0}$ and $S_{2,r_2,j}$ is a true collision of Helix, that means there is a state collision at positions $i_1(r_1,0)$ and $i_2(r_2,j)$, and this state collision can be preserved for $2^{31}-8j$ rounds by the proof of Lemma 1. Thus for any instance of $n$-Helix there is also a state collision at positions $i_1(r_1,0)$ and $i_2(r_2,j)$, and this state collision can be preserved for $2^{31}-8j$ rounds. By the definition of $n$-Helix, $S'_{1,r_1,m}$ is determined only by the states from round $i_1(r_1,m)-n$ to $i_1(r_1,m)+7$, and $S'_{2,r_2,j+m}$ is determined only by the states from round $i_2(r_2,j+m)-n$ to $i_2(r_2,j+m)+7$. So $S'_{1,r_1,m}$ and $S'_{2,r_2,j+m}$ is a true collision. □

Thus, if $m$ is much smaller than $2^{28}$, we can extend our distinguishing attack to any instance of $n$-Helix with the same time and data complexities. Therefore, such type of enhanced version of Helix has the same security level as Helix with respect to state collision attacks. Furthermore, our key recovery attack on Helix can be extended on any instance of 0-Helix, because if we find a collision of the form $S'_{1,0,1}=S'_{2,r_2,1+j}$, we can obtain a collision of the form $S'_{1,0,0}=S'_{2,r_2,j}$.

## 5 Conclusion

In this paper, we have studied the security of Helix and Phelix against both distinguishing and key recovery attacks. First, a flaw in Muller's second attack on Helix has been identified, which results in the failure of Muller's second attack. Based on this finding, we propose two attacks on Helix without nonce reused. Then we show that our distinguishing attack on Helix can be easily extended to Phelix with almost the same data complexity and time complexity, which reveals that Phelix has the same security level as Helix with respect to internal state collision attacks. This is the first distinguishing attack on Phelix without nonces reused. Furthermore, a new insight on the design of such dedicated ciphers with built-in authentication has been discovered, which we hope to be helpful in the design of dedicated authenticated encryption algorithms.

**Acknowledgement.** The authors gratefully acknowledge the anonymous referees, whose comments helped to improve the presentation.

## References

1. Ferguson, N., Whiting, D., Schneier, B., Kelsey, J., Lucks, S., Kohno, T.: Helix: Fast Encryption and Authentication in a Single Cryptographic Primitive. In: Johansson, T. (ed.) FSE 2003. LNCS, vol. 2887, pp. 330–346. Springer, Heidelberg (2003)
2. Whiting, D., Schneier, B., Lucks, S., Muller, F.: Phelix: Fast Encryption and Authentication in a Single Cryptographic Primitive. Technical Report 2005/027. In eSTREAM, ECRYPT Stream Cipher Project (2005)
3. Bernstein, D.J.: Salsa20. Technical Report 2005/025. In eSTREAM, ECRYPT Stream Cipher Project (2005)
4. Muller, F.: Differential Attacks against the Helix Stream Cipher. In: Roy, B., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 94–108. Springer, Heidelberg (2004)
5. Paul, S., Preneel, B.: Near Optimal Algorithms for Solving Differential Equations of Addition with Batch Queries. In: Maitra, S., Veni Madhavan, C.E., Venkatesan, R. (eds.) INDOCRYPT 2005. LNCS, vol. 3797, pp. 90–103. Springer, Heidelberg (2005)
6. Paul, S., Preneel, B.: Solving Systems of Differential Equations of Addition. In: Boyd, C., González Nieto, J.M. (eds.) ACISP 2005. LNCS, vol. 3574, pp. 75–88. Springer, Heidelberg (2005)
7. Wu, H., Preneel, B.: Differential-Linear Attacks Against the Stream Cipher Phelix. In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 87–100. Springer, Heidelberg (2007)
8. Phorum eStream. Key recovery attacks on Phelix (2006), `http://www.ecrypt.eu.org/stream/phorum/read.php?1,883,921`
9. eSTREAM. ECRYPT stream cipher project, `http://www.ecrypt.eu.org/stream/portfolio.pdf`
10. DIAC. Directions in Authenticated Ciphers, `http://hyperelliptic.org/DIAC/`

# Attacks on Multi-Prime RSA with Small Prime Difference

Hui Zhang[1] and Tsuyoshi Takagi[2]

[1] Graduate School of Mathematics, Kyushu University, Japan
h-zhang@math.kyushu-u.ac.jp
[2] Institute of Mathematics for Industry, Kyushu University, Japan
takagi@imi.kyushu-u.ac.jp

**Abstract.** We consider some attacks on multi-prime RSA (MPRSA) with a modulus $N = p_1p_2 \dots p_r$ $(r \geq 3)$. It is believed that the small private exponent attack on the MPRSA is less effective than that on RSA (see Hinek et al.'s work at SAC 2003), which means that one can use a smaller private exponent in the MPRSA than that in the original RSA. However, our attacks show that private exponents which are significantly beyond Hinek's bound may be insecure when the prime difference $\Delta$ ($\Delta = p_r - p_1 = N^{\gamma}$, $0 < \gamma < 1/r$, suppose $p_1 < p_2 < \cdots < p_r$) is small. By exploring the relation between $\phi(N)$ and its upper bound, our proposed small private exponent attack can make full use of the benefit brought by small prime difference. It is shown that the MPRSA is insecure when $\delta < 1 - \sqrt{1 + \gamma - 2/r}$, where $\delta$ is the exponential of the private exponent $d$ with base $N$, i.e., $d = N^{\delta}$. This result is a perfect extension of the best known small private exponent attack. We also present a Fermat-like factoring attack on the MPRSA which can directly factor the modulus $N$ when $\Delta < N^{1/r^2}$. These results surpass those of Bahig et al. (ICICS 2012) and the attacks are experimentally proved effective in practice.

**Keywords:** multi-prime RSA, lattice, cryptanalysis, small prime difference.

## 1 Introduction

The RSA cryptosystem [16] is one of the most important public-key cryptosystems that has been widely used in the secure Web communication (such as the SSL protocol and the TLS protocol). The original version of the RSA is described as follows.

**The RSA Cryptosystem:**

- Let $N$ be the product of two large primes $p$ and $q$. The public exponent $e$ and the private exponent $d$ satisfy $ed \equiv 1 \mod \phi(N)$, where $\phi(N) = (p-1)(q-1)$ is Euler's totient function. The public key is the pair $(N, e)$ and the private key is $d$.
- Encryption algorithm: $c = m^e \mod N$ where $m \in \mathbb{Z}_N$ is the plaintext, $c$ is the ciphertext.

– Decryption algorithm: $m = c^d \mod N$.

Multi-prime RSA is a variant of the original RSA in which the modulus has more than two distinct primes. The key generation algorithm in the MPRSA is essentially the same as that in the RSA, except that the modulus requires $r$ ($r \geq 3$) distinct primes instead of two, i.e., $N = p_1 p_2 \cdots p_r$. Consequently, $\phi(N) = \Pi(p_i - 1)$. The encryption algorithm and the decryption algorithm are identical with those of the original RSA.

In this paper we only consider the MPRSA with balanced primes, i.e., the primes $p_i$'s are of the same bit size. If we assume $p_i$'s are labeled in ascending order, i.e., $p_1 < p_2 < \cdots < p_r$, then we have

$$\frac{1}{2}N^{1/r} < p_1 < N^{1/r} < p_r < 2N^{1/r}.$$

The prime difference of MPRSA is defined as $\Delta = p_r - p_1 = N^{\gamma}$ where $0 < \gamma < 1/r$.

The main advantage of the MPRSA is its efficiency in decryption. Specifically, when Chinese remainder theorem is used in decryption, the main costs will be $r$ modular exponentiations with $(n/r)$-bit moduli, where $n$ is the bit length of $N$. Compared to 2 modular exponentiations with $(n/2)$-bit moduli in the RSA, this leads to a theoretic speed-up of a factor by up to $r^2/4$. Boneh and Shacham [4] experimentally observed a speed-up by a factor of 1.73 for 3-MPRSA (the modulus has 3 prime factors) with a 1024-bit modulus. Moreover, most mathematical attacks (such as the small exponent attacks, partial key exposure attack etc.) become less effective as $r$ increases. (However, $r$ can not be unlimitedly large because for the elliptic curve factorization method the difficulty of factoring an MPRSA modulus decreases with increasing $r$. In most cases, $r$ is set 3,4 or 5. Refer to [7] for more details.) Therefore, MPRSA might be a practical alternative to RSA when decryption costs need to be lowered. Now the MPRSA has already been supported by PKCS #1 v2.1 [17] and COMPAQ company [6,7].

Obviously, using small private exponents can also decrease the decryption costs. However, too small private exponents may render the system completely insecure. Wiener [20] first showed that RSA is insecure if $d < N^{1/4}$. Boneh and Durfee [3] improved this result to $d < N^{0.292}$ by using lattice reduction method. They also conjectured that the right bound below which RSA is insecure might be $d < N^{1/2}$. But this conjecture has not been proved yet. Let $\Delta = |p-q|$ be the prime difference of the original RSA. De Weger [19] found that the small private exponent attacks (both Wiener's and Boneh & Durfee's) can be enhanced when $\Delta$ is small and the insecure bound can be improved to $d < N^{1/2}$, even to $d < N$. It is common knowledge that Fermat's factoring technique [19] makes RSA insecure when the prime difference is too small ($\Delta < N^{1/4}$). To resist this attack, standards, such as ANSI X9.31 [1] and FIPS 168-3 [15] require that the two primes should differ in the first 100 bits, i.e., for 1024-bit RSA, $\Delta$ should be larger than $N^{0.402}$. De Weger's work showed that though $\Delta$ is up to this standard, it still can lead to an enhanced small private exponent attack as long as it is smaller than the general size (in the original RSA, the general size of $\Delta$ is $N^{1/2}$).

Ciet et al. [5] generalized Wiener's as well as Boneh & Durfee's small private exponent attack to the MPRSA. Bahig et al. [2] generalized de Weger's result and showed that the small prime difference can also enhance the small private exponent attack on the MPRSA. To be specific, in the case of 3-MPRSA, Ciet et al.'s insecure bound is about $d < N^{0.18}$; Bahig et al. observed that the insecure $d$ could be as large as $N^{1/3}$.

**Our Contribution.** To the best of our knowledge, Bahig et al.'s work [2] is the only one which studies on the security of MPRSA with small prime difference. However, compared with de Weger's research on the original RSA, their work is just a glimpse into this topic from the perspective of the continued fraction technique. In this paper, we show our comprehensive studies which include not only the small private exponent attack but also a factoring attack. We found that the attacks based on the lattice reduction techinique perform much stronger than that based on the continued fraction technique.

– In the respect of attacking the MPRSA with small private exponent, we give our lattice reduction based attack with a bound $\delta < 1 - \sqrt{1 + \gamma - \frac{2}{r}}$, where $\delta = \log_N d$ and $\gamma = \log_N \Delta$, which is superior to Bahig et al.'s result [2]. Especially, if the prime difference $\Delta$ is of general size, i.e., $\Delta = N^{1/r}$, then the condition for our attack becomes $\delta < 1 - \sqrt{1 - \frac{1}{r}}$ which exactly coincides with Ciet et al.'s best result [5] on attacking the MPRSA with small private exponent. Thus, the best known small private exponent attack on the MPRSA can be regarded as a special case of our attack.
– In [2], the authors presented that when extending de Weger's attacks to MPRSA, they also tried to generalize the Fermat's factoring attack but failed. In this paper, we present a factoring attack on the MPRSA which can directly factor the modulus $N$ in polynomial time in $\log(N)$ when the prime difference is smaller than $N^{1/r^2}$. When $r = 2$, the condition of this attack becomes $\Delta \leq N^{1/4}$ which means that it performs as strong as the Fermat's factoring attack on the original RSA.

We compare our attacks with the previous works, including Ciet et al.'s attack [5] (bound Eq. (4) in Section 3) which is the strongest one among all the previous small private exponent attacks on MPRSA and Bahig et al.'s attack [2] (bound Eq. (5) in Section 3) which is the first/only attack on MPRSA with small prime difference.

Figure 1 shows the respective comparisons in terms of $r = 3$. The shaded areas are the insecure $\delta, \gamma$ pairings. Obviously, our small private exponent attack is superior to others. Our Fermat-like factoring attack further demonstrates the vulnerability of the MPRSA when the prime difference is extremely small. It should be noted that there will be less insecure area as $r$ increases, which means all these attacks become less effective as $r$ increases. However, for any given $r$, our small private exponent attack is always superior to the other attacks.

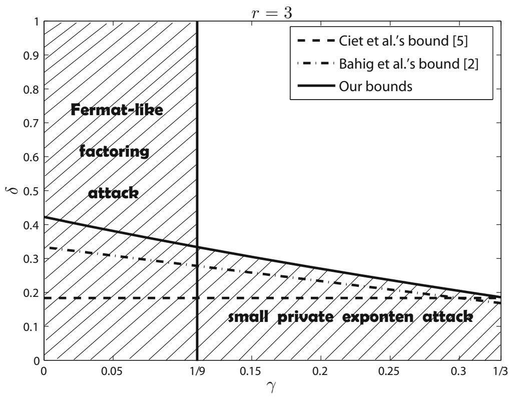


**Fig. 1.** Comparison between our attacks and previous ones on 3-MPRSA

Our work proves that small prime difference is also a vulnerable feature for the MPRSA. It is necessary to check the primes generated in the key generation step of the MPRSA, especially for the cases of using small private exponents.

## 2 Previous Attacks on Original RSA

In this section we outline several previous attacks on the original RSA that are relevant to our work.

### 2.1 Wiener's Small Private Exponent Attack

Wiener's continued fraction attack is the first significant attack on the small private exponent RSA. As the public exponent $e$ and the private exponent $d$ satisfy $ed \equiv 1 \mod \phi(N)$, there exists some integer $k$ which satisfies

$$ed = 1 + k\phi(N). \tag{1}$$

Eq. (1) can be rewritten as

$$\frac{e}{\phi(N)} - \frac{k}{d} = \frac{1}{d\phi(N)}.$$

If $N$ is a balanced modulus, then $\phi(N)$ is relatively close to $N$. It can be proved that when $d$ is small, the publicly known fraction $e/N$ is a good approximation to the secret fraction $k/d$. Hence $k/d$ can be found from the convergents in the continued fraction expansion of $e/N$. Wiener showed that given only the public key $(N, e)$, this attack works well when $d < N^{1/4}$. (We refer readers to Wiener's original paper [20] for more details.)

## 2.2 Boneh & Durfee's Small Private Exponent Attack

Boneh and Durfee proposed a much stronger attack which is based on Coppersmith's methods [8] of finding small solutions of modular/integer equations. Using the LLL reduction algorithm [13], Coppersmith found a way of constructing new polynomials, with the same solutions as the target one, which have sufficiently small coefficients. And then, the small solutions can be efficiently found simply by solving the new constructed polynomials over integer. For more information, we refer the reader to Coppersmith's original paper [8] and other related works [11,3].

Let $s = (p + q - 1)$. Then, $\phi(N) = (p-1)(q-1) = N - s$. Therefore, Eq. (1) can be rewritten as

$$-k(N-s) \equiv 1 \mod e \tag{2}$$

Boneh and Durfee showed that solving Eq. (2) for the unknowns $k$ and $s$ leads to a heuristic small private exponent attack on RSA with $d < N^{0.292}$. It should be noted that this attack relies on the assumption that the first few small polynomials obtained by lattice reduction are algebraically independent (which allows a system of polynomials to be solved over $\mathbb{Z}$). This assumption is often claimed that this assumption is valid in practice [3,9].

## 2.3 Fermat's Factoring Attack

It is well known that RSA modulus with small prime difference is insecure due to Fermat's factoring method. Let $N = pq$ be a balanced RSA modulus and $p < q$. The prime difference is $\Delta = q - p \leq N^{1/2}$. Fermat's factoring method is to find positive integers $x, y$ (other than $N+1$ and $N-1$) such that $x^2 - y^2 = 4N$. If succeed, then let $p' = \frac{1}{2}(x+y)$ and $q' = \frac{1}{2}(x-y)$ which must satisfy $p'q' = N$. To find such $x, y$ we simply try $x = \lceil 2N^{1/2} \rceil, \lceil 2N^{1/2} \rceil + 1, \ldots,$ until $x^2 - 4N$ is a square. It can be proved that when $\Delta < cN^{1/4}$, the number of values for $x$ that have to be tried is at most $\frac{c^2}{4}$. Therefore, when $c$ is a small constant, factoring $N$ is trivial.

## 2.4 De Weger's Improvements on the Small Private Exponent Attacks

De Weger [19] found that both Wiener's and Boneh & Durfee's small private exponent attack can be improved when the prime difference is small. Let $\Delta = N^\gamma$ and $d = N^\delta$. Wiener's bound was improved from $\delta < \frac{1}{4}$ to $\delta < \frac{3}{4} - \gamma$, $0 < \gamma < \frac{1}{2}$. Boneh & Durfee's bound was improved from $\delta < 0.292$ to $\delta < 1 - \sqrt{2\gamma - \frac{1}{2}}$, $0 < \gamma < \frac{1}{2}$ (which is an amended result by [12]). We cite de Weger's figure to illustrate the results of above attacks.

From Figure 2 we can see that the RSA cryptosystem becomes more vulnerable to de Weger's small private exponent attack when prime difference is small. To avoid such attacks, de Weger recommended to build in a check for prime difference in the implementation of RSA key generation, especially in the cases of using small private exponents.

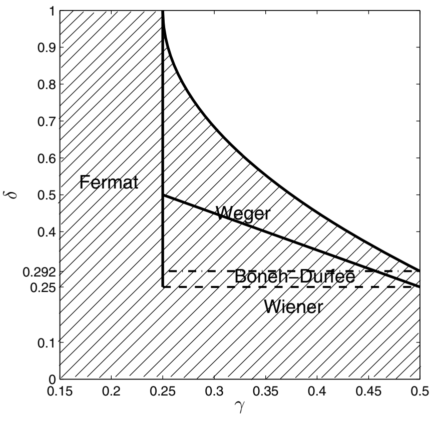


**Fig. 2.** Regions for $\delta$ and $\gamma$ for which RSA is shown to be insecure [19]

## 3 Previous Attacks on MPRSA

In this section we recall several relevant attacks on MPRSA which are mainly extensions of the attacks in Section 2.

### 3.1 Ciet et al.'s Extensions of the Small Private Exponent Attacks

Both Wiener's and Boneh & Durfee's attacks on original RSA were extended to MPRSA by Ciet et al. [5] and then surveyed by Hinek et al. in their published work [10]. Here we give a short overview of these results.

The extension of Wiener's attack (based on continued fraction technique) to MPRSA and yields following result.

– Let $N$ be an $r$-prime RSA modulus with balanced primes, let $e = N^{\alpha}$ be a valid public exponent and $d = N^{\delta}$ be its corresponding private exponent. Given the public key, if
$$\delta \leq \frac{1}{2r}, \tag{3}$$
then the modulus can be factored in time polynomial in $\log N$.

In terms of the bounds, this result is a weak one among all small private exponent attacks on MPRSA. The attack based on lattice reduction technique can get a larger bound which is also the strongest known small private exponent attack on MPRSA. The result is as follows.

– Let $N$ be an $r$-prime RSA modulus with balanced primes, let $e = N^{\alpha}$ be a valid public exponent and $d = N^{\delta}$ be its corresponding private exponent. Given the public key, if
$$\delta \leq \frac{r - \sqrt{r(r-1)}}{r}, \tag{4}$$
then the modulus can be factored in time polynomial in $\log N$.

### 3.2 Bahig et al.'s Extension of de Weger's Attack

Bahig et al. [2] extended de Weger's attack to MPRSA. More specifically, they extended the continued fraction based small exponent attack on MPRSA to the case of modulus $N$ with small prime difference and got similar improvement as de Weger's work.

– Let $N = p_1 p_2 \cdots p_r$ be balanced MPRSA modulus, $p_r - p_1 = N^{\gamma}$. If the private exponent $d$ satisfies $2d^2 + 1 < \frac{n^{2/r-\gamma}}{6r}$, then the modulus can be factored in time polynomial in $\log N$.

Let $d = N^{\delta}$. Bahig et al.'s bound can be rewritten in a similar form as (3) and (4) as follows.

$$\delta \leq \frac{1}{r} - \frac{\gamma}{2}. \tag{5}$$

## 4 The Proposed Attacks

In this section, we introduce our attacks on the MPRSA. The first one is a small private exponent attack which is a generalization of de Weger's attack based on lattice reduction technique. The second one is also based on the lattice reduction technique which can efficiently factor the modulus when the prime difference is extremely small.

### 4.1 Small Private Exponent Attack

We first introduce a framework to show how to recover the private exponent when it is sufficiently small. Recall that the private exponent $d$ and the public exponent $e$ of the MPRSA satisfy $ed \equiv 1 \mod \phi(N)$, where $\phi(N) = \prod_{i=1}^{r}(p_i - 1)$. It follows that there exists an integer $k$ such that

$$ed = 1 + k\phi(N) = 1 + k(N - \sum_{i=1}^{r} \frac{N}{p_i} + \sum_{\substack{i,j=1 \\ i<j}}^{r} \frac{N}{p_i p_j} - \cdots + (-1)^r). \tag{6}$$

Let $s = N - \phi(N) = \sum\limits_{i=1}^{r} \frac{N}{p_i} - \sum\limits_{\substack{i,j=1 \\ i<j}}^{r} \frac{N}{p_i p_j} + \cdots - (-1)^r$. Eq. (6) can be rewritten as $k(N - s) = 1 \mod e$ in which $k$ and $s$ are unknown. If we can find $s$, then we can obtain $\phi(N)$ as well as the private exponent $d$.

As mentioned in Section 2, Boneh & Durfee [3] showed their observation on the original RSA ($r = 2$). They also generalized this problem, which is called small inverse problem, as follows.

Given two integers $A$ and $B$, the small inverse problem is to find an integer close to $A$ such that its inverse modulo $B$ is small, i.e., to find small integers $x$ and $y$ satisfying $x(A + y) \equiv 1 \mod B$.

Herrmann & May [9] also presented a method to construct optimized lattices used in the small private exponent attack. Their method got the same bound ($\delta < 1 - \frac{1}{2}\sqrt{2} \approx 0.292$) as Boneh & Durfee's but with a much simpler proof.

Our attack is actually a direct application of solving the small inverse problem. However, in the previous works [3,9], the authors only considered the case when the bound of $y$ is fixed, i.e., $y < B^{1/2}$, and try to maximize the bound of $x$. In the following, we introduce a generalized result of the small inverse problem which can be derived from the previous works with small changes.

**Theorem 1.** *Given large integers $A$ and $B$, let $X = B^{\alpha}$ and $Y = B^{\beta}$ where $0 < \alpha, \beta < 1$ satisfy*

$$\alpha \leq 1 - \sqrt{\beta}. \tag{7}$$

*Then we can probabilistically find all solutions $(x_0, y_0)$ of $x(A + y) \equiv 1 \mod B$ with $|x_0| < X$ and $|y_0| < Y$ in polynomial time.*

For completeness, we provide a full description of the proof which is based on Herrmann & May's method in Appendix.

Obviously, the above framework does not take the size of the prime difference $\Delta$ into account. Actually, small $\Delta$ can enhance the small private exponent attack if this feature can be fully exploited. Though $s$ is unknown, we find its lower bound can be obtained. In our work, we first calculate the lower bound of $s$ and then we estimate $s$ with this lower bound in the attack. The new unknown now is replaced with the error of the estimate. We prove that this new unknown is smaller than $s$ when $\Delta$ is small. That means we can solve the small inverse problem with larger $k$ and thus cryptanalyze the MPRSA with larger $d$. To detail our attack, some lemmas are necessary.

**Lemma 1.** *Let $N$ be a large positive integer, $r$ be a small positive integer, $D^{(i)}$ be an $i$ dimensional domain defined as $D^{(i)} = \{(x_1, x_2, \ldots, x_i) : (x_1, x_2, \ldots, x_i) \in \mathbb{R}^i, N^{1/r}/2 < x_j < 2N^{1/r} \text{ for } j = 1, 2, \ldots, i\}$ and $R = \{1, 2, \ldots, r\}$. For $k = 1, 2, \ldots, r - 1$, define the functions $f_k(x_1, x_2, \ldots, x_r)$ as*

$$f_k = \sum_{\substack{\{i_1,\ldots,i_k\} \subset R \\ i_1 < \cdots < i_k}} x_{i_1} x_{i_2} \ldots x_{i_k}.$$

*In the domain $D^{(r)}$, if $x_i$'s satisfy $x_1 x_2 \cdots x_r = N$, then*
**i.** *$f_k(x_1, x_2, \ldots, x_r) \geq C_k^r N^{k/r}$, where $C_k^r = \frac{r!}{k!(r-k)!}$ is the binomial coefficient.*
**ii.** *$\Delta_{j+1} - \Delta_j \geq 0$ for $j = 1, \ldots, r - 2$, where $\Delta_j = f_j(x_1, \ldots, x_r) - C_j^r N^{j/r}$.*

*Proof.* **i.** As $x_1 x_2 \cdots x_r = N$ and $x_i \neq 0$ for $i = 1, 2, \ldots, r$, we have $x_r = \frac{N}{x_1 x_2 \cdots x_{r-1}}$. Then $f_k$ with constraint $x_1 x_2 \cdots x_r = N$ can be rewritten as

$$\tilde{f}_k(x_1 x_2 \cdots x_{r-1}) = \sum_{\substack{\{i_1,\ldots,i_k\} \subset R' \\ i_1 < \cdots < i_k}} x_{i_1} x_{i_2} \ldots x_{i_k} + \sum_{\substack{\{i_1,\ldots,i_{k-1}\} \subset R' \\ i_1 < \cdots < i_{k-1}}} \frac{N}{x_{i_1} \ldots x_{i_{k-1}}},$$

where $R' = \{1, 2, \ldots, r - 1\}$. Now we compute the extremum of $\tilde{f}_k$ in $D^{(r-1)}$.

For each $k$, partial derivatives of $\tilde{f}_k$ are

$$\frac{\partial \tilde{f}_k}{\partial x_i} = \sum_{\substack{\{i_1,\dots,i_{k-1}\}\subset R'/\{i\} \\ i_1<\dots<i_{k-1}}} x_{i_1}x_{i_2}\dots x_{i_{k-1}} - \sum_{\substack{\{i_1,\dots,i_{k-2}\}\subset R'/\{i\} \\ i_1<\dots<i_{k-2}}} \frac{N}{x_{i_1}\dots x_{i_{k-2}}\cdot x_i^2}$$

By solving the simultaneous equations $\frac{\partial \tilde{f}_k}{\partial x_i} = 0$, for $i = 1, \dots, r-1$ we get that $(N^{1/r}, N^{1/r}, \dots, N^{1/r})$ is the only stationary point of function $\tilde{f}_k$ in domain $D^{(r-1)}$.

Let $X_0 = (N^{1/r}, N^{1/r}, \dots, N^{1/r})$. The Hessian matrix of $\tilde{f}_k$ at the point of $X_0$, i.e., $\left(\frac{\partial^2 \tilde{f}_k}{\partial x_i \partial x_j}(X_0)\right)_{1\le i,j\le r-1}$ can be easily proved to be a positive definite matrix. Therefore, $X_0$ is a minimum point of $\tilde{f}_k$. That means, with the constraint condition $x_1x_2\cdots x_r = N$ and $(x_1, \dots, x_r) \in D^{(r)}$, $f_k$ reaches its minimum $C_k^r N^{k/r}$ at the point $(N^{1/r}, N^{1/r}, \dots, N^{1/r})$.

**ii.** The same goes for the function $f_{j+1} - f_j$; we can prove that $f_{j+1} - f_j$ reaches its minimum $C_{j+1}^r N^{(j+1)/r} - C_j^r N^{j/r}$ in domain $D^{(r)}$ with the constraint condition that $x_1x_2\cdots x_r = N$. Therefore, $\Delta_{j+1} - \Delta_j \ge 0$. □

**Lemma 2.** *Let $N = p_1p_2\dots p_r$ be a balanced MPRSA modulus, $p_1 < p_2 < \dots < p_r$, $p_r - p_1 = N^\gamma$, $0 < \gamma < 1/r$. Let $s_{r-1} = \sum_{i=1}^{r} \frac{N}{p_i}$ and $E_{r-1} = rN^{1-1/r}$. Then*

$$0 < s_{r-1} - E_{r-1} < 2rN^{1+\gamma-\frac{2}{r}}$$

*Proof.* According to Lemma 1, we have $s_{r-1} = f_{r-1}(p_1, p_2, \dots, p_r) > rN^{1-1/r} = E_{r-1}$. Therefore $s_{r-1} - E_{r-1} > 0$.

$$\begin{aligned} s_{r-1} - E_{r-1} &= \sum_{i=1}^{r}\left(\frac{N}{p_i} - \frac{N}{N^{1/r}}\right) = \sum_{i=1}^{r} \frac{N(N^{1/r} - p_i)}{N^{1/r}p_i} \\ &< r\cdot\frac{N(p_r - p_1)}{\frac{1}{2}N^{2/r}} = 2rN^{1+\gamma-\frac{2}{r}}. \end{aligned}$$

□

**Proposition 1.** *Let $E_k = C_k^r N^{k/r}$ and $s_k = \sum_{\substack{\{i_1,\dots,i_k\}\subset R \\ i_1<\dots<i_k}} p_{i_1}p_{i_2}\cdots p_{i_k}$, where $R = \{1, 2, \dots, r\}$. Consequently $s = N - \phi(N) = \sum_{i=1}^{r} \frac{N}{p_i} - \sum_{\substack{i,j=1 \\ i<j}}^{r} \frac{N}{p_ip_j} + \dots - (-1)^r = s_{r-1} - s_{r-2} + \dots - (-1)^r$. Let $E = E_{r-1} - E_{r-2} + \dots - (-1)^r$ be our estimation of $s$. Then the estimation error $\mathcal{E} = s - E$ satisfies*

$$0 < \mathcal{E} < 2rN^{1+\gamma-\frac{2}{r}}.$$

*Proof.* Let $\mathcal{E}_i = s_i - E_i$ for $i = 1, 2, \dots, r-1$. According to Lemma 1, we have

$$s_i = f_i(p_1, p_2, \dots, p_r) > C_i^r N^{i/r} = E_i \Rightarrow \mathcal{E}_i > 0$$

for $i = 1, 2, \ldots, r-1$ and

$$s_i - E_i > s_{i-1} - E_{i-1} \Rightarrow \mathcal{E}_i > \mathcal{E}_{i-1}$$

for $i = 2, \ldots, r-1$. Accordingly, we get $\sum_{i=1}^{k}(-1)^{k-i}\mathcal{E}_i > 0$ and $\mathcal{E}_{k+1} > \sum_{i=1}^{k}(-1)^{k-i}\mathcal{E}_i$ for $k = 1, 2, \ldots, r-2$. Therefore,

$$0 < \mathcal{E} < \mathcal{E}_{r-1}$$

since

$$\mathcal{E} = \sum_{i=1}^{r-1}(-1)^{r+1-i}\mathcal{E}_i = \mathcal{E}_{r-1} - \sum_{i=1}^{r-2}(-1)^{r-i}\mathcal{E}_i.$$

By Lemma 2, we have $0 < s - e < 2rN^{1+\gamma-\frac{2}{r}}$. □

**Theorem 2 (Small Private Exponent Attack).** *Let $N = p_1 p_2 \ldots p_r$ be an MPRSA balanced modulus, $p_1 < p_2 < \cdots < p_r$, $p_r - p_1 = N^{\gamma}$, $0 < \gamma < 1/r$. Let $e$ be a public exponent with full size and $d = N^{\delta}$, $0 < \delta < 1$, be its corresponding private key. Given the public key $(N, e)$, for every integer $r \geq 3$, $\gamma$ and $\delta$ satisfy*

$$\delta < 1 - \sqrt{1 + \gamma - \frac{2}{r}}, \tag{8}$$

*then the private key $d$ can be probabilistically found in time polynomial in log$(N)$.*

*Proof.* Let $s = \sum\limits_{i=1}^{r} \frac{N}{p_i} - \sum\limits_{\substack{i,j=1 \\ i<j}}^{r} \frac{N}{p_i p_j} + \cdots - (-1)^r$, $E = \lfloor \sum\limits_{i=1}^{r-1}(-1)^{r-1-i}C_i^r N^{i/r} - (-1)^r \rfloor$ and $\mathcal{E} = s - E$. Then equation $ed = 1 \mod \phi(N)$ can be rewritten as $k\phi(N) = 1 \mod e$ for some integer $k$. As $\phi(N) = N - s = N - E - \mathcal{E}$, we have

$$k(N - E - \mathcal{E}) = 1 \mod e. \tag{9}$$

We assume $e$ is full size, i.e., $e$ is roughly the same order of magnitude as the modulus $N$ and $\phi(N)$. As

$$k = \frac{ed-1}{\phi(N)} < \frac{ed}{\phi(N)}, \text{ and } d < N^{\delta},$$

we can get a rough bound of $k$ that $|k| < e^{\delta}$. By Proposition 1, we also have $|\mathcal{E}| < 2rN^{1+\gamma-\frac{2}{r}} \approx e^{1+\gamma-\frac{2}{r}}$.

Therefore, finding $k$ and $\mathcal{E}$ from Eq. (9) is a small inverse problem in which the large known integers are $N - E$ and $e$, and the bound of the unknowns are respectively $|k| < e^{\delta}$ and $|\mathcal{E}| < e^{1+\gamma-\frac{2}{r}}$. According to Theorem 1, when $\gamma$ and $\delta$ satisfy

$$\delta < 1 - \sqrt{1 + \gamma - \frac{2}{r}},$$

$k$ and $\mathcal{E}$ can be probabilistically found in polynomial time in log$(N)$.

Then we can get $\phi(N) = N - E - \mathcal{E}$ and compute $d = e^{-1} \mod \phi(N)$. □

**Notes**: If the prime difference $\Delta$ is of the general size, i.e., $\Delta = N^{1/r}$, then the condition for our attack becomes $\delta < 1 - \sqrt{1 - 1/r}$ which exactly coincides with Ciet et al.'s result. And when $r = 2$, the condition becomes $\delta < 1 - \sqrt{1/2} \approx 0.292$ which is the best known result of small private exponent attack on original RSA.

### 4.2 Fermat-Like Factoring Attack

We have recalled that for original RSA, Fermat's factoring technique can factor the modulus $N = pq$ efficiently when $|p - q| < N^{1/4}$. Bahig et al. presented that when extending de Weger's attacks to MPRSA, they also tried to generalize the Fermat's factoring attack but failed. In this section, we provide a way of factoring the modulus of MPRSA when the prime difference is small, which obtains a result quite similar to that of Fermat's factoring attack. We call this attack Fermat-like Factoring. This attack is based on Coppersmith's work of finding small solutions of a univariate modular equation. We first recall a general form of Coppersmith's original univariate modular result, stated by May [14], and then demonstrate our attack.

**Theorem 3 (May [14]).** *Let $N$ be an integer of unknown factorization, which has a divisor $b \geq N^{\beta}$. Let $f_b(x)$ be a monic univariate polynomial of degree $n$, let $\epsilon > 0$. Then, for a sufficiently large $N$, we can find all solution $x_0$ for the equation $f_b(x) \equiv 0 \mod b$, such that $|x_0| \leq N^{\beta^2/n - \epsilon}$ in time polynomial in* $\log N$, *$1/\epsilon$ and linear in the number of solutions.*

**Theorem 4 (Factoring Attack).** *Let $N = p_1 p_2 \dots p_r$ be a balanced MPRSA modulus, $p_1 < p_2 < \dots < p_r$, $p_r - p_1 = N^{\gamma}$, $0 < \gamma < 1/r$. If $\gamma \leq \frac{1}{r^2}$, then the modulus $N$ can be factored in time polynomial in* $\log N$.

*Proof.* Let $p = \lfloor N^{1/r} \rfloor$. As $N$ is a balanced modulus, $p_1 < p_2 < \dots < p_r$, we have

$$p_1 < p < p_r \text{ and } |p - p_i| < p_r - p_1 = N^{\gamma}.$$

Let $x_i = p_i - p$ for $i = 1, 2, \dots, r$. Then $|x_i| < N^{\gamma}$. Define a monic linear polynomial $f(x) = x + p$, which has root $x_i$ modulo $p_i$ since $f(x_i) = x_i + p = p_i = 0 \mod p_i$. As $p_i > 1/2 N^{1/r}$, we have $|p_i| > N^{1/r - \epsilon'}$ for some $\epsilon'$ which will be neglectable when $N$ is large. According to Theorem 3, by using the Coppersmith's method, we can find all the solutions of equation $f(x) = 0 \mod p_i$ in time polynomial in $\log N$ if $|x_i| \leq N^{1/r^2 - \hat{\epsilon}}$, i.e., $\gamma \leq 1/r^2$. Namely, the modulus $N$ can be factored in this way when $\gamma \leq 1/r^2$. □

**Notes**: For $r = 2$, this attack can factor the modulus $N$ when $\Delta \leq N^{1/4}$ which achieves the same effect as the Fermat's factoring method.

## 5 Experimental Results

In this section, we show some experimental data which provides an overview of the effectiveness of our attacks. We implemented our attacks on a personal computer with Intel CPU T230(1.73GHz), 4G RAM. The NTL Library [18] is used to construct the lattice and reduce the basis (LLL algorithm) and Maple system is used to solve the equations constructed from the LLL-reduced lattice basis.

Based on a 24 dimensional lattice, we mounted our small private exponent attack on instances of 3-MPRSA with different sizes of prime differences and obtained an experimental bound, shown as Figure 3. This bound consists of the largest sizes of private exponents, for different given prime differences, that we could use to successfully mount our attack with a 24 dimensional lattice. It is not a bound for the attacks in practice but only the largest size of private exponent that we could break with our set of experiments. Actually, the performance of our attack is strongly related to the size of the used lattice. We can make the experimental bound as close to our theoretical bound as we want at the expense of using larger lattice dimensions.

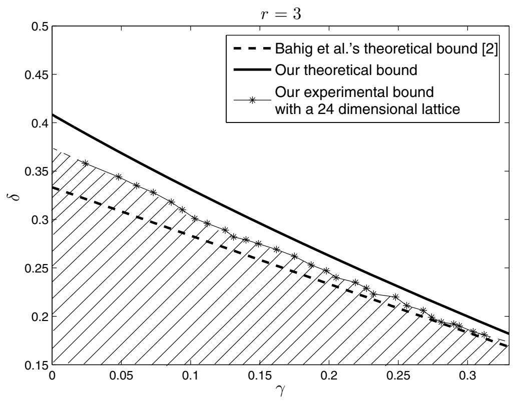


**Fig. 3.** Experimental bound of our attack on 3-MPRSA with a 24 dimensional lattice

It is clear that our attack works well in practice. When the prime differences are of the same size, our attack (using a 24 dimensional lattice) can break instances of MPRSA with private exponents exceeding Bahig et al.'s bound.

In Table 1, we illustrate the effectiveness of our small private exponent attack mounted on instances of MPRSA with a 1024-bit modulus. The prime difference is fixed around 0.15. We list the dimension $\omega$ of the used lattice, the experimental bound $\delta_{exp}$ and the corresponding runtime of each attack. Table 1 demonstrates that we can expect the experimental bound to increase up to the theoretical bound with the increasing dimension of the lattice used in the attack. We only show the data of the attacks at $\gamma \approx 0.15$ in Table 1. In fact, this conclusion is tenable for all the values of $\gamma$. As LLL reduction algorithm is a polynomial time algorithm, our attack will also be time polynomial in $\omega$ and $\log N$. Thus the runtime is roughly the same for a given choice of $\omega$ and $\log N$, regardless of the value of $\gamma$.

**Table 1.** Effectiveness of our small private exponent attack on 3-MPRSA with different size of lattice

| $\gamma$ | $\delta_{exp}$ | $\omega$ | runtime |
|---|---|---|---|
| 0.149 | 0.151 | 6 | 8.3 s |
| 0.150 | 0.212 | 8 | 4 m 20 s |
| 0.150 | 0.241 | 15 | 3 h 21 m |
| 0.149 | 0.275 | 24 | 47 h 48m |

s: second, m: minute, h: hour
When $\gamma = 0.15$, our theoretical bound on $\delta$ is 0.304.

**Table 2.** Effectiveness of the Fermat-like factoring attack

| $r$ | $\log N$ | $\gamma_{the}$ | $\gamma_{exp}$ | $\omega$ | runtime |
|---|---|---|---|---|---|
| $r = 3$ | 1024 | 0.111 | 0.108 | 8 | 2.8 s |
| $r = 4$ | 4096 | 0.0625 | 0.062 | 8 | 28 s |
| $r = 5$ | 8192 | 0.04 | 0.039 | 8 | 87 s |

The experimental results in Table 2 demonstrate the efficiency of the Fermat-like factoring attack. For several possible choices of $r$ and $\log N$, we list the theoretical bound $\gamma_{the}$, the observed experimental bound $\gamma_{exp}$, the dimension $\omega$ of the used lattice and the runtime of each attack. In our attacks $\omega$ is set 8. We tried to launch attacks with larger $\omega$ but the experimental bound can not be further improved. We know that this factoring attack becomes less effective (requires smaller prime difference) as $r$ increases, but for a given $r$, we find the experimental bounds are very close to the theoretical ones and the runtime keeps quite short i.e., only a few seconds.

## 6 Conclusion

We presented two attacks on MPRSA with small prime difference which can be regarded as an entire extension of de Weger's attack. From the results of our research, we can get a similar conclusion as de Weger's: small prime difference is a vulnerable feature for MPRSA. In particular, the prime difference can be used to enhance the small private exponent attack as long as it is smaller than its general size ($\varDelta < N^{1/r}$). Moreover, an extremely small prime difference ($\varDelta < N^{1/r^2}$) can lead to an efficient factorization of the modulus. We should note that when the primes are generated randomly and independently, with high probability the prime difference will be of the size of $N^{1/r}$. However, we still recommend to check the primes generated in the key generation step, especially for the case of using small private exponents.

## References

1. ANSI X9.31-1998, Digital signatures using reversible public key cryptography for the financial services industry (rDSA), American National Standards Institute (1998)
2. Bahig, H.M., Bhery, A., Nassr, D.I.: Cryptanalysis of multi-prime RSA with small prime difference. In: Chim, T.W., Yuen, T.H. (eds.) ICICS 2012. LNCS, vol. 7618, pp. 33–44. Springer, Heidelberg (2012)
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., Shacham, H.: Fast variants of RSA. CryptoBytes 5(1), 1–9 (2002)
5. Ciet, M., Koeune, F., Laguillaumie, F., Quisquater, J.-J.: Short private exponent attacks on fast variants of RSA. UCL Crypto Group Technical Report Series CG-2002/4, University Catholique de Louvain (2002)
6. Collins, T., Hopkins, D., Langford, S., Sabin, M.: Public key cryptographic apparatus and method. US patent #5, 848, 149 (1997)
7. Compaq Computer Corperation: Cryptography using Compaq multiprime technology in a parallel processing environment (2000)
8. Coppersmith, D.: Small solutions to polynomial equations, and low exponent RSA vulnerabilities. Journal of Cryptology 10, 233–260 (1997)
9. Herrmann, M., May, A.: Maximizing small root bounds by linearization and applications to small secret exponent RSA. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 53–69. Springer, Heidelberg (2010)
10. Hinek, M.J., Low, M.K., Teske, E.: On some attacks on multiprime RSA. In: Nyberg, K., Heys, H.M. (eds.) SAC 2002. LNCS, vol. 2595, pp. 385–404. Springer, Heidelberg (2003)
11. Howgrave-Graham, N.: Finding small roots of univariate modular equations revisited. In: Darnell, M. (ed.) Cryptography and Coding 1997. LNCS, vol. 1355, pp. 131–142. Springer, Heidelberg (1997)
12. Kühnel, M.: RSA vulnerabilities with small prime difference. In: Armknecht, F., Lucks, S. (eds.) WEWoRC 2011. LNCS, vol. 7242, pp. 122–136. Springer, Heidelberg (2012)
13. Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261, 513–534 (1982)
14. May, A.: Secret exponent attacks on RSA-type schemes with moduli $N = p^r q$. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 218–230. Springer, Heidelberg (2004)
15. National Institute of Standards and Technology: Digital signature standard, FIPS Publication 186-3 (2009), `http://www.nist.gov/cmvp`
16. Rivest, R.L., Shamir, A., Adleman, L.M.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM 21, 120–126 (1978)
17. RSA Laboratories: Public Key Cryptography Standards PKCS #1 v2.1: RSA cryptography standard (2001)
18. Shoup, V.: NTL number theory C++ library, `http://www.shoup.net/ntl`
19. de Weger, B.: Cryptanalysis of RSA with small prime difference. Applicable Algebra in Engineering, Communication and Computing 13, 17–28 (2002)
20. Wiener, M.: Cryptanalysis of short RSA secret exponents. IEEE Transactions on Information Theory 36, 553–558 (1990)

## Appendix

## Proof of Theorem 1

First we need to perform a linearization of the original polynomial $f(x, y) = xy + Ax - 1$. Let $u = xy - 1$. Then we get a linear polynomial $\bar{f}(u, x) = Ax + u$.

For some fixed integer $m$ and $t$, $m \geq t$, we now construct x-shifts:

$$\bar{g}_{i,k}(u, x) := x^i \bar{f}^k B^{m-k} \quad \text{for} \quad k = 0, \ldots, m \text{ and } i = 0, \ldots, m - k,$$

and y-shifts

$$\bar{h}_{j,k}(u, x, y) := y^j \bar{f}^k B^{m-k} \quad \text{for} \quad j = 0, \ldots, t \text{ and } k = \lfloor \frac{m}{t} \rfloor j, \ldots, m,$$

where $\lfloor \frac{m}{t} \rfloor$ means the largest integer which is smaller than $\frac{m}{t}$. We use $xy = u + 1$ to substitute each occurrence of $xy$ by the term $u + 1$ in y-shifts, and then construct a lattice by using the coefficient vectors of $\bar{g}_{i,k}(Uu, Xx)$ and $\bar{h}_{j,k}(Uu, Xx, Yy)$ as basis vectors, where $U, X, Y$ are bounds of $u, x, y$. For example, when $m = 2$ and $t = 1$, the constructed basis is as follows.

$$\begin{array}{c|ccccccc}
 & 1 & x & u & x^2 & ux & u^2 & u^2y \\
\hline
g_{0,0} & B^2 & & & & & & \\
g_{1,0} & & B^2X & & & & & \\
g_{0,1} & B & ABX & BU & & & & \\
g_{2,0} & & & & B^2X^2 & & & \\
g_{1,1} & & & & ABX^2 & BUX & & \\
g_{0,2} & & & & A^2X^2 & 2AUX & U^2 & \\
g_{1,2} & & A^2X & 2AU & & A^2UX & 2AU^2 & U^2Y
\end{array}$$

We can see that the above matrix is a triangular matrix. Actually, for any choice of $m$ and $t$, the corresponding matrix keeps being triangular. There is no doubt for the upper part constructed from the x-shifts. Now we look at an arbitrary y-shift $y^i \bar{f}^l$, where $i \in \{0, 1, \ldots, t\}$ and $l \in \{\lfloor \frac{m}{t} \rfloor i, \ldots, m\}$, (the factor $B^{m-l}$ is omitted as it does not influence the set of monomials). Since $\bar{f} = Ax + u$ we can expand $y^i \bar{f}^l$ as

$$u^l y^i + C_1^l A u^{l-1} x y^i + \cdots + C_l^l A^l x^l y^i.$$

Now we shall prove that the first term introduce a new monomial $u^l y^i$ and all other monomials are already present in the upper part. Let us look at the second term after the substitution of $xy$

$$u^{l-1} x y^i = u^{l-1}(u + 1) y^{i-1} = u^l y^{i-1} + u^{l-1} y^{i-1}.$$

The monomials $u^l y^{i-1}$ and $u^{l-1} y^{i-1}$ appear in $y^{i-1} \bar{f}^l$ and $y^{i-1} \bar{f}^{l-1}$, respectively. In general, the $(j + 1)^{th}$ term of the binomial expansion, for $j = 1, \ldots, i - 1$, contains monomials that appear in $y^{i-j} \bar{f}^{l-k}$ for $k = 0, \ldots, j$. For $i \leq j \leq l$, it is easy to prove that the $(j + 1)^{th}$ term contains monomials that appear in the x-sifts.

Now we will show that if $y^i\bar{f}^l$ is a y-shift, then all of $y^{i-j}\bar{f}^{l-k}$ for $j = 1, \ldots, i-1$ and $k = 0, \ldots, j$ are also used as y-shifts. Since $y^i\bar{f}^l$ is in the set of y-shifts, we know that $l \in \{\lfloor\frac{m}{t}\rfloor i, \ldots, m\}$ and therefore $l-k \in \{\lfloor\frac{m}{t}\rfloor i - j, \ldots, m\}$ when $k = 0, \ldots, j$. For $y^{i-j}\bar{f}^{l-j}$ on the other hand, we have $l-j \in \{\lfloor\frac{m}{t}\rfloor(i-j), \ldots, m\}$. As $m \geq t$, we have $\lfloor\frac{m}{t}\rfloor \geq 1$. Thus, $\lfloor\frac{m}{t}\rfloor(i-j) \leq \lfloor\frac{m}{t}\rfloor i - j$ and therefore $y^{i-j}\bar{f}^{l-k}$, for $k = 0, \ldots, j$, are also in the y-shift set.

Therefore, we can conclude that the constructed basis is a triangular matrix. Let $t = \tau m$ for some $0 < \tau \leq 1$. If we denote its determinant to be $\det(L) = X^{s_x}Y^{s_y}U^{s_u}B^{s_b}$, then we can figure out that

$$\begin{aligned}
s_x &= \sum_{k=0}^{m}\sum_{i=0}^{m-k} i = \frac{1}{6}m^3 + o(m^3) \\
s_y &= \sum_{j=1}^{\tau m}\sum_{k=j/\tau}^{m} j = \frac{\tau^2}{6}m^3 + o(m^3) \\
s_u &= \sum_{k=0}^{m}\sum_{i=0}^{m-k} k + \sum_{j=1}^{\tau m}\sum_{k=j/\tau}^{m} k = (\frac{1}{6} + \frac{\tau}{3})m^3 + o(m^3) \\
s_b &= \sum_{k=0}^{m}\sum_{i=0}^{m-k}(m-k) + \sum_{j=1}^{\tau m}\sum_{k=j/\tau}^{m}(m-k) = (\frac{1}{3} + \frac{\tau}{6})m^3 + o(m^3)
\end{aligned}$$

and the dimension of the lattice is

$$\dim(L) = \sum_{k=0}^{m}\sum_{i=0}^{m-k} 1 + \sum_{j=1}^{\tau m}\sum_{k=j/\tau}^{m} 1 = (\frac{1}{2} + \frac{\tau}{2})m^2 + o(m^2).$$

Using these values together with the upper bounds $X = B^\alpha$, $Y = B^\beta$, $U = B^{\alpha+\beta}$ on the variables in the usual enabling condition $\det L = X^{s_x}Y^{s_y}U^{s_u}B^{s_b} \leq B^{m\dim(L)}$, we obtain that $\alpha$ and $\beta$ should satisfy

$$\beta\tau^2 + (2\alpha + 2\beta - 2)\tau + (2\alpha + \beta - 1) \leq 0.$$

For any values of $\alpha$ and $\beta$, the left-hand side of this inequality is minimized when $\tau$ is chosen to be $\tau = \frac{1-\alpha-\beta}{\beta}$. Substituting this back into the inequality yields the enabling equation $\alpha^2 - 2\alpha + 1 - \beta \geq 0$. Finally, we derive the condition that

$$\alpha \leq 1 - \sqrt{\beta}.$$

□

# Factoring Multi-power RSA Modulus $N = p^r q$ with Partial Known Bits

Yao Lu[1,2], Rui Zhang[1], and Dongdai Lin[1]

[1] State Key Laboratory of Information Security (SKLOIS)
Institute of Information Engineering (IIE)
Chinese Academy of Sciences (CAS)
[2] University of Chinese Academy of Sciences (UCAS)
lywhhit@gmail.com, {r-zhang,ddlin}@iie.ac.cn

**Abstract.** Factoring large integers is a fundamental problem in algebraic number theory and modern cryptography, which many cryptosystems, e.g. RSA, are based on. Up to now, there is no known polynomial-time algorithm to solve it with classical computers. However, in practice side-channel attacks usually cause serious damage: Even if a small proportion of bits in the secret primes is leaked, one may efficiently factor.

In this paper, we study the problem of factoring with partial known bits for multi-power RSA modulus $N = p^r q$. In 1999, Boneh, Durfee and Howgrave-Graham showed that this problem can be solved efficiently given a $\frac{1}{r+1}$-fraction of the most significant bits (MSB) of $p$. In their attack, the unknown bits are located in one consecutive block. We propose two lattice-based approaches that extend the number of unknown blocks to arbitrary $n$ ($n \geq 1$). The advantage of our approaches is that now knowledge of a $\frac{\ln(1+r)}{r}$-fraction of the bits of $p$ is already sufficient (for any $n$). In fact, our result is a first step towards unifying and extending previous works by Boneh-Durfee-Howgrave (Crypto'99) and Herrmann-May (Asiacrypt'08).

**Keywords:** factoring with known bits, RSA, combined attacks.

## 1 Introduction

The RSA cryptosystem is the most widely used public-key cryptosystem today. Denote an RSA modulus by $N = pq$ that is the product of two primes $p, q$ of the equal length. Let $e$ be a positive integer that is co-prime to Euler's totient function $\phi(N)$. The RSA encryption function takes a message $m$ (encoded as an element in $\mathbb{Z}_N$) to the $e$-th power in the ring $\mathbb{Z}_N$. However, since RSA is based on arithmetic modulo large numbers, it can be slow in the resource-constrained environments.

To speed up decryption, some variants of RSA are suggested, (see a nice survey here [3]). Among them, an important one is the multi-power RSA scheme proposed by Takagi [20] in 1998. This fast RSA variant modifies the structure of the standard RSA modulus ($N = pq$ with $p, q$ are of the same bit-size), and uses modulus of the form $N = p^r q$ ($r > 1$) where two primes $p, q$ are equal in length. Compared to the standard RSA scheme, the multi-power RSA is more efficient in both key generation and decryption. Besides, modulus of this type has been applied in many cryptographic designs, e.g., the Okamoto-Uchiyama cryptosystem [16], or better known via EPOC and ESIGN [5], which uses the modulus $N = p^2 q$.

The security of the multi-power RSA, like that of standard RSA, is related to the hardness of factoring large integers. Until now there is no known polynomial-time algorithm to factorize large numbers except quantum algorithms. The best algorithm to date is Number Field Sieve (NSF), which works in sub-exponential time. However, in a real-world implementation, partial information regarding the factoring of $N$ can be leaked by side-channel attacks, hence it is crucial to study how this affects the hardness of the factoring problem.

In the context of standard RSA where $N = pq$, there have been a number of results. In 1985, Rivest and Shamir [17] first studied the problem factoring with known bits, they designed an algorithm to factor $N$ given $\frac{2}{3}$-fraction of the bits of $p$. In 1996, Coppersmith [4] improved this bound to $\frac{1}{2}$. Note that for the above results, the unknown bits are within one consecutive block. The case of $n$ blocks was first considered by Herrmann and May in [9], where they showed that $\ln 2 \approx 70\%$ known bits in $p$ are sufficient to factor $N = pq$, however, the running time of their algorithm is polynomial only for $n = \mathcal{O}(\log \log N)$ blocks.

In this paper, we consider a more general problem: factoring multi-power RSA modulus $N = p^r q$ $(r \geq 1)$ with known bits in $p$. Note that Boneh, Durfee and Howgrave-Graham [2] first considered this problem in 1999 and they shown that $N$ can be recovered efficiently given $\frac{1}{r+1}$-fraction of the most significant bits (MSB) of $p$. Similar to [17,4], it is assumed that the unknown bits are located in one consecutive block, however, very often the leaked bits obtained by side-channel attack can be scattered over all positions in $p$, for instance, the cold-boot attack reported in [7].

Therefore, a very interesting question rises that how can we efficiently utilize the fragmentary known bits in $p$ for the multi-power RSA problem? As far as we are currently aware, there is no known works dealing with it. On the other hand, when $r = 1$, the multi-power RSA problem degenerates to the standard RSA problem. Studying the multi-power RSA problem helps to understand the security of the standard RSA problem. In this paper, we investigate factoring $N = p^r q$ with known bits, which can be viewed as a step towards unifying all the previous works [17,4,2,9].

**Our Contributions.** Let $p \geq N^{\beta}$, and let $n$ denote the number of the unknown blocks. In this paper we show that we can factorize the multi-power RSA modulus $N = p^r q$ given a

$$1 - \frac{1}{r\beta}\left(1 - (1 - r\beta)^{\frac{n+1}{n}} - (n+1)(1 - r\beta)\left(1 - \sqrt[n]{1 - r\beta}\right)\right)$$

fraction of the bits in $p$ together with their positions. Our results generalize the previous results in the following sense:

- For $\beta = \frac{1}{r+1}$ and $n = 1$, which means that we are given the most or least significant bits of $p$ for modulus $N = p^r q$, our results show that we can factorize $N$ given a $\frac{1}{r+1}$-fraction of the most or least significant bits, which is exactly Boneh-Durfee-Howgrave showed in [2].
- For $\beta = \frac{1}{r+1}$, $r = 1$ and $n$ is large, which means that we are given many bit blocks of $p$ for modulus $N = pq$, our results show that we can factorize $N$ given $\ln 2 \approx 70\%$ of the bits of $p$ (see Table 1), which also matches Herrmann and May showed [9].

Unfortunately, similar to [9], the running time of our algorithm heavily depends on $n$, to be specific, our algorithm is polynomial-time only for $n = \mathcal{O}(\log \log N)$ blocks.

**Our Treatments.** Technically, we develop two methods for finding small roots of linear modular polynomials $f(x_1, \cdots, x_n) = a_0 + a_1 x_1 + \cdots + a_n x_n \mod p$, where $p$ is an unknown divisor of some known $N$ and $N \equiv 0 \mod p^r$.

Our first method is to transform the above constrained polynomials to multivariate integer polynomials $h(x_1, \cdots, x_{n+1}) = N - f(x_1, \cdots, x_n)^r x_{n+1} = N - (a_0 + a_1 x_1 + \cdots + a_n x_n)^r x_{n+1}$ , then analyze the integer polynomials using Coppersmith's method [4].

Our second method uses Herrmann-May's idea [9] in the $N = p^r q$ setting and makes some suitable improvements. Herrmann and May solved the problem of finding small roots of linear modular polynomials $f(x_1, \cdots, x_n) = a_0 + a_1 x_1 + \cdots + a_n x_n \mod p$ for some unknown divisor $p$ of known modulus $N$. In our second method, we make additional use of the fact that $N \equiv 0 \mod p^r$. More precisely, we construct a lattice with subtle structures to reflect the property for $N = p^r q$. In general, the two methods both achieve the same asymptotic bounds, however, their performances may differ significantly. We give detailed discussions on this.

The rest of the paper is organized as follows. In Section 2, we review some useful notations and facts. In Section 3, we review some previous works on the target problem. In Section 4 5, we present two methods to prove our main theorems, and give detailed discussions as well as comparisons of the two methods. In Section 6, we give a conclusion.

## 2 Preliminary

Consider a set of linearly independent vectors $u_1, \cdots, u_w \in \mathbb{Z}^n$, with $w \leqslant n$. The lattice $\mathcal{L}$, spanned by $\{u_1, \cdots, u_w\}$, is the set of all integer linear combinations of the vectors $u_1, \cdots, u_w$. The number of vectors is the dimension of the lattice. The set $u_1, \cdots, u_w$ is called a basis of $\mathcal{L}$. In lattices with arbitrary dimension, finding the shortest vector is a very hard problem, however, approximations of a shortest vector can be obtained in polynomial-time by applying the well-known *LLL* basis reduction algorithm [13].

**Lemma 1 (LLL [13]).** *Let $L$ be a lattice of dimension $w$. In polynomial-time, the LLL-algorithm outputs reduced basis vector $v_i$, $1 \leqslant i \leqslant w$ that satisfy*

$$\| v_1 \| \leqslant \| v_2 \| \leqslant \cdots \leqslant \| v_i \| \leqslant 2^{\frac{w(w-1)}{4(w+1-i)}} \det(\mathcal{L})^{\frac{1}{w+1-i}}$$

**Lemma 2 (Howgrave-Graham [10]).** *Let $g(x_1, \cdots, x_k) \in \mathbb{Z}[x_1, \cdots, x_k]$ be an integer polynomial that consists of at most $w$ monomials. Suppose that*

1. *$g(y_1, \cdots, y_k) = 0 \mod p^m$ for $| y_1 | \leqslant X_1, \cdots, | y_k | \leqslant X_k$ and*
2. *$\| g(x_1 X_1, \cdots, x_k X_k) \| < \frac{p^m}{\sqrt{w}}$*

*Then $g(y_1, \cdots, y_k) = 0$ holds over the integers.*

We mention a useful lemma (Lemma 2 of [8]).

**Lemma 3 (Herrmann [8]).** *Let $P = \{(i_1, \ldots, i_n) \in \mathbb{Z}^n | \forall j, \sum_{j=1}^n \leq r \wedge i_j \geq 0\}$ be an $n$-dimensional simplex in $\mathbb{Z}^n$. Then the number of points in $P$ is $\binom{r+n}{n}$, and $\forall\ j \in \{1, \ldots, n\}$, $s_j = \sum_{(i_1,\ldots,i_n)\in P} i_j = \binom{r+n}{r-1}$.*

Let $g(x_1, \cdots, x_k) = \sum_{i_1,\cdots,i_k} a_{i_1,\cdots,i_k} x_1^{i_1} \cdots x_k^{i_k}$. We define the norm of $g$ by the Euclidean norm of its coefficient vector: $\| g \|^2 = \sum_{i_1,\cdots,i_k} a^2_{i_1,\cdots,i_k}$.

In our analysis we relies on the following assumption when extracting the final roots efficiently, which was used in [9].

**Assumption 1.** *The lattice-based construction yields algebraically independent polynomials, the common roots of these polynomials can be efficiently computed using techniques like calculation of the resultants or finding a* Gröbner basis.

May [14] gives upper bounds on the size of the solutions of a univariate equation modulo an unknown divisor.

**Theorem 1 (May [14]).** *Let $N$ be an integer of unknown factorization, which has a divisor $p \geq N^\beta$, $0 < \beta \leq 1$. Let $f(x)$ be a univariate polynomial of degree $\delta$. Then we can find in time $O(c\delta^5 \log^9 N)$ all solutions $x_0$ for the equation*

$$f(x) = 0 \mod p \quad \text{with} \quad |x_0| \leq cN^{\frac{\beta^2}{\delta}}.$$

Herrmann and May [9] give upper bounds on the size of the solutions of a multivariate linear equations modulo an unknown divisor of a known composite.

**Theorem 2 (Herrmann-May [9]).** *Let $\epsilon \geq 0$ 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 $f(x_1, \ldots, x_n) = 0 \pmod p$ with $|x_1^0| \leq N^{\gamma_1}, \ldots, |x_n^0| \leq N^{\gamma_n}$ if*

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

*The running time of the algorithm is polynomial in $\log N$ and $(\frac{e}{\epsilon})^n$.*

## 3 Previous Works

In this section we review the previous works on the problem of factoring $N = p^r q$ with known bits in the secret prime $p$.

### 3.1 BDH Method

This attack was proposed by Boneh, Durfee and Howgrave-Graham [2], later revisited by May [15], as will be referred to as the "BDH Method". Basically, they considered the scenario that a few most significant bits in the prime $p$ are known to the attacker. Consider the univariate polynomial

$$f(x) = (\tilde{p} + x)^r \mod p^r$$

For simplicity, we assume that $p$ and $q$ are of the same bit-size. Set $\beta = \frac{r}{r+1}$, $\delta = r$ and $c = 1$. Applying Theorem 1, the $LLL$ algorithm recovers all roots $x$ with

$$|x_0| \leq M^{\frac{\beta^2}{\delta}} = N^{\frac{r}{(r+1)^2}}$$

Since $N$ is roughly of the size $p^{r+1}$, this means that we need an approximation $\tilde{p}$ with $|p - \tilde{p}| \leq p^{\frac{r}{r+1}}$. In other words, we need a $\frac{1}{r+1}$-fraction of the most significant bits in order to factorize $N$ in polynomial-time.

### 3.2 Herrmann-May Method

In Asiacrypt'08, Herrmann and May [9] proposed an algorithm to find solutions to linear equations modulo an unknown divisors $p$ of a known composite integer $N$, which can be also directly used in this scenario. Consider the multivariate linear polynomial

$$f(x_1, x_2, \ldots, x_n) = a_0 + a_1 x_1 + a_2 x_2 + \cdots + a_n x_n \mod p$$

where $a_k = 2^l$ if the $k$-th unknown blocks starts in the $l$-th bit position. For simplicity, we assume that $p$ and $q$ are of the same bit-size, thus $\beta = \frac{1}{r+1}$. Applying Theorem 2, the $LLL$ algorithm recovers all roots with

$$\sum_{i=1}^{n} \gamma_i \leq 1 - \frac{r(n+1)}{r+1} + n(\frac{r}{r+1})^{\frac{n+1}{n}} - \epsilon$$

If $n$ is large, we obtain the limit of the above equation

$$\lim_{n \to \infty} (1 - \frac{r(n+1)}{r+1} + n(\frac{r}{r+1})^{\frac{n+1}{n}}) = \frac{1}{r+1} + \frac{r}{1+r} \ln(\frac{r}{1+r})$$

It shows that if we known a $\ln(1 + \frac{1}{r})^r$-fraction of bits in $p$, we can recover the unknown bits regardless of the number of unknown blocks $n$.

*Remark 1.* For a fixed $N$ and a growing $r$, the modulus should be, from an information-theoretic point of view, easier to factor than standard modulo. However, Herrmann-May Method does not exploit the unbalanced relation between $p$ and $q$. On the other hand, though the BDH Method take the advantage of the relation, but they do not extend the problem of factoring $N = p^r q$ with known bits to an arbitrary number of unknown blocks.

## 4 Our Results

In this section, we present our main theorem, which can be acquired via two different approaches. We further analyze and compare our two approaches.

### 4.1 The Main Theorem

Our main result is summarized below.

**Theorem 3.** *Let $N$ be a sufficiently large composite integer (of unknown factorization) with a divisor $p^r$ ($p \geq N^{\beta}$ and an integer $r \geq 1$). 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 $f(x_1, \ldots, x_n) = 0 \pmod p$ with $|x_1^0| \leq N^{\gamma_1}, \ldots, |x_n^0| \leq N^{\gamma_n}$ if*

$$\sum_{i=1}^{n} \gamma_i < \frac{1}{r}\left(1 - (1 - r\beta)^{\frac{n+1}{n}} - (n+1)(1 - r\beta)\left(1 - \sqrt[n]{1 - r\beta}\right)\right)$$

*The running time of the algorithm is polynomial in $\log N$ but exponential in $n$.*

*Remark 2.* In Theorem 3 we require that the linear polynomial is monic with respect to one of the variables, i.e., $a_i = 1$ $(1 \leq i \leq n)$. This is usually not a restriction, since we could multiply $f(a_1, \cdots, a_n)$ by $a_i^{-1} \bmod N$. If this inverse does not exist, we can factorize $N$.

The proof of Theorem 3 is postponed to Section 5. An immediate application of Theorem 3 is to factorize $N$ given bits of $p$. The following theorem shows that the extreme case when $n$ becomes very large.

**Theorem 4.** *Let $N = p^r q$ where $p$ and $q$ are of equal length. Suppose a $\frac{\ln(1+r)}{r}$-fraction of the bits are known for $n$ blocks in $p$ ($n$ is large), then under Assumption 1, we can recover $p$. The running time of the algorithm is polynomial in $\log N$ but exponential in $n$.*

*Proof.* We model this factoring with known bits problem as the multivariate linear polynomial

$$f(x_1, x_2, \ldots, x_n) = a_0 + a_1 x_1 + a_2 x_2 + \cdots + a_n x_n \mod p$$

**Table 1.** Lower bounds for the fraction of known bits in $p$ to factor $N = p^r q$

| $r$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| [9] | 69.4% | 81.1% | 86.4% | 89.3% | 91.2% | 92.5% | 93.5% | 94.3% | 94.9% | 95.4% |
| Ours | 69.4% | 55.0% | 46.3% | 40.3% | 35.9% | 32.5% | 29.8% | 27.5% | 25.6% | 24.0% |

where $a_k = 2^l$ if the $k$-th unknown blocks starts in the $l$-th bit position. Moreover, if $n$ goes to infinity, from Theorem 3, we have

$$\lim_{n\to\infty}\left(\frac{1}{r}\left(1-(1-r\beta)^{\frac{n+1}{n}}-(n+1)(1-r\beta)\left(1-\sqrt[n]{1-r\beta}\right)\right)\right)$$
$$=\beta+\frac{(1-r\beta)\ln(1-r\beta)}{r}$$

It shows that if $n$ is very large, we can recover $p$ regardless of $n$. Conversely, once a

$$\left(1-\frac{1}{r\beta}\right)\ln(1-r\beta)$$

fraction of the bits from $p$ together with their positions are given, we are able to recover the missing bits. Suppose that the bit length of $|p| = |q|$, i.e. $\beta = \frac{1}{r+1}$, we need a $\frac{\ln(r+1)}{r}$-fraction of known bits from $p$. □

For small $r$, we provide in Table 1 the lower bounds on the percentage of the leakage of $p$. In particular, for $r = 1$, we obtain the same result as Herrmann and May [9].

## 5 Proof of Theorem 3

We use two different approaches to prove Theorem 3: The first one is based on the general strategy of finding small roots of a multivariate integer equation, and the second one from the Herrmann-May method, where we extend the original algorithm with additional using the algebraic property $N \equiv 0 \mod p^r$. Before giving the detailed proofs, we prove a useful lemma.

**Lemma 4 (Sarkar-Maitra [18]).** *For a large positive integer $m$, we have*

$$\sum_{k=0}^{m} k^n = \frac{1}{n+1}m^{n+1} + o(m^{n+1})$$

*Proof.* Let $S = \sum_{k=0}^{m} k^n$, then $\int_0^m x^n dx < S < \int_1^{m+1} x^n dx$, therefore,

$$\frac{1}{n+1}m^{n+1} < S < \frac{1}{n+1}(m+1)^{n+1}$$

The claimed result can then be easily obtained. □

## 5.1 Our First Approach

Consider a polynomial defined over integers:

$$h(x_1, \cdots, x_{n+1}) = N - f(x_1, \cdots, x_n)^r x_{n+1} = N - (a_0 + a_1 x_1 + \cdots + a_n x_n)^r x_{n+1}$$

where $x_{n+1}$ stands for $\frac{N}{p^r}$. We try to find the small roots of $h(x_1, \cdots, x_{n+1})$ by defining a collection of shift polynomials

$$g_{i_1,\ldots,i_{n+1}}(x_1, \ldots, x_{n+1}) = x_1^{i_1} \cdots x_{n+1}^{i_{n+1}} h(x_1, \ldots, x_{n+1})$$

where $x_1^{i_1} \cdots x_{n+1}^{i_{n+1}} \in S$ for a set of monomials $S$, which is specified below:

$$S = \bigcup_{0 \le \sum_{i=1}^{n} t_i \le m - tr} \{x_1^{i_1+t_1} \cdots x_n^{i_n+t_n} x_{n+1}^{i_{n+1}} \mid x_1^{i_1} \cdots x_n^{i_n} x_{n+1}^{i_{n+1}} \text{ is monomial of } h^t\}$$

Here $t = \tau m$, and we will optimize it later. We also define the set $M$ as the set of all polynomials that appeared in the shift polynomials. More precisely, $S$ and $M$ can be described as follows:

$$x_1^{i_1} \cdots x_n^{i_{n+1}} x_{n+1}^{i_{n+1}} \in S \Leftrightarrow \begin{cases} i_1 & = 0, \ldots, m, \\ i_2 & = 0, \ldots, m - i_1, \\ \cdots & \\ i_n & = 0, \ldots, m - \sum_{j=1}^{n-1} i_j, \\ i_{n+1} & = \max\{0, \lceil \frac{\sum_{j=1}^{n} i_j - m}{r} + t \rceil\}, \ldots, t \end{cases}$$

$$x_1^{i_1} \cdots x_n^{i_{n+1}} x_{n+1}^{i_{n+1}} \in M \Leftrightarrow \begin{cases} i_1 & = 0, \ldots, m + r, \\ i_2 & = 0, \ldots, m + r - i_1, \\ \cdots & \\ i_n & = 0, \ldots, m + r - \sum_{j=1}^{n-1} i_j, \\ i_{n+1} & = \max\{0, \lceil \frac{\sum_{j=1}^{n} i_j - m}{r} + t \rceil\}, \ldots, t + 1 \end{cases}$$

Next we construct the matrix. Let $s = |S|$ denote the total number monomials belong to $S$ and $d = |M| - |S|$ denote the difference of the number of the monomials belong to $M$ and $S$.

The upper left $d \times d$ block is diagonal, where the rows represent the monomials $x_1^{i_1} \cdots x_n^{i_n} x_{n+1}^{i_{n+1}} \in M \setminus S$. The diagonal entry of the row corresponding to $x_1^{i_1} \cdots x_n^{i_n} x_{n+1}^{i_{n+1}}$ is $(X_1^{i_1} \cdots X_n^{i_n} X_{n+1}^{i_{n+1}})^{-1}$. The lower left $s \times d$ block contains only zeros.

The last $s$ columns of the matrix represent the shift polynomials $g_{i_1,\ldots,i_{n+1}}$, for $x_1^{i_1} \cdots x_{n+1}^{i_{n+1}} \in S$. We define the polynomial order for our collection. Let $g_{i_1,\ldots,i_{n+1}}$, $g_{j_1,\ldots,j_{n+1}}$ be two polynomials. If $i_1 + \cdots + i_n > j_1 + \cdots + j_n$, then $\text{order}(g_{i_1,\ldots,i_{n+1}}) < \text{order}(g_{j_1,\ldots,j_{n+1}})$, if $i_1 + \cdots + i_n = j_1 + \cdots + j_n$ then $\text{order}(g_{i_1,\ldots,i_{n+1}}) < \text{order}(g_{j_1,\ldots,j_{n+1}}) \Leftrightarrow i_{n+1} > j_{n+1}$. If we sort the polynomial according to this order, we obtain an upper triangular lattice. A lattice $\mathcal{L}$ for the case $(n, r, t, m) = (1, 2, 1, 3)$ is depicted in Figure 1.

$$
\begin{array}{l|cccccc|cccccc}
 & & & & & & & \mathbf{x_1^3x_2h} & \mathbf{x_1^2x_2h} & \mathbf{x_1x_2h} & \mathbf{x_1h} & \mathbf{x_2h} & \mathbf{h} \\
x_1^5x_2^2 & * & & & & & & -a_1^2 & & & & & \\
x_1^4x_2^2 & & * & & & & & 2a_0a_1 & -a_1^2 & & & & \\
x_1^3x_2^2 & & & * & & & & -a_0^2 & 2a_0a_1 & -a_1^2 & & & \\
x_1^2x_2^2 & & & & * & & & & -a_0^2 & 2a_0a_1 & & -a_1^2 & \\
x_1x_2^2 & & & & & * & & & & -a_0^2 & & 2a_0a_1 & \\
x_2^2 & & & & & & * & & & & & -a_0^2 & \\
\hline
x_1^3x_2 & & & & & & & N & & & -a_1^2 & & \\
x_1^2x_2 & & & & & & & & N & & 2a_0a_1 & & -a_1^2 \\
x_1x_2 & & & & & & & & & N & -a_0^2 & & 2a_0a_1 \\
x_1 & & & & & & & & & & N & & \\
x_2 & & & & & & & & & & & N & -a_0^2 \\
1 & & & & & & & & & & & & N
\end{array}
$$

**Fig. 1.** The Matrix for the case $n = 1$, $r = 2$, $t = 1$, $m = 3$

It is easy to see, the determinate of the lattice $\mathcal{L}$ is

$$\det(\mathcal{L}) = N^s \cdot \prod_{x_1^{i_1}\cdots x_{n+1}^{i_{n+1}} \in M\setminus S} (x_1^{i_1} \cdots x_{n+1}^{i_{n+1}})^{-1} = N^s \prod_{i=1}^{n+1} X_i^{-s_i}$$

For the lattice attack to work, we require the enabling condition $\det(\mathcal{L}) > 1$. Here, we denote $s_i$ as the contribution of $X_i$ to the determinant. Next we compute the above unknowns.

First we compute the value of $s$.

$$
\begin{aligned}
s = |S| &= \sum_{0 \le \sum_{j=1}^n i_j \le m} \quad \sum_{i_{n+1}=\max\{0, \lceil \frac{\sum_{j=1}^n i_j - m}{r} + t \rceil\}}^{t} 1 \\
&= (t+1) \sum_{0 \le \sum_{j=1}^n i_j \le m} 1 + \sum_{m - tr \le \sum_{j=1}^n i_j \le m} -\lceil \frac{\sum_{j=1}^n i_j - m}{r} + t \rceil \\
&= (t+1) \sum_{0 \le \sum_{j=1}^n i_j \le m} 1 - \sum_{k=1}^{t} \sum_{m+(k-t-1)r < \sum_{j=1}^n i_j \le m+(k-t)r} k \\
&= (t+1)\binom{m+n}{m} - \sum_{k=1}^{t} k \binom{m - tr + kr + n}{m - tr + kr} \\
&\quad + \sum_{k=1}^{t} k \binom{m - tr + (k-1)r + n}{m - tr + (k-1)r}
\end{aligned}
$$

$$
\begin{aligned}
&= \sum_{k=0}^{t} \binom{m-tr+kr+n}{m-tr+kr} \\
&\approx \sum_{k=0}^{t} \frac{(m-tr+kr)^n}{n!} \\
&= \frac{1-(1-\tau r)^{n+1}}{r(n+1)!} m^{n+1} + o(m^{n+1}) \qquad \text{(Using } \ t=\tau m)
\end{aligned}
$$

Following we compute the value of $s_i$ $(1 \le i \le n)$.

$$
\begin{aligned}
& s_1 = \cdots = s_n \\
&= \sum_{0 \le \sum_{j=1}^n i_j \le m+r} \ \sum_{i_{n+1}=\max\{0, \lceil \frac{\sum_{j=1}^n i_j - m}{r} + t \rceil\}}^{t+1} i_1 \\
&\quad - \sum_{0 \le \sum_{j=1}^n i_j \le m} \ \sum_{i_{n+1}=\max\{0, \lceil \frac{\sum_{j=1}^n i_j - m}{r} + t \rceil\}}^{t} i_1 \\
&= \sum_{0 \le \sum_{j=1}^n i_j \le m+r} i_1 \\
&= \binom{m+r+n}{m+r-1} \\
&\approx \frac{1}{(n+1)!} m^{n+1} + o(m^{n+1})
\end{aligned}
$$

At last we compute the value of $s_{n+1}$.

$$
\begin{aligned}
& s_{n+1} \\
&= \sum_{0 \le \sum_{j=1}^n i_j \le m+r} \ \sum_{i_{n+1}=\max\{0, \lceil \frac{\sum_{j=1}^n i_j - m}{r} + t \rceil\}}^{t+1} i_{n+1} \\
&\quad - \sum_{0 \le \sum_{j=1}^n i_j \le m} \ \sum_{i_{n+1}=\max\{0, \lceil \frac{\sum_{j=1}^n i_j - m}{r} + t \rceil\}}^{t} i_{n+1} \\
&= (t+1) \sum_{0 \le \sum_{j=1}^n i_j \le m+r} 1 \\
&= (t+1) \binom{m+r+n}{m+r} \\
&\approx \frac{\tau}{n!} m^{n+1} + o(m^{n+1}) \qquad \text{(Using } \ t=\tau m)
\end{aligned}
$$

Using the values together with the upper bounds $X_i = N^{\gamma_i}(1 \le i \le n)$, $X_{n+1} = N^{1-r\beta}$ on the variables in the usual enabling condition $\det(\mathcal{L}) > 1$, we obtain the condition

$$\sum_{i=1}^{n} \gamma_i < \frac{1}{r}\left(1 - (1-\tau r)^{n+1}\right) - \tau(n+1)(1-r\beta)$$

Setting $\tau = \frac{1-\sqrt[n]{1-r\beta}}{r}$, the condition reduces to

$$\sum_{i=1}^{n} \gamma_i < \frac{1}{r}\left(1 - (1-r\beta)^{\frac{n+1}{n}} - (n+1)(1-r\beta)\left(1 - \sqrt[n]{1-r\beta}\right)\right)$$

The running time is dominated by $LLL$ reduction, which is polynomial in the dimension of the lattice and in the bitsize of the entries. Recall that our lattice's dimension is $\mathcal{O}(m^{n+1})$ and the product $X_1^{i_1} \cdots X_{n+1}^{i_{n+1}}$ is upper bound by $2m\log N$ ($\sum_{j=1}^{n} i_j \le m$, $i_{n+1} \le t$, and $X_i < N(1 \le i \le n+1)$ ), therefore, the total running time for this algorithm is polynomial in $\log N$ but exponential in $n$.

*Remark 3.* We also analyze the integer polynomial $h(x_1, \cdots, x_{n+1})$ using Jochemsz and May's [11] strategy, and obtained the same asymptotic bounds. Here we adopt the Coppersmith method because of its major practical advantage, which has been pointed out in [11].

## 5.2 Our Second Approach

We will prove Theorem 3 by making a small twist to the Herrmann-May method. They proposed a heuristic lattice-based algorithm for finding small solutions of linear modular polynomial $f(x_1, \cdots, x_n) = a_0 + a_1x_1 + \cdots + a_nx_n \bmod p$. In our method, we make additional use of the fact that $N \equiv 0 \bmod p^r$.

We define the following collection of polynomials which share a common root modulo $p^t$

$$g_{i_2,\ldots,i_n,k} = x_2^{i_2} \cdots x_n^{i_n} f^k N^{\max\{\lceil \frac{t-k}{r} \rceil, 0\}}$$

where $i_j \in \{0, \ldots, m\}$ such that $\sum_{j=2}^{n} i_j \le m - k$, and the parameter $t = \tau m$ has to be optimized. For comparison, we give the definition of Herrmann-May's collection of polynomials

$$g_{i_2,\ldots,i_n,k} = x_2^{i_2} \cdots x_n^{i_n} f^k N^{\max\{t-k, 0\}}$$

The idea behind the above transformation is that we try to eliminate powers of $N$ in the diagonal entries in order to keep the lattice determinant as small as possible.

Next we can construct the lattice $\mathcal{L}$ using the similar method of Herrmann-May, therefore, the lattice has triangular form, then the determinant $\det(\mathcal{L})$ is then simply the product of the entries on the diagonal:

$$\det(\mathcal{L}) = \prod_{i=1}^{n} X_i^{s_{x_i}} N^{s_N}$$

Let $d$ denote the dimension of $\mathcal{L}$, $t = r \cdot h + c$ ($h, c \in \mathbb{Z}$ and $0 \leq c < r$). A straightforward but tedious computation yields that

$$\begin{aligned}
s_{x_i} &= \binom{m+n}{m-1} = \frac{1}{(n+1)!} m^{n+1} + o(m^{n+1}) \\
s_N &= \sum_{k=0}^{t-1} \sum_{0 \leq \sum_{j=2}^{n} i_j \leq m-k} \lceil \frac{t-k}{r} \rceil \\
&= \sum_{i=1}^{h} \sum_{k=t-ir}^{t-ir+r-1} \sum_{0 \leq \sum_{j=2}^{n} i_j \leq m-k} i + \sum_{\substack{k=0 \\ c \geq 1}}^{c-1} \sum_{0 \leq \sum_{j=2}^{n} i_j \leq m-k} (h+1) \\
&= \sum_{i=0}^{h} \sum_{k=0}^{t-1-ir} \sum_{0 \leq \sum_{j=2}^{n} i_j \leq m-k} 1 \\
&= \frac{(n+1)\tau - 1 + (1-\tau)^{n+1}}{(n+1)!r} m^{n+1} + o(m^{n+1}) \\
d &= \binom{m+n}{m} = \frac{1}{n!} m^n + o(m^n)
\end{aligned}$$

By ignoring the low-order terms, the necessary condition to obtain $n$ equations over integer from Lemma 2 is given by

$$\det(\mathcal{L})^{\frac{1}{d-n+1}} < N^{\beta \tau m}$$

Let $X_i = N^{\gamma_i} (1 \leq i \leq n)$. Combining the values with the above condition, we obtain

$$\sum_{i=1}^{n} \gamma_i < \frac{1}{r} \left(1 - (1-\tau)^{n+1}\right) - \tau(n+1)(\frac{1}{r} - \beta)$$

By setting $\tau = 1 - \sqrt[n]{1 - r\beta}$, the condition reduces to

$$\sum_{i=1}^{n} \gamma_i < \frac{1}{r} \left(1 - (1-r\beta)^{\frac{n+1}{n}} - (n+1)(1-r\beta) \left(1 - \sqrt[n]{1-r\beta}\right)\right)$$

The running time is dominated by $LLL$ reduction, therefore, the same as the first approach, the total running time for this approach is polynomial in $\log N$ but exponential in $n$.

*Remark 4.* Our second approach is quite different from the BDH Method of 3.1 [2]. For comparison, our method is converted to find small roots of a univariate linear modular polynomial, whereas the BDH Method's target is a univariate modular polynomial of degree $r$. It is easy to see that our method is simpler and more effective.

**Table 2.** Comparisons of performances of our methods for different $r$ and $n$

| $r$ | $n$ | $\beta$ | Method of Sec. 5.1 | | | Method of Sec. 5.2 | | |
|---|---|---|---|---|---|---|---|---|
| | | | $(m,t)$ | $\dim(\mathcal{L})$ | $\sum_{i=1}^{n} \delta_i$ | $(m,t)$ | $\dim(\mathcal{L})$ | $\sum_{i=1}^{n} \delta_i$ |
| 2 | 2 | 1/3 | (10,2) | 139 | 0.1318 | (15,6) | 136 | 0.1500 |
| 2 | 2 | 1/3 | (14,2) | 277 | 0.1519 | (20,8) | 231 | 0.1597 |
| 2 | 3 | 1/3 | (6,1) | 119 | 0.0272 | (7,2) | 120 | 0.0761 |
| 2 | 3 | 1/3 | (8,1) | 249 | 0.0815 | (9,2) | 220 | 0.0942 |
| 3 | 2 | 1/4 | (10,1) | 102 | 0.1087 | (12,6) | 91 | 0.1030 |
| 3 | 2 | 1/4 | (15,2) | 282 | 0.1223 | (20,10) | 231 | 0.1271 |

## 5.3 Discussions

We present two different approaches to prove our main theorem. They both get the same asymptotic bounds, however, for a fixed lattice parameter $d$, their performances are significantly different. Table 2 gives some comparisons of two methods for small $r$, $n$.

One can observe that for the same scale of $d$, the method of 5.2 offers better theoretical results than the method of 5.1. The reason is that the method of 5.2 uses Herrmann-May's idea for solving modular polynomial equation, while the method of 5.1 uses the generalized strategy for solving integer polynomial equation. Thus the lattice dimension in the method of 5.1 is $\mathcal{O}(m^{n+1})$ in a certain parameters while that of the method of 5.2 is $\mathcal{O}(m^n)$, therefore, the method of 5.2 is more effective in practice.

We found several examples for the similar phenomenon in previous literature: Boneh-Durfee [1] showed that the secret key $d$ can be recovered from $(e, N)$ in polynomial-time if $d < N^{0.284}$ by solving a modular polynomial equation, while Ernst et al. [6] obtained the same bound by solving an integer polynomial equation; Ernst et al. [6] studied the problem of partial key exposed attack attacks on RSA, which utilizes the strategy of finding small root of the integer polynomial equation, while Sarkar et al. [19] offered the same asymptotic result as [6] for certain range using a modular polynomial equation. These showed that compared with the method of integer version, modular version has great advantage: obtaining better upper bounds ([1] improved to $N^{0.292}$), and better practical performances at the same cost (see [19] and our comparisons of two methods). However, the integer version also has strengths in some respects: irreplaceability by modular version (see [12]), and a broader range of solvable parameters ([6] and [19]). All the results enlighten us on how to choose the proper method in different situations. Moreover, the relation between these two methods is not clear yet, and we believe that many interesting facts are still hidden.

We show that for cryptographic applications, the RSA variant with modulus $N = p^r q$ should be used with more care. In particular, we show that less bits are sufficient to factorize $N$ as $r$ grows. In the case of the $p$ and $q$ with 1024 bits, when $r = 2$, we should know 564 bits of $p$ to factorize $N$, but when $r = 4$, we only require 413 bits of $p$ for factorization. Thus, based on our analysis, it seems secure to use $r = 2$ rather than $r = 4$.

## 6 Conclusion

We propose two heuristic lattice-based approaches to find small roots of linear modular polynomials $f(x_1, \ldots, x_n) = a_0 + a_1x_1 + \cdots + a_nx_n \bmod p$, where $p$ is an unknown divisor of some known $N$ and $N \equiv 0 \bmod p^r$. A direct implication of this results is factorization of multi-power RSA modulus $N = p^rq$ with given bits of $p$. We show that the knowledge of a $\frac{\ln(1+r)}{r}$-fraction of the bits of $p$ is sufficient to get the factorization. However, similar to [9], the running time of our algorithm heavily depends on the number of unknown blocks $n$, to be specific, our algorithm is polynomial-time only for $n = \mathcal{O}(\log \log N)$ blocks.

**Acknowledgments.** We would like to thank the anonymous reviewers for helpful comments. This work is supported by the National 973 Program of China under Grant No. 2011CB302400, IIEs Research Project on Cryptography under Grant No. Y3Z001C102, One Hundred Talents Project of the Chinese Academy of Sciences under Grant No. NSFC61100225, the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant No. XDA06010701.

## References

1. Boneh, D., Durfee, G.: Cryptanalysis of RSA with private key $d$ less than $n^{0.292}$. IEEE Transactions on Information Theory 46(4), 1339–1349 (2000)
2. Boneh, D., Durfee, G., Howgrave-Graham, N.: Factoring $n = p^rq$ for large $r$. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 326–337. Springer, Heidelberg (1999)
3. Boneh, D., Shacham, H.: Fast variants of RSA. CryptoBytes 5(1), 1–9 (2002)
4. Coppersmith, D.: Small solutions to polynomial equations, and low exponent RSA vulnerabilities. Journal of Cryptology 10(4), 233–260 (1997)
5. The EPOC and the ESIGN Algorithms. IEEE P1363: Protocols from Other Families of Public-Key Algorithms (1998), `http://grouper.ieee.org/groups/1363/StudyGroup/NewFam.html`
6. 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)
7. Halderman, J.A., Schoen, S.D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J.A., Feldman, A.J., Appelbaum, J., Felten, E.W.: Lest we remember: cold-boot attacks on encryption keys. Communications of the ACM 52(5), 91–98 (2009)
8. Herrmann, D.I.M.: Lattice-based Cryptanalysis using Unravelled Linearization. PhD thesis (2011)
9. 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)

10. Howgrave-Graham, N.: Finding small roots of univariate modular equations revisited. In: Darnell, M. (ed.) Cryptography and Coding 1997. LNCS, vol. 1355, pp. 131–142. Springer, Heidelberg (1997)
11. 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)
12. Jochemsz, E., May, A.: A polynomial time attack on RSA with private CRT-exponents smaller than $n^{0.073}$. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 395–411. Springer, Heidelberg (2007)
13. Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261(4), 515–534 (1982)
14. May, A.: New RSA vulnerabilities using lattice reduction methods. PhD thesis (2003)
15. May, A.: Using lll-reduction for solving RSA and factorization problems. In: The LLL algorithm, pp. 315–348 (2010)
16. Okamoto, T., Uchiyama, S.: A new public-key cryptosystem as secure as factoring. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 308–318. Springer, Heidelberg (1998)
17. Rivest, R.L., Shamir, A.: Efficient factoring based on partial information. In: Pichler, F. (ed.) EUROCRYPT 1985. LNCS, vol. 219, pp. 31–34. Springer, Heidelberg (1986)
18. Sarkar, S., Maitra, S.: Cryptanalysis of RSA with more than one decryption exponent. Information Processing Letters 110(8), 336–340 (2010)
19. Sarkar, S., Sen Gupta, S., Maitra, S.: Partial key exposure attack on RSA – improvements for limited lattice dimensions. In: Gong, G., Gupta, K.C. (eds.) INDOCRYPT 2010. LNCS, vol. 6498, pp. 2–16. Springer, Heidelberg (2010)
20. Takagi, T.: Fast rsa-type cryptosystem modulo $p^k q$. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 318–326. Springer, Heidelberg (1998)

# Toward Separating the Strong Adaptive Pseudo-freeness from the Strong RSA Assumption

Masayuki Fukumitsu, Shingo Hasegawa, Shuji Isobe, Eisuke Koizumi, and Hiroki Shizuya

Graduate School of Information Sciences, Tohoku University, 41 Kawauchi, Aoba-ku, Sendai, 980–8576 Japan
fukumitsu@isl.is.tohoku.ac.jp,
{hasegawa,iso,koizumi,shizuya}@cite.tohoku.ac.jp

**Abstract.** The notion of pseudo-freeness of a group was introduced by Hohenberger, and formalized by Rivest in order to unify cryptographic assumptions. Catalano, Fiore and Warinschi proposed the adaptive pseudo-free group as a generalization of pseudo-free group. They showed that the RSA group $\mathbb{Z}_N^\times$ is pseudo-free even if the adversary against pseudo-freeness is allowed to operate adaptively, provided that the adaptive behavior of the adversary is restricted by some specific parametric distribution. They also proposed the notion of strong adaptive pseudo-freeness in which the adaptive behavior of the adversary is not restricted. However, it remains open whether $\mathbb{Z}_N^\times$ is also strongly-adaptive pseudo-free under the strong RSA (SRSA) assumption.

In this paper, we give a negative circumstantial evidence for the question. We show that the SRSA assumption does not imply the strong adaptive pseudo-freeness of $\mathbb{Z}_N^\times$, as far as the algebraic reduction is concerned. The algebraic reduction means that the algorithm of the black-box reduction performs only group operations for elements in $\mathbb{Z}_N^\times$. Our result indicates that the strong adaptive pseudo-freeness for the RSA group $\mathbb{Z}_N^\times$ cannot be shown under the SRSA assumption, by employing only current proof techniques which are used in ordinary security proofs.

## 1 Introduction

**Background.** The notion of pseudo-free group was originally introduced by Hohenberger [17], and formalized by Rivest [24] for unifying cryptographic assumptions. Intuitively, a family $\{\mathbb{G}_N\}$ of computational groups is pseudo-free if $\{\mathbb{G}_N\}$ is indistinguishable from free groups $\mathcal{F}(A)$ generated by a set $A = \{a_1, a_2, \ldots, a_m\}$ of polynomially many symbols. Rivest [24] showed that several cryptographic assumptions including the RSA assumption [23], the discrete logarithm (DL, for short) assumption [12] and the strong RSA (SRSA, for short) assumption [5,13] hold on pseudo-free groups.

The indistinguishability of pseudo-free groups is described by using equations over $\mathcal{F}(A)$. A family $\{\mathbb{G}_N\}$ is pseudo-free if there exists no probabilistic polynomial time (PPT, for short) adversary $\mathcal{A}$ that on a given pair $(N, \alpha)$ of a group index $N$ and an assignment $\alpha : A \longrightarrow \mathbb{G}_N$ which interprets the symbols in $A$ as elements in $\mathbb{G}_N$, tries to find a witness pair $(\lambda, \psi)$ of an equation $\lambda$ and a solution $\psi$ such that $\lambda$ has no solution over the free group $\mathcal{F}(A)$, but the corresponding equation $\lambda_\alpha$ has a solution $\psi$ over $\mathbb{G}_N$, where $\lambda_\alpha$ is determined uniquely via the assignment $\alpha$. Existence of such a pair $(\lambda, \psi)$ witnesses that $\mathbb{G}_N$ is not a free group, since $\lambda_\alpha$ should have no solution if $\mathbb{G}_N$ is a free group.

Rivest [24] proposed a question whether one may extend the pseudo-freeness so that it can deal with adaptive adversaries. For this question, Catalano, Fiore and Warinschi [10] introduced the notion of the adaptive pseudo-free group, which is defined by a security game between a challenger and an adversary. Their adaptive pseudo-freeness means that there exists no PPT adversary that on a given game pair $(N, \alpha)$ determined by the challenger, cannot output a witness pair $(\lambda, \psi)$ as in the Rivest's static setting even if the adversary is allowed to adaptively receive a solution of equation queries. Here, the "adaptive" behavior of the adversary is restricted in the sense that the adversary is not allowed to unrestrainedly choose equations to be solved, but those are chosen according to some specified parametric distribution $\rho$ over the set of possible queries. Namely, on an input parameter $M$ sent from the adversary, the challenger chooses an equation according to the distribution $\rho(M)$, and replies the equation and the corresponding solution to the adversary. In [10], they showed that the RSA group $\mathbb{Z}_N^\times$ is adaptive pseudo-free under the SRSA assumption for some specific class of parametric distributions as in the static case [19,20].

They also proposed the notion of strong adaptive pseudo-freeness in [10], in which the adaptive behavior of the adversary is not restricted. However, it is not known that there exists an example of strongly-adaptive pseudo-free groups. In particular, it remains open whether or not $\mathbb{Z}_N^\times$ is strongly-adaptive pseudo-free, whereas $\mathbb{Z}_N^\times$ is shown to be adaptive pseudo-free under the SRSA assumption.

**Contribution.** In this paper, we give a negative circumstantial evidence for the open question. We formalize the strong adaptive pseudo-freeness in a reasonable setting, and show that the SRSA assumption does not imply the strong adaptive pseudo-freeness of the RSA group $\mathbb{Z}_N^\times$, as far as the black-box algebraic reduction (algebraic reduction, for short) is concerned. Informally, an algorithm is said to be algebraic with respect to a group $\mathbb{G}$ if the algorithm performs only group operations for elements in $\mathbb{G}$ and its execution can be easily traced. In particular, on any given input elements $y_1, \ldots, y_n \in \mathbb{G}$, any element $g \in \mathbb{G}$ computed in the execution of the algorithm belongs to the subgroup $\langle y_1, \ldots, y_n \rangle$ generated by the input elements, and moreover the expression $g = \prod_{i=1}^n y_i^{c_i}$ should be easily retrieved. We note that employing algebraic algorithms is not of exceedingly restricted setting, because most reductions concerning the pseudo-free group [10,18,19,20], and ordinary security proofs (e.g. [21,22]) are performed on algebraic algorithms. Our result means that the strong adaptive pseudo-freeness for the RSA group $\mathbb{Z}_N^\times$ cannot be shown under the SRSA assumption, by employing only current proof techniques which are used in ordinary security proofs. Comparing the result of [10], our result can be seen as a circumstantial evidence that the adaptive behavior of the pseudo-free adversary should be restricted in some sense.

**Overview.** We here describe the outline of the proof of our result. We show that if the SRSA assumption implies the strong adaptive pseudo-freeness of $\mathbb{Z}_N^\times$, then the SRSA assumption fails. We follow Micciancio's setting [19,20], namely for an SRSA instance $(N, y)$, $N$ is a product of distinct safe primes, and $y$ is restricted to quadratic residues modulo $N$. As for equations, we only consider univariate equations over the free abelian group $\mathcal{F}(A)$ generated by a set $A = \{a_i\}_{i=1}^m$ of polynomially many symbols as in [10]. Namely any equation $\lambda$ in a variable $x$ is expressed as $x^e = \prod_{i=1}^m a_i^{s_i}$ for some exponents $e, s_1, \ldots, s_m \in \mathbb{Z}$.

We firstly explain the situation where the SRSA assumption implies the strong adaptive pseudo-freeness of the RSA group $\mathbb{Z}_N^\times$. This situation is expressed by a *black-box reduction* $\mathcal{R}$ that breaks SRSA with a black-box access to an adversary $\mathcal{A}$ that violates the strong adaptive pseudo-freeness of $\mathbb{Z}_N^\times$. $\mathcal{R}$ plays the security game concerning strong adaptive pseudo-freeness in which $\mathcal{R}$ is the challenger. If such an adversary $\mathcal{A}$ exists, $\mathcal{R}$ should behave as follows. On any given SRSA instance $(N, y)$, $\mathcal{R}$ aims to find a solution $(x, E)$ such that $x^E \equiv y \pmod N$ by exploiting the adversary $\mathcal{A}$. $\mathcal{R}$ generates an assignment $\alpha : A \longrightarrow \mathbb{Z}_N^\times$ by choosing $\alpha(a)$ uniformly at random from the group $\mathrm{QR}_N$ of quadratic residues mod $N$ for each $a \in A$ following the setting in [10,19,20]. Then $\mathcal{R}$ gives the game pair $(N, \alpha)$ to $\mathcal{A}$. $\mathcal{A}$ adaptively queries an equation $\lambda : x^e = \prod_{i=1}^m a_i^{s_i}$ to the challenger, namely $\mathcal{R}$, at most polynomially many times. For each equation query, $\mathcal{R}$ should reply a solution $\psi$ over $\mathbb{Z}_N^\times$ for the interpreted equation $\lambda_\alpha : x^e = \prod_{i=1}^m \alpha(a_i)^{s_i}$ of $\lambda$ to $\mathcal{A}$. Eventually, $\mathcal{A}$ outputs a witness pair $(\lambda^*, \psi^*)$ of a nontrivial equation $\lambda^* : x^{e^*} = \prod_{i=1}^m a_i^{s_i^*}$ together with a solution $\psi^*$ of the interpreted equation $\lambda_\alpha^* : x^{e^*} = \prod_{i=1}^m \alpha(a_i)^{s_i^*}$ over $\mathbb{Z}_N^\times$. By using $(\lambda^*, \psi^*)$, $\mathcal{R}$ finds a solution for the SRSA instance $(N, y)$.

In the security game of the strong adaptive pseudo-freeness, we consider some reasonable settings as follows. The first one is that $\mathcal{R}$ always replies a trivial solution if a queried equation from $\mathcal{A}$ has a trivial solution. The second is that if $\mathcal{R}$ fails to find a correct answer for some queried equation despite that it has a solution over $\mathbb{Z}_N^\times$, $\mathcal{A}$ aborts and outputs the special symbol $\perp$ as in [9].

When $\mathcal{R}$ is algebraic with respect to $\mathrm{QR}_N$, all elements in $\mathrm{QR}_N$ produced in the execution of $\mathcal{R}$ belong to the subgroup $\langle y \rangle$ generated by the instance $y \in \mathrm{QR}_N$ given to $\mathcal{R}$. Then each element $\alpha(a_i) \in \mathrm{QR}_N$ is expressed as $\alpha(a_i) = y^{d_i}$ for some integer $d_i \in \mathbb{Z}$. Therefore, for any queried equation $\lambda : x^e = \prod_{i=1}^m a_i^{s_i}$ from $\mathcal{A}$, the interpreted equation $\lambda_\alpha : x^e = \prod_{i=1}^m \alpha(a_i)^{s_i}$ is of the form $x^e = y^D$ over $\mathbb{Z}_N^\times$, where $e, s_1, \ldots, s_m \in \mathbb{Z}$ and $D = \sum_{i=1}^m s_i d_i$.

We shall construct a *meta-reduction* $\mathcal{M}$ that breaks SRSA. Recall that if a strongly-adaptive pseudo-free adversary $\mathcal{A}$ exists, then the algorithm $\mathcal{R}$ can find a solution for SRSA by using $\mathcal{A}$. We therefore consider a PPT algorithm $\mathsf{Sim}_\mathcal{A}$ that simulates the strongly-adaptive pseudo-free adversary $\mathcal{A}$, or that looks for a solution $(x, E)$ for an SRSA instance $(N, y)$ during its execution. $\mathcal{M}$ aims to break SRSA by executing $\mathcal{R}$ and $\mathsf{Sim}_\mathcal{A}$. The idea for the construction of $\mathsf{Sim}_\mathcal{A}$ is as follows. Recall that the interpreted equation $\lambda_\alpha$ for any equation query $\lambda : x^e = \prod_{i=1}^m a_i^{s_i}$ from the adversary $\mathsf{Sim}_\mathcal{A}$ is expressed as $x^e = y^D$, where $D = \sum_{i=1}^m d_i s_i$, provided that $\mathcal{R}$ is algebraic. Suppose that for the equation

$\lambda$, the interpreted equation $\lambda_\alpha : x^e = y^D$ has a solution and $e \nmid D$. In this case, if $\mathsf{Sim}_{\mathcal{A}}$ queries the equation $\lambda$ to the challenger $\mathcal{R}$ and receives a correct solution $\psi \in \mathbb{Z}_N^\times$ for $\lambda_\alpha$, then it can easily find a solution $(x, E)$ for the given SRSA instance $(N, y)$ [20]. Thus it suffices that one can efficiently construct such equations. The construction of equations is as follows. Let $P'$ and $Q'$ be distinct primes such that $P = 2P' + 1$ and $Q = 2Q' + 1$ for the RSA modulus $N = PQ$. We aim to find an exponent $e$ such that $e \in \mathbb{Z}_{P'Q'}^\times$ and $e$ is strictly greater than $|D|$, the absolute value of $D$. Then we see that $e \nmid D$, and the element $y^{De^{-1}}$ is a solution for the equation $x^e = y^D$ since $y \in \mathrm{QR}_N$ and the order of $\mathrm{QR}_N$ is $P'Q'$.

**Related Works**

**Pseudo-free Groups.** One of the open questions posed in [24] is whether there exists an example of the pseudo-free group. For this question, Micciancio [19,20] showed that the RSA group $\mathbb{Z}_N^\times$ is pseudo-free under the SRSA assumption. Jhanwar and Barua [18] showed the pseudo-freeness for $\mathbb{Z}_N^\times$ under a condition slightly different from Micciancio's one. Anokhin [4] constructed a class of pseudo-free groups under the integer factoring assumption.

In [15,16], some variants of pseudo-free groups has been studied. Hasegawa, Isobe, Shizuya and Tashiro [15] also showed that the computational Diffie-Hellman assumption [24] holds on pseudo-free groups in some varied form.

Catalano, Fiore and Warinschi [10] constructed a secure signature scheme based on the adaptive pseudo-free groups. Their signature scheme is the first direct cryptographic application of the pseudo-free groups.

**Algebraic Algorithms.** Paillier and Vergnaud [21] introduced the notion of algebraic reduction to show an impossibility result for the universal unforgeability against the key only attack of the Schnorr's signature [25] under the DL assumption in the standard model. There are some impossibility results showed by using the algebraic reduction. Bresson, Monnerat and Vergnaud [9] showed the separation among the One-More DL problems [6,7]. Abe, Groth and Ohkubo [1] showed an impossibility result for the existential unforgeability against the chosen message attack of the structure-preserving signatures over bilinear groups under non-interactive assumptions. Abe, Haralambiev and Ohkubo [2] also studied the relationship between the length of the key and that of the commitment for the commitment schemes over bilinear groups.

The algebraic reduction is also used to show a lower bound for a loss factor of security reduction of the Schnorr's signature under the DL assumption in the random oracle model [21,14,26]. Villar [27] showed a lower bound for the loss factor, of reduction algorithms that solves the decisional Diffie-Hellman problem [8] with access to an adversary solving the rank problem.

## 2 Preliminaries

A prime $P$ is *safe* if $P = 2P' + 1$ for some prime $P'$. Let $\mathbb{N}_{\mathsf{RSA}}^{\mathsf{safe}}$ be the set of all RSA composites $N = PQ$ such that $P$ and $Q$ are distinct safe primes.

We assume that there are infinitely many safe primes. Although it is open whether or not this assumption holds, this assumption is widely believed to hold, e.g. [3]. For any $N \in \mathbb{N}$, we use $\mathbb{Z}_N$ and $\mathbb{Z}_N^{\times}$ to denote the residue ring $\mathbb{Z}/N\mathbb{Z}$ and its group of units, respectively. $\mathrm{QR}_N$ designates the group of quadratic residues mod $N$.

We write $x \in_{\mathrm{R}} X$ to denote that $x$ is chosen uniformly at random from the finite set $X$. A function $\nu(k)$ is *negligible* if for any polynomial $\mu$, there exists a constant $k_0$ such that $\nu(k) < 1/\mu(k)$ for any $k \geq k_0$. Throughout this paper, $k$ indicates a security parameter.

### 2.1 Pseudo-free Groups

**Computational Groups [24].** Let $\{\mathbb{G}_N\}_{N \in \mathcal{N}}$ be a family of finite groups indexed by an index set $\mathcal{N} = \cup_{k \geq 0} \mathcal{N}(k)$. We assume that each group index $N \in \mathcal{N}(k)$ and each element of $\mathbb{G}_N$ are expressed by words of polynomial length in $k$, respectively. $\{\mathbb{G}_N\}_{N \in \mathcal{N}}$ is said to be a family of *computational groups* if its group operations such as the composition (i.e. the group law) and the sampling can be efficiently executed. Note that the sampling is not necessarily uniform probability distribution. For the more formal definition, refer to [19,20,24]. Throughout this paper, we assume that $\mathbb{G}_N$ is abelian for any $N \in \mathcal{N}$.

**Free Abelian Groups.** Let $A = \{a_1, a_2, \ldots, a_m\}$ be a finite set of distinct symbols. For the set $A$, a new symbol set $A^{-1}$ is defined by $A^{-1} = \left\{a_1^{-1}, a_2^{-1}, \ldots, a_m^{-1}\right\}$. We denote by $\mathcal{F}(A)$ the free abelian group generated by the set $A$. The set $A$ is called the set of generators. For the detail, refer to [10,15,20,24].

**Equations Over Groups.** We consider only univariate equations over a free abelian group $\mathcal{F}(A)$ as in [10]. Let $x$ be a *variable*. An *equation in $x$ with symbols in $A$* is a pair $\lambda = (w_1, w_2)$, where $w_1$ is a word of the form $x^e$ with some exponent $e \in \mathbb{N}$, and $w_2$ is a word over $A$ of finite length. Since $\mathcal{F}(A)$ is abelian, we may assume that $w_2$ is expressed in a way that $w_2 = \prod_{i=1}^m a_i^{s_i}$ with some exponents $s_1, s_2, \ldots, s_m \in \mathbb{Z}$. Then we write the equation $\lambda = (w_1, w_2)$ by $x^e = \prod_{i=1}^m a_i^{s_i}$. We express the equation $\lambda : x^e = \prod_{i=1}^m a_i^{s_i}$ with the tuple $(e, \boldsymbol{s})$ of exponents, where $\boldsymbol{s} = (s_1, s_2, \ldots, s_m)$. Equations that have solutions in $\mathcal{F}(A)$ are *trivial*, others are *nontrivial*. The triviality of an equation $x^e = \prod_{i=1}^m a_i^{s_i}$ can be easily verified by the following lemma.

**Lemma 1 ([24]).** *An equation $x^e = \prod_{i=1}^m a_i^{s_i}$ is trivial over $\mathcal{F}(A)$ if and only if $e \mid s_i$ for any $1 \leq i \leq m$.*

Let $\mathbb{G}$ be any abelian group, and let $\alpha : A \longrightarrow \mathbb{G}$ be an *assignment* map that interprets each symbol $a \in A$ to a group element $\alpha(a) \in \mathbb{G}$. We write $\lambda_\alpha$ for the equation $\lambda : x^e = \prod_{i=1}^m a_i^{s_i}$ interpreted over $\mathbb{G}$ via $\alpha$, namely $\lambda_\alpha$ is the equation $x^e = \prod_{i=1}^m \alpha(a_i)^{s_i}$ over $\mathbb{G}$. $\psi \in \mathbb{G}$ is a *solution* for $\lambda_\alpha$ if $\psi^e = \prod_{i=1}^m \alpha(a_i)^{s_i}$ holds.

**Pseudo-Freeness.** A family $\{\mathbb{G}_N\}_{N \in \mathcal{N}}$ is *pseudo-free* if for any set $A$ of polynomial size in $k$, there exists no PPT adversary $\mathcal{A}$ that on a pair $(N, \alpha)$ of a group index $N \in \mathcal{N}(k)$ and an assignment $\alpha : A \longrightarrow \mathbb{G}_N$, outputs a *witness pair* $(\lambda, \psi)$

of an equation $\lambda$ and a solution $\psi$ such that $\lambda$ is nontrivial over $\mathcal{F}(A)$ but $\lambda_\alpha$ has a solution $\psi$ over $\mathbb{G}_N$, with nonnegligible probability, where the probability is taken over the random choice of $N \in \mathcal{N}(k)$, that of $\alpha(a)$ according to the designated sampling algorithm over $\mathbb{G}_N$ for each $a \in A$, and the coin flips of $\mathcal{A}$.

## 2.2 Strongly-Adaptive Pseudo-free Groups

Catalano, Fiore and Warinschi [10] introduced the notion of the adaptive pseudo-freeness as a generalization of the Rivest's "static" pseudo-freeness in order to handle adaptive adversaries. Intuitively, the adaptive pseudo-freeness means that no PPT adversary outputs a witness pair $(\lambda, \psi)$ as in the static case with non-negligible probability, even if he is allowed to adaptively receive a solution of a queried equation polynomially many times. In their setting, the queried equations are chosen according to some specified parametric distribution. They also informally define in [10] the *strong* adaptive pseudo-freeness in a way that there is no such restriction, namely the adversary is allowed to freely choose his queries. Following [10], we formally define the strong version of adaptive pseudo-freeness by the strongly-adaptive pseudo-free (SAPF, for short) game.

Before defining the game, we need to formalize the triviality of an equation that the adversary outputs in the adaptive setting [10]. In the static setting, the adversary is only required to output an equation that has no solution over $\mathcal{F}(A)$. However, the adaptive setting requires a more sophisticated treatment, because the adversary has already learned some pairs of an equation and its solution via his adaptive queries. Intuitively, an equation $\lambda^*$ (output by the adversary) is *trivial* with respect to the queried equations set $\Lambda = \{\lambda^{(1)}, \lambda^{(2)}, \ldots, \lambda^{(q)}\}$ if $\lambda^*$ can be efficiently derived from the equations in $\Lambda$ by using the group laws and some basic algebraic procedures. For the detail, refer to Section 3.2 of [10]. We note that when $\Lambda = \emptyset$, the triviality of equations is exactly equivalent to that in the static case.

**The Strongly-Adaptive Pseudo-free Game.** The SAPF game between the challenger and the adversary is a variant of the adaptive pseudo-free game defined in [10]. Let $\mathcal{G} = \{\mathbb{G}_N\}_{N \in \mathcal{N}}$ be a computational group family and $A = \{a_1, a_2, \ldots, a_m\}$ be a set of $m = \mathrm{poly}(k)$ symbols.

Setup. The challenger chooses a group index $N \in \mathcal{N}(k)$ at random. Then, he sets an assignment $\alpha : A \longrightarrow \mathbb{G}_N$ by choosing an element $\alpha(a) \in \mathbb{G}_N$ at random according to the designated sampling algorithm. The adversary is given the *game pair* $(N, \alpha)$ of the descriptions of $\mathbb{G}_N$ and the assignment $\alpha$.

Equations queries. The adversary $\mathcal{A}$ is allowed to adaptively query to the challenger on equations, and to receive their solutions. For each $t$-th query, $\mathcal{A}$ chooses an arbitrary equation $\lambda^{(t)} = (e_t, \boldsymbol{s}^t)$ and hands it to the challenger. The challenger returns a correct solution $\psi_t \in \mathbb{G}_N$ for the interpreted equation $\lambda_\alpha^{(t)}$ : $x^{e_t} = \prod_{i=1}^{m} \alpha(a_i)^{s_i^t}$ to the adversary.

Challenge. The adversary outputs a witness pair $(\lambda^*, \psi^*)$ of an equation $\lambda^* = (e^*, \boldsymbol{s}^*)$ and a solution $\psi^*$ of the interpreted equation $\lambda_\alpha^*$ over $\mathbb{G}_N$. The challenger outputs 1 if $\lambda^*$ is nontrivial with respect to $\Lambda$, the set of queried equations and corresponding solutions appeared in Equations queries phase, and $\psi^*$ is a correct solution for $\lambda^*_\alpha$.

An adversary $\mathcal{A}$ is said to $(q, \epsilon_{\mathcal{A}})$*-win the strongly-adaptive pseudo-free* (*SAPF, for short*) *game for the family* $\mathcal{G}$ if $\mathcal{A}$ makes at most $q$ queries, and the probability that the challenger outputs 1 in the game is at least $\epsilon_{\mathcal{A}}$, where the probability is taken over the random choices of the index $N \in \mathcal{N}(k)$ and the element $\alpha(a) \in \mathbb{G}_N$ for each $a \in A$, and the coin flips of $\mathcal{A}$.

**Definition 1 (Strongly-Adaptive Pseudo-Free Groups).** *A family $\mathcal{G}$ of computational groups is* strongly-adaptive pseudo-free*, if for any set $A$ of polynomial size in $k$, any polynomial $q$ in $k$, and any nonnegligible function $\epsilon_{\mathcal{A}}$ in $k$, there exists no PPT adversary $\mathcal{A}$ that $(q, \epsilon_{\mathcal{A}})$-wins the SAPF game for $\mathcal{G}$.*

*Remark:* In the adaptive pseudo-free game given in [10], the equation queries of the adversary $\mathcal{A}$ is determined by some specific parametric distribution. On the other hand, in the SAPF game, $\mathcal{A}$ can freely choose his queries. It is therefore necessary to consider the situation where $\mathcal{A}$ queries an equation which has no solution over $\mathbb{G}_N$. In this paper, we assume that the challenger outputs the special symbol $\perp$ provided that a queried equation has no solution over $\mathbb{G}_N$.

## 2.3 Algebraic Algorithms

The concept of algebraic algorithm was introduced by Paillier and Vergnaud [21]. Intuitively, an algorithm $\mathcal{R}$ is *algebraic with respect to a computational group* $\mathbb{G}$ if $\mathcal{R}$ performs only the group operation for the elements in $\mathbb{G}$ and the execution of $\mathcal{R}$ can be easily traced. In particular, on any input elements $y_1, \ldots, y_n \in \mathbb{G}$, any element $g \in \mathbb{G}$ computed in the execution of $\mathcal{R}$ belongs to the subgroup $\langle y_1, \ldots, y_n \rangle$ generated by the input elements, and moreover the expression $g = \prod_{i=1}^{n} y_i^{c_i}$ should be easily retrieved.

We follow the formal definition given in [26]. An algorithm $\mathcal{R}$ is *algebraic* for a computational group family $\{\mathbb{G}_N\}_{N \in \mathcal{N}}$, if the following algorithm Extract is provided. Extract receives any tuple $(N, y_1, \ldots, y_n, \mathsf{aux}, g, \omega)$ as input, where $N \in \mathcal{N}$ is a group index, $y_1, \ldots, y_n \in \mathbb{G}_N$ are elements that are given to $\mathcal{R}$ as input, aux is any word given to $\mathcal{R}$ as an auxiliary input, $g \in \mathbb{G}_N$ is a *target* group element and $\omega$ denotes a random coin used in $\mathcal{R}$. Then Extract finds a tuple $(c_1, \ldots, c_n)$ of exponents such that $g = \prod_{i=1}^{n} y_i^{c_i}$, provided that $g$ is actually produced in the execution of $\mathcal{R}$ on the input tuple $(N, y_1, \ldots, y_n, \mathsf{aux})$ with the random coin $\omega$. If there is no correct exponents $(c_1, \ldots, c_n)$, then Extract may output any word. Extract is required to run in polynomial time in the running time of $\mathcal{R}$. In particular, if $\mathcal{R}$ runs in polynomial time in the security parameter $k$, then Extract should run in polynomial time in $k$.

We consider an algebraic algorithm $\mathcal{R}$ that has an access to an oracle $\mathcal{A}$. We assume that Extract correctly computes a tuple $(c_1, \ldots, c_n)$ for the given target $g \in \mathbb{G}_N$ that is produced after the $t$-th query if all the correct answers for the previous $t$ queries are provided as a part of the auxiliary input aux. Note that if the target element $g \in \mathbb{G}_N$ is produced before the first query, it is not required to provide any additional inputs to $\mathsf{Extract}$ as in [9].

## 2.4 Strong RSA Assumption

An RSA modulus generator $\mathsf{GenMod}$ outputs a $k$-bit RSA composite $N \in \mathbb{N}^{\text{safe}}_{\mathsf{RSA}}$ on each input $1^k$. We follow the setting of [10,11,20] that $N$ is restricted to a product of two distinct safe primes. The SRSA assumption is defined as follows:

**Definition 2 (Strong RSA).** *A probabilistic algorithm $\mathcal{A}$ $\epsilon$-*breaks SRSA *if*

$$\Pr\bigl[x^E \equiv y \pmod N : N \leftarrow \mathsf{GenMod}(1^k), y \in_{\mathrm{R}} \mathrm{QR}_N, (x, E) \leftarrow \mathcal{A}(N, y)\bigr] \geq \epsilon,$$

*where $x \in \mathbb{Z}_N^\times$ and $E > 1$ is a natural number, and the probability is taken over the coin flips of* $\mathsf{GenMod}$ *and $\mathcal{A}$, and the uniformly random choice $y$ from $\mathrm{QR}_N$. The pair $(N, y)$ is called* an SRSA instance*. Then, the* strong RSA (SRSA, for short) assumption *holds if for any nonnegligible function $\epsilon$, there exists no PPT algorithm $\mathcal{A}$ that $\epsilon$-breaks SRSA.*

We can restrict ourselves to the case where $y$ is an element in quadratic residues mod $N$. Cramer and Shoup [11] mentioned that this is not an essential restriction by showing that breaking the SRSA assumption for $y \in \mathrm{QR}_N$ implies breaking the SRSA assumption for an arbitrary $y \in \mathbb{Z}_N^\times$. We employ the following lemma.

**Lemma 2 ([20]).** *Let $N \in \mathbb{N}^{\text{safe}}_{\mathsf{RSA}}$ be any $k$-bit RSA composite. Let $e$ and $D$ be any integers of binary length at most polynomial in $k$, and let $\psi \in \mathbb{Z}_N^\times$ and $y \in \mathrm{QR}_N$ such that $\psi^e \equiv y^D \pmod N$. If $e \nmid D$, then a pair $(x, E)$ such that $x^E \equiv y \pmod N$ can be found in polynomial time in $k$ on the tuple $(N, e, D, \psi, y)$.*

## 3 Main Theorem

In this section, we show that the SRSA assumption does not imply the strong adaptive pseudo-freeness for the family $\{\mathbb{Z}_N^\times\} = \{\mathbb{Z}_N^\times\}_{N \in \mathcal{N}}$ of the RSA groups with respect to the algebraic reduction, where $\mathcal{N} = \mathbb{N}^{\text{safe}}_{\mathsf{RSA}}$. In this paper, we adopt any sampling algorithm for the family $\{\mathbb{Z}_N^\times\}$ which chooses an element $g \in \mathrm{QR}_N$ at random in a way that the probability distribution is statistically close to the uniform distribution. For example, if a generator $y$ of $\mathrm{QR}_N$ is fixed, then such a sampling algorithm can be easily done by choosing an exponent $d \in_{\mathrm{R}} \{i\}_{i=0}^{B-1}$ with sufficiently large $B$ and set $g = y^d$. (See Lemma 2 of [20].)

Before stating the main result, we give a remark on the strong adaptive pseudo-freeness of the RSA groups $\{\mathbb{Z}_N^\times\}$. In $\mathsf{Equations\ queries}$ phase of the SAPF game, the adversary is allowed to choose an equation query arbitrarily. However, for $\mathbb{Z}_N^\times$, this setting does not seem to work properly without care. For example, assume that the adversary $\mathcal{A}$ queries the equation $(2, (2, 0, \ldots, 0))$, namely $x^2 = a_1^2$, and receives a solution $c \in \mathbb{Z}_N^\times$ such that $c \neq \pm\alpha(a_1)$. Then $\mathcal{A}$

can easily factor $N$. Once $N$ is factored, the adversary can easily find a witness pair $(\lambda^*, \psi^*)$. Therefore, $\mathbb{Z}_N^\times$ would not be strongly-adaptive pseudo-free in the strict sense. In this paper, we exclude such a situation. Instead, for any equation query $\lambda : x^e = \prod_{i=1}^m a_i^{s_i}$, the challenger is assumed to return a *canonical solution* $\psi'$ for the interpreted equation $\lambda_\alpha$, namely $\psi'$ is a solution for the interpreted equation $\lambda'_\alpha$ of the reduced equation $\lambda' : x^{e'} = \prod_{i=1}^m \alpha(a_i)^{s'_i}$, where $e' = e/d$, $s'_i = s_i/d$ and $d = \gcd(e, s_1, \ldots, s_m)$. For example, the challenger always returns the canonical solution $\psi = \alpha(a_1)$ on the query $x^2 = a_1^2$.

**The Situation.** We describe our setting for the situation that *the SRSA assumption implies the strong adaptive pseudo-freeness for the family* $\{\mathbb{Z}_N^\times\}$ *of the RSA groups*. We formalize this statement by the following contrapositive setting similarly to [9,21]: there exist a PPT algorithm $\mathcal{R}$, nonnegligible functions $\epsilon_0$ and $\epsilon_1$ and a polynomial $q$ such that for any SAPF adversary $\mathcal{A}$ who $(q, \epsilon_1)$-wins the SAPF game on the family $\{\mathbb{Z}_N^\times\}$, $\mathcal{R}$ $\epsilon_0$-breaks SRSA with a black-box access to the adversary $\mathcal{A}$. We may assume without loss of generality that $q \geq 2$. Through the black-box access, $\mathcal{R}$ would play the SAPF game with the adversary $\mathcal{A}$ in which $\mathcal{R}$ is placed at the challenger's position.

Given an SRSA instance $(N, y)$, $\mathcal{R}$ follows Setup phase of the SAPF game, namely $\mathcal{R}$ chooses a game pair $(N, \alpha)$, especially the assignment map $\alpha$ is chosen by selecting $\alpha(a)$ uniformly at random from $\mathrm{QR}_N$ for each $a \in A$. We assume as in [10,19,20] that the index $N$ of the game pair is always the same as the modulus $N$ of the given SRSA instance. Moving to Equations queries phase, $\mathcal{A}$ makes equation queries $\lambda = (e, \boldsymbol{s})$ at most $q$ times. Since $\mathcal{R}$ is now playing the role of the challenger, $\mathcal{R}$ replies the answer for each of the queries, but $\mathcal{R}$ may fail to reply the correct answer because the reduction $\mathcal{R}$ is polynomial-time bounded. Eventually, the game is completed with $\mathcal{A}$'s output: a "winning" witness pair $(\lambda^*, \psi^*)$ of the SAPF game, or "losing" symbol $\perp$. After the game, $\mathcal{R}$ would find a correct solution $(x, E)$ for the given SRSA instance $(N, y)$ with nonnegligible probability $\epsilon_0$.

In this paper, we force the reduction $\mathcal{R}$ to be *algebraic with respect to the group* $\mathrm{QR}_N$ *for any* $N \in \mathcal{N}$. Consequently, any element $g \in \mathrm{QR}_N$ produced in the execution of $\mathcal{R}$ is generated by the given SRSA instance $y$ and the expression $g = y^d$ is easily retrieved by the extraction algorithm Extract, provided that $g$ is actually computed in the execution of $\mathcal{R}(N, y)$. In particular, for the assignment $\alpha$ and each $a \in A$, $\alpha(a)$ is of the form $\alpha(a) = y^d$ and the exponent $d$ can be easily retrieved.

We now ready to state our main theorem that the SRSA assumption does not imply the strong adaptive pseudo-freeness for the family $\{\mathbb{Z}_N^\times\}$ of the RSA groups with respect to the algebraic reduction.

**Theorem 1.** *If the SRSA assumption implies the strong adaptive pseudo-freeness for the family* $\{\mathbb{Z}_N^\times\}$ *of the RSA groups with respect to the algebraic reduction, then the SRSA assumption does not hold.*

*Proof.* Assume that the SRSA assumption implies the strong adaptive pseudo-freeness for the family $\{\mathbb{Z}_N^\times\}$ of the RSA groups with respect to the algebraic reduction. Then, there exist a PPT algorithm $\mathcal{R}$, nonnegligible functions $\epsilon_0$ and $\epsilon_1$ and a polynomial $q \geq 2$ such that $\mathcal{R}$ is algebraic with respect to $\mathrm{QR}_N$ for any $N \in \mathcal{N}$, and $\mathcal{R}$ $\epsilon_0$-breaks SRSA with a black-box access to any PPT adversary $\mathcal{A}$ who $(q, \epsilon_1)$-wins the SAPF game for the family $\left\{\mathbb{Z}_N^\times\right\}$. This means that for any security parameter $k$, $\mathcal{R}$ breaks SRSA with at least nonnegligible probability $\epsilon_0$ for an instance $(N, y)$ of an RSA modulo $N$ generated by $\mathsf{GenMod}(1^k)$ and an element $y \in \mathrm{QR}_N$ chosen uniformly at random. We may assume without loss of generality that $y$ is a generator of $\mathrm{QR}_N$, because a solution for an SRSA instance $(N, y)$ can be easily found when $y$ is not a generator of $\mathrm{QR}_N$ [20].

**Construction of the Meta-reduction $\mathcal{M}$.** We shall construct a PPT algorithm $\mathcal{M}$ that breaks SRSA with no oracle access at least nonnegligible probability. The reduction $\mathcal{R}$ $\epsilon_0$-breaks SRSA with the black-box access to any SAPF adversary who $(q, \epsilon_1)$-wins the SAPF game. Therefore, we provide for the reduction $\mathcal{R}$ a simulator $\mathsf{Sim}_{\mathcal{A}}$ that plays a $(q, \epsilon_1)$-winning SAPF adversary's role to $\mathcal{R}$. In other words, from the $\mathcal{R}$'s viewpoint, $\mathsf{Sim}_{\mathcal{A}}$ looks like a "real" $(q, \epsilon_1)$-winning adversary, namely it actually $(q, \epsilon_1)$-wins the SAPF game provided that the reduction $\mathcal{R}$ is supposed to be *ideal* as a challenger in a sense that $\mathcal{R}$ always replies a correct answer to each query from $\mathsf{Sim}_{\mathcal{A}}$. If $\mathsf{Sim}_{\mathcal{A}}$ is set to the adversary's position, then $\mathcal{R}$ would $\epsilon_0$-break SRSA via playing the SAPF game with $\mathsf{Sim}_{\mathcal{A}}$. Thus, our meta-reduction $\mathcal{M}$ is constructed by involving $\mathcal{R}$ and $\mathsf{Sim}_{\mathcal{A}}$.

For the algorithm $\mathsf{Sim}_{\mathcal{A}}$, we may assume without loss of generality that the final output of $\mathsf{Sim}_{\mathcal{A}}$ can be a correct solution $(x, E)$ for the given SRSA instance $(N, y)$ if $\mathsf{Sim}_{\mathcal{A}}$ fortunately finds it, instead of a witness pair $(\lambda^*, \psi^*)$ of the SAPF game. This does not lower the success probability $\epsilon_0$ of the reduction $\mathcal{R}$. Thus, our simulator $\mathsf{Sim}_{\mathcal{A}}$ is to be a PPT algorithm that on a given instance $(N, \alpha, y, \omega)$, where $(N, y)$ is an SRSA instance, $\alpha$ is a game pair of the SAPF game presented by $\mathcal{R}$ and $\omega$ is a random coin used by $\mathcal{R}$ when $\mathcal{R}$ generates the assignment map $\alpha$, responds one of the following items (I)–(III):

(I) $\mathsf{Sim}_{\mathcal{A}}$ finds a witness pair $(\lambda^*, \psi^*)$ of a nontrivial equation $\lambda^*$ and a solution $\psi^*$ for the interpreted equation $\lambda^*_\alpha$.

(II) In a fortunate case, $\mathsf{Sim}_{\mathcal{A}}$ may find a solution $(x, E)$ for the given SRSA instance $(N, y)$ in its execution. If $\mathsf{Sim}_{\mathcal{A}}$ meets its fortunate case, then $\mathsf{Sim}_{\mathcal{A}}$ outputs the solution $(x, E)$.

(III) $\mathsf{Sim}_{\mathcal{A}}$ may abort with the output $\perp$ in an unfortunate case.

We note that $\mathsf{Sim}_{\mathcal{A}}$ is required only to come into either the case (I) or the case (II) with at least probability $\epsilon_1$ and within $q$ queries to $\mathcal{R}$ provided that $\mathcal{R}$ is ideal as a challenger. By using $\mathsf{Sim}_{\mathcal{A}}$, the algorithm $\mathcal{M}$ is constructed as in Fig. 1. If $\mathsf{Sim}_{\mathcal{A}}$ is constructed in that way, $\mathcal{R}$ $\epsilon_0$-breaks SRSA and consequently the resulting algorithm $\mathcal{M}$ will succeed with probability at least $\epsilon_0$.

**Construction of $\mathsf{Sim}_{\mathcal{A}}$.** In order to construct the algorithm $\mathcal{M}$, it suffices to construct the simulator $\mathsf{Sim}_{\mathcal{A}}$. Since $\mathcal{R}$ is algebraic with respect to $\mathrm{QR}_N$ for any $N$, there exists a polynomial time algorithm $\mathsf{Extract}$ that on a tuple $(N, y, g, \omega)$, where $g$ is a target element in $\mathrm{QR}_N$ that is produced in the execution of $\mathcal{R}$ on the input $(N, y)$ with the random coin $\omega$, returns an exponent $d$ such that $g = y^d$.

On any given SRSA instance $(N, y)$,

(M-1) $\mathcal{M}$ executes $\mathcal{R}$ on the instance $(N, y)$ with using a random coin $\omega$ chosen by $\mathcal{M}$.
(M-2) When $\mathcal{R}$ submits a game pair $(N, \alpha)$ to the adversary, $\mathcal{M}$ executes $\mathsf{Sim}_{\mathcal{A}}$ on the tuple $(N, \alpha, y, \omega)$. Then $\mathcal{R}$ and $\mathsf{Sim}_{\mathcal{A}}$ play the SAPF game.
(M-3) $\mathsf{Sim}_{\mathcal{A}}$ outputs a response $c \in \{(\lambda^*, \psi^*), \perp, (x, E)\}$, and halts.
(M-4) After receiving the response $c$, $\mathcal{M}$ behaves as follows:
(M-4a) either $c = (\lambda^*, \psi^*)$ or $c = \perp$: $\mathcal{M}$ continues to simulate $\mathcal{R}$, and outputs the final output of $\mathcal{R}$, and halts; and
(M-4b) $c = (x, E)$: $\mathcal{M}$ outputs $(x, E)$, and halts.

**Fig. 1.** Configuration of $\mathcal{M}$

We involve $\mathsf{Extract}$ in the construction of $\mathsf{Sim}_{\mathcal{A}}$. The algorithm $\mathsf{Sim}_{\mathcal{A}}$ is depicted in **Algorithm 1**. On a given input tuple

– if $\alpha(a_i) = 1 \in \mathrm{QR}_N$ for all $1 \leq i \leq m$, then $\mathsf{Sim}_{\mathcal{A}}$ outputs a correct witness pair $(\lambda^*, \psi^*)$ in the step (A-1) (the case (I)); or
– if $\alpha(a_{i_0}) \neq 1$ for some $i_0$, then in the step (A-2), $\mathsf{Sim}_{\mathcal{A}}$ attempts to find a solution $(x, E)$ of the SRSA instance $(N, y)$ by interacting to the challenger $\mathcal{R}$: if $\mathsf{Sim}_{\mathcal{A}}$ has found a solution $(x, E)$ for SRSA, then $\mathsf{Sim}_{\mathcal{A}}$ outputs the solution $(x, E)$ (the case (II)), or otherwise, $\mathsf{Sim}_{\mathcal{A}}$ outputs $\perp$ (the case (III)).

**Correctness of $\mathsf{Sim}_{\mathcal{A}}$.** (A-1) We consider the case where $\alpha(a_i) = 1$ for all $1 \leq i \leq m$. Note that the triviality of the equations in this case is equivalent to the one in the static case, because $\mathsf{Sim}_{\mathcal{A}}$ makes no query to the challenger $\mathcal{R}$. Namely, the triviality of the equation merely means that it has no solution over the free group $\mathcal{F}(A)$. Therefore, the equation $\lambda^* = (2, (3, \ldots, 3))$ is nontrivial by Lemma 1. Moreover, because $\alpha(a_i) = 1$ for all $1 \leq i \leq m$, $\psi = 1 \in \mathbb{Z}_N^\times$ is a solution for the interpreted equation $\lambda_\alpha^*$. Thus the output $(\lambda^*, \psi^*)$ of $\mathsf{Sim}_{\mathcal{A}}$ is a correct witness pair. This is the case (I).

(A-2) We next consider the case where $\alpha(a_{i_0}) \neq 1$ for some $1 \leq i_0 \leq m$. We show that $\mathsf{Sim}_{\mathcal{A}}$ outputs a solution $(x, E)$ for the given SRSA instance $(N, y)$ by interacting to the challenger $\mathcal{R}$ or outputs $\perp$. Since the assignment $\alpha$ is generated before the game pair $(N, \alpha)$ is given to $\mathcal{A}$, for each $1 \leq i \leq m$, $\mathsf{Sim}_{\mathcal{A}}$ can retrieve an exponent $d_i$ such that $\alpha(a_i) = y^{d_i}$ by executing $\mathsf{Extract}$ on the tuple $(N, y, \alpha(a_i), \omega)$. Note that the exponent $d_i$ exists, because the element $\alpha(a_i) \in \mathrm{QR}_N$ is produced by the algebraic algorithm $\mathcal{R}$, and hence it belongs to the subgroup $\langle y \rangle$ generated by $y \in \mathrm{QR}_N$. Therefore, for any equation $\lambda = (e, (s_1, \ldots, s_m))$, the interpreted equation $\lambda_\alpha$ over $\mathbb{Z}_N^\times$ is expressed in a way that $x^e = \prod_{i=1}^m \alpha(a_i)^{s_i} = \prod_{i=1}^m \left(y^{d_i}\right)^{s_i} = y^{\sum_{i=1}^m d_i s_i}$. We say that an equation $\lambda = (e, (s_1, \ldots, s_m))$ is *good* if the interpreted equation $\lambda_\alpha : x^e = y^D$, where $D = \sum_{i=1}^m d_i s_i$, has a solution $\psi \in \langle y \rangle$ and $e \nmid D$. Note that if there is no solution of $\lambda_\alpha$ in $\langle y \rangle$, $\mathcal{R}$ cannot find the solution of $\lambda_\alpha$, although it has a solution

**Algorithm 1.** $\mathsf{Sim}_{\mathcal{A}}$.

```
Input. a tuple $(N, \alpha, y, \omega)$.
Output. one of the following: a pair $(\lambda^*, \psi^*)$ of a nontrivial equation and its interpreted
solution, a pair $(x, E)$ such that $x^E \equiv y \pmod N$ or the special symbol $\perp$.

(A-1) If $\alpha(a_i) = 1$ for all $1 \le i \le m$, then set $\lambda^* \leftarrow (2, (3, \dots, 3))$ and $\psi^* \leftarrow 1$, output
the tuple $(\lambda^*, \psi^*)$, and halt.
(A-2) If $\alpha(a_{i_0}) \ne 1$ for some $i_0$, then for each $1 \le i \le m$, retrieve an exponent $d_i$ of
the element $\alpha(a_i) \in \mathrm{QR}_N$ such that $\alpha(a_i) = y^{d_i}$ by executing $\mathsf{Extract}(N, y, \alpha(a_i), \omega)$.

(A-2a) Choose $s_1, \dots, s_m \in_{\mathrm{R}} \mathbb{Z}_N$, and set $D \leftarrow \sum_{i=1}^m d_i s_i$.
    If $D = 0$, then reset $s_{i_0} \leftarrow s_{i_0} + 1$, and $D \leftarrow \sum_{i=1}^m d_i s_i$.
    Set $\boldsymbol{s} \leftarrow (s_1, \dots, s_m)$.
(A-2b) Set $e_1 \leftarrow |D| + 1$ and $e_2 \leftarrow |D| + 2$.
    If $y^{e_1 e_2} \equiv 1 \pmod N$, then output the pair $(x, E) = (y, e_1 e_2 + 1)$, and halt.
    Else, proceed to (A-2c).
(A-2c) For each $t \in \{1, 2\}$, submit the equation $\lambda^{(t)} = (e_t, \boldsymbol{s})$ to $\mathcal{R}$, and then receive a
    solution $\psi_t$ from $\mathcal{R}$.
    If $\psi_{t_0}$ is a correct solution for $\lambda_\alpha^{(t_0)}$ for some $t_0 \in \{1, 2\}$, then compute a pair $(x, E)$
    such that $x^E \equiv y \pmod N$ by Lemma 2, output $(x, E)$, and halt.
    Else, output $\perp$, and halt.
```

in $\mathbb{Z}_N^\times$. This is because $\mathcal{R}$ is algebraic. If $\mathsf{Sim}_{\mathcal{A}}$ queries a good equation $\lambda$ to the challenger $\mathcal{R}$ and succeeds to receive a correct solution $\psi \in \langle y \rangle$ for $\lambda_\alpha$, then $\mathsf{Sim}_{\mathcal{A}}$ can find a correct solution $(x, E)$ for the SRSA instance $(N, y)$ by Lemma 2. This is the case (II).

We now show that one of the following events occurs: (i) at least one of the equation queries $\lambda^{(1)} = (e_1, \boldsymbol{s})$ and $\lambda^{(2)} = (e_2, \boldsymbol{s})$ generated in the steps (A-2a) and (A-2b) is a good equation, or (ii) a correct solution $(x, E)$ for the SRSA instance $(N, y)$ is found.

It is easy to observe that the integer $D = \sum_{i=1}^m d_i s_i$ generated in (A-2a) is not zero. In the step (A-2b), $\mathsf{Sim}_{\mathcal{A}}$ computes integers $e_1 = |D| + 1$ and $e_2 = |D| + 2$. If $y^{e_1 e_2} \equiv 1 \pmod N$, then it is obvious that the pair $(x, E) = (y, e_1 e_2 + 1)$ is a solution for the given SRSA instance $(N, y)$. Otherwise, by the claims **Claim 1** and **Claim 2**, it is shown that there exists an index $t_0 \in \{1, 2\}$ such that for the equation $\lambda^{(t_0)} = (e_{t_0}, \boldsymbol{s})$, the interpreted equation $\lambda_\alpha^{(t_0)}$ has a solution over $\mathbb{Z}_N^\times$ and $e_{t_0} \nmid D$, where $\boldsymbol{s} = (s_1, \dots, s_m)$ has been generated in (A-2a).

**Claim 1.** *Assume that* $y^{e_1 e_2} \not\equiv 1 \pmod N$. *Then, there exists an index* $t_0 \in \{1, 2\}$ *such that the interpreted equation* $\lambda_\alpha^{(t_0)} : x^{e_{t_0}} = y^D$ *has a solution* $\psi_{t_0} \in \langle y \rangle$ *for the equation* $\lambda^{(t_0)} = (e_{t_0}, \boldsymbol{s})$.

**Claim 2.** *The integers* $e_1$ *and* $e_2$ *found in (A-2b) satisfy* $e_1 \nmid D$ *and* $e_2 \nmid D$.

In the step (A-2c), for each $t \in \{1, 2\}$, $\mathsf{Sim}_{\mathcal{A}}$ queries the equation $\lambda^{(t)} = (e_t, \boldsymbol{s})$ to the challenger $\mathcal{R}$, and then receives the solution $\psi_t$ over $\mathbb{Z}_N^\times$ of the interpreted equation $\lambda_\alpha^{(t)} : x^{e_t} = y^D$.

When $\psi_{t_0}$ is a correct solution for $\lambda_\alpha^{(t_0)}$ for some $t_0 \in \{1,2\}$, we have $\psi_{t_0}^e \equiv y^D \pmod{N}$. Since $e_{t_0} \nmid D$ by **Claim 2**, $\mathsf{Sim}_\mathcal{A}$ can find a solution $(x, E)$ for the given SRSA instance $(N, y)$ by Lemma 2. This is the case (II).

Otherwise, $\psi_t$ is not a correct solution of $\lambda_\alpha^{(t)}$ for any $t \in \{1,2\}$. By **Claim 1**, this means that for some index $t_0 \in \{1,2\}$, $\mathcal{R}$ failed to find a solution of $\lambda_\alpha^{(t_0)}$ despite that $\lambda_\alpha^{(t_0)}$ has a solution in the subgroup $\langle y \rangle$. This is the unfortunate case, namely the case (III). Therefore, $\mathsf{Sim}_\mathcal{A}$ outputs $\perp$, and halts.

It immediately follows from the construction that with the probability 1, $\mathsf{Sim}_\mathcal{A}$ outputs either a witness pair $(\lambda^*, \psi^*)$ or a solution $(x, E)$ for the given SRSA instance $(N, y)$, if $\mathcal{R}$ is ideal as a challenger, namely $\mathsf{Sim}_\mathcal{A}$ can receive a correct reply of the equation query $\lambda^{(t)}$ from $\mathcal{R}$ for each $t \in \{1,2\}$.

Finally, we estimate the success probability of $\mathcal{M}$. The following claim shows that $\mathcal{M}$ breaks SRSA with at least probability $\epsilon_0$.

**Claim 3.** *$\mathcal{M}$ $\epsilon_0$-breaks SRSA.*

Thus the SRSA assumption does not hold. □

**Acknowledgment.** The authors would like to thank anonymous reviewers for their invaluable comments and suggestions on an earlier version of this paper.

## References

1. Abe, M., Groth, J., Ohkubo, M.: Separating Short Structure-Preserving Signatures from Non-interactive Assumptions. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 628–646. Springer, Heidelberg (2011)
2. Abe, M., Haralambiev, K., Ohkubo, M.: Group to Group Commitments Do Not Shrink. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 301–317. Springer, Heidelberg (2012)
3. Agrawal, M., Kayal, N., Saxena, N.: PRIMES Is in P. Annals of Mathematics 160(2), 781–793 (2004)
4. Anokhin, M.: Constructing a Pseudo-Free Family of Finite Computational Groups under the General Integer Factoring Intractability Assumption. Electronic Colloquium on Computational Complexity. report no.114 (2012)
5. Barić, N., Pfitzmann, B.: Collision-Free Accumulators and Fail-Stop Signature Schemes Without Trees. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 480–494. Springer, Heidelberg (1997)
6. Bellare, M., Namprempre, C., Pointcheval, D., Semanko, M.: The Power of RSA Inversion Oracles and the Security of Chaum's RSA-Based Blind Signature Scheme. In: Syverson, P. (ed.) FC 2001. LNCS, vol. 2339, pp. 319–338. Springer, Heidelberg (2002)
7. Bellare, M., Namprempre, C., Pointcheval, D., Semanko, M.: The One-More-RSA-Inversion Problems and the Security of Chaum's Blind Signature Scheme. J. Cryptology 16(3), 185–215 (2008)
8. Boneh, D.: The Decision Diffie-Hellman Problem. In: Buhler, J.P. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 48–63. Springer, Heidelberg (1998)

9. Bresson, E., Monnerat, J., Vergnaud, D.: Separation Results on the "One-More" Computational Problems. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 71–87. Springer, Heidelberg (2008)
10. Catalano, D., Fiore, D., Warinschi, B.: Adaptive Pseudo-free Groups and Applications. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 207–223. Springer, Heidelberg (2011)
11. Cramer, R., Shoup, V.: Signature Schemes Based on the Strong RSA Assumption. J. ACM TISSEC 3(3), 161–185 (2000)
12. Diffie, W., Hellman, M.: New Directions in Cryptography. IEEE Trans. on Information Theory 22(6), 644–654 (1976)
13. 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)
14. Garg, S., Bhaskar, R., Lokam, S.V.: Improved Bounds on Security Reductions for Discrete Log Based Signatures. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 93–107. Springer, Heidelberg (2008)
15. Hasegawa, S., Isobe, S., Shizuya, H., Tashiro, K.: On the Pseudo-Freeness and the CDH Assumption. International Journal of Information Security 8(5), 347–355 (2009)
16. Hirano, T., Tanaka, K.: Variations on Pseudo-Free Groups. Research Reports, Series C: Computer Science, C-239, Tokyo Institute of Technology (2007)
17. Hohenberger, S.: The Cryptographic Impact of Groups with Infeasible Inversion. Master's thesis, EECS Dept., MIT (2003)
18. Jhanwar, M.P., Barua, R.: Sampling from Signed Quadratic Residues: RSA Group Is Pseudofree. In: Roy, B., Sendrier, N. (eds.) INDOCRYPT 2009. LNCS, vol. 5922, pp. 233–247. Springer, Heidelberg (2009)
19. Micciancio, D.: The RSA Group is Pseudo-Free. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 387–403. Springer, Heidelberg (2005)
20. Micciancio, D.: The RSA Group is Pseudo-Free. J. Cryptology 23(2), 169–186 (2010)
21. Paillier, P., Vergnaud, D.: Discrete-Log-Based Signatures May Not Be Equivalent to Discrete Log. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 1–20. Springer, Heidelberg (2005)
22. Pointcheval, D., Stern, J.: Security Arguments for Digital Signatures and Blind Signatures. J. Cryptology. 13(3), 361–396 (2000)
23. Rivest, R.L., Shamir, A., Adleman, L.M.: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM 21(2), 120–126 (1978)
24. Rivest, R.L.: On the Notion of Pseudo-Free Groups. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 505–521. Springer, Heidelberg (2004)
25. Schnorr, C.P.: Efficient Identification and Signatures for Smart Cards. In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 688–689. Springer, Heidelberg (1990)
26. Seurin, Y.: On the Exact Security of Schnorr-Type Signatures in the Random Oracle Model. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 554–571. Springer, Heidelberg (2012)
27. Villar, J.L.: Optimal Reductions of Some Decisional Problems to the Rank Problem. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 80–97. Springer, Heidelberg (2012)

## A Proofs of the Claims in Theorem 1

**Claim 1.** *Assume that* $y^{e_1 e_2} \not\equiv 1 \pmod{N}$*. Then, there exists an index* $t_0 \in \{1,2\}$ *such that the interpreted equations* $\lambda_\alpha^{(t_0)} : x^{e_{t_0}} = y^D$ *has a solution* $\psi_{t_0} \in \langle y \rangle$ *for the equation* $\lambda^{(t_0)} = (e_{t_0}, \boldsymbol{s})$*.*

*Proof.* Assume that $y^{e_1 e_2} \not\equiv 1 \pmod{N}$. Then, we now show that one of $e_1 \in \mathbb{Z}_{P'Q'}^\times$ and $e_2 \in \mathbb{Z}_{P'Q'}^\times$ holds. We assume that $e_1, e_2 \notin \mathbb{Z}_{P'Q'}^\times$. Then, we have $\gcd(e_1, P'Q'), \gcd(e_2, P'Q') \in \{P', Q', P'Q'\}$. If either $\gcd(e_1, P'Q') = P'Q'$ or $\gcd(e_2, P'Q') = P'Q'$ holds, then $y^{e_1 e_2} \equiv 1 \pmod{N}$ holds, since $y \in \mathrm{QR}_N$ and the order $\mathrm{ord}(\mathrm{QR}_N)$ of $\mathrm{QR}_N$ is $P'Q'$. This is a contradiction. Otherwise, we assume without loss of generality that $\gcd(e_1, P'Q') = P'$ holds. Then, there exists an integer $b_1 \in \mathbb{Z}$ such that $e_1 = b_1 P'$. Moreover, we have $e_2 = e_1 + 1 \not\equiv 0 \pmod{P'}$, and hence $P' \nmid e_2$. By the assumption, $\gcd(e_2, P'Q') = Q'$ holds. Then, there exists an integer $b_2 \in \mathbb{Z}$ such that $e_2 = b_2 Q'$. It follows that $e_1 e_2 = b_1 b_2 P'Q'$. This implies that $y^{e_1 e_2} \equiv 1 \pmod{N}$. This is a contradiction. Thus, one of $e_1 \in \mathbb{Z}_{P'Q'}^\times$ and $e_2 \in \mathbb{Z}_{P'Q'}^\times$ holds.

Let $t_0 \in \{1,2\}$ be an index such that $e_{t_0} \in \mathbb{Z}_{P'Q'}^\times$. Since $y \in \mathrm{QR}_N$ and $\mathrm{ord}(\mathrm{QR}_N) = P'Q'$, the interpreted equation $\lambda_\alpha^{(t_0)} : x^{e_{t_0}} = y^D$ has a solution $y^{De_{t_0}^{-1}}$ that belongs to the subgroup $\langle y \rangle$, where $e_{t_0}^{-1}$ denotes the inverse of $e_{t_0}$ in $\mathbb{Z}_{P'Q'}^\times$. □

**Claim 2.** *The integers* $e_1$ *and* $e_2$ *found in (A-2b) satisfy* $e_1 \nmid D$ *and* $e_2 \nmid D$*.*

*Proof.* It follows from $D \neq 0$ and $0 < |D| < |D| + 1 = e_1 < e_2$ that $e_1 \nmid D$ and $e_2 \nmid D$. □

**Claim 3.** $\mathcal{M}$ $\epsilon_0$*-breaks SRSA.*

*Proof.* We denote by $\Pr[\mathsf{Succ}_\mathcal{M}]$ and $\Pr[\mathsf{Succ}_\mathcal{R}]$ the success probability of $\mathcal{M}$ and $\mathcal{R}$, respectively. It is guaranteed that $\Pr[\mathsf{Succ}_\mathcal{R}] \geq \epsilon_0$ by **Correctness of** $\mathsf{Sim}_\mathcal{A}$.

Unity designates the event that the assignment $\alpha : A \longrightarrow \mathbb{Z}_N^\times$ of the game pair $(N, \alpha)$ satisfies that $\alpha(a_i) = 1$ for all $1 \leq i \leq m$.

The event Unity happens. In this case, $\mathsf{Sim}_\mathcal{A}$ outputs the witness pair $(\lambda^*, \psi^*)$ of a nontrivial equation and its corresponding solution, and then $\mathcal{M}$ outputs the final output of $\mathcal{R}$ at (M-4a). Therefore, we have

$$\Pr[\mathsf{Succ}_\mathcal{M} \wedge \mathsf{Unity}] = \Pr[\mathsf{Succ}_\mathcal{R} \wedge \mathsf{Unity}]. \tag{1}$$

The event Unity does not happen. Let SolveEq denote the event that for some index $t_0 \in \{1,2\}$, $\mathcal{R}$ correctly solves the interpreted equation $\lambda_\alpha^{(t_0)} : x^{e_{t_0}} = y^D$ of the queried equation $\lambda^{(t_0)}$ from $\mathsf{Sim}_\mathcal{A}$ during playing the SAPF game.

If the event SolveEq happens, $\mathcal{M}$ outputs the pair $(x, E)$ returned from $\mathsf{Sim}_\mathcal{A}$ at (M-4b). Since $x^E \equiv y \pmod{N}$ holds under the event SolveEq by the correctness of $\mathsf{Sim}_\mathcal{A}$, we have $\Pr[\mathsf{Succ}_\mathcal{M} \mid \neg\mathsf{Unity} \wedge \mathsf{SolveEq}] = 1$. This implies that

$$\begin{aligned} \Pr[\mathsf{Succ}_\mathcal{M} \wedge (\neg\mathsf{Unity} \wedge \mathsf{SolveEq})] &= \Pr[\mathsf{Succ}_\mathcal{M} \mid \neg\mathsf{Unity} \wedge \mathsf{SolveEq}] \Pr[\neg\mathsf{Unity} \wedge \mathsf{SolveEq}] \\ &= \Pr[\neg\mathsf{Unity} \wedge \mathsf{SolveEq}] \\ &\geq \Pr[\mathsf{Succ}_\mathcal{R} \wedge (\neg\mathsf{Unity} \wedge \mathsf{SolveEq})]. \end{aligned} \tag{2}$$

If the event $\mathsf{SolveEq}$ does not happen, $\mathcal{M}$ outputs the final output of $\mathcal{R}$ in (M-4a). Therefore, we have

$$\Pr[\mathsf{Succ}_{\mathcal{M}} \wedge (\neg\mathsf{Unity} \wedge \neg\mathsf{SolveEq})] = \Pr[\mathsf{Succ}_{\mathcal{R}} \wedge (\neg\mathsf{Unity} \wedge \neg\mathsf{SolveEq})]\,. \tag{3}$$

Putting together Eqs. (1)–(3), we have

$$\begin{aligned}
\Pr[\mathsf{Succ}_{\mathcal{M}}] &= \Pr[\mathsf{Succ}_{\mathcal{M}} \wedge \mathsf{Unity}] + \Pr[\mathsf{Succ}_{\mathcal{M}} \wedge \neg\mathsf{Unity}] \\
&= \Pr[\mathsf{Succ}_{\mathcal{M}} \wedge \mathsf{Unity}] + \Pr[\mathsf{Succ}_{\mathcal{M}} \wedge (\neg\mathsf{Unity} \wedge \mathsf{SolveEq})] \\
&\quad + \Pr[\mathsf{Succ}_{\mathcal{M}} \wedge (\neg\mathsf{Unity} \wedge \neg\mathsf{SolveEq})] \\
&\geq \Pr[\mathsf{Succ}_{\mathcal{R}} \wedge \mathsf{Unity}] + \Pr[\mathsf{Succ}_{\mathcal{R}} \wedge (\neg\mathsf{Unity} \wedge \mathsf{SolveEq})] \\
&\quad + \Pr[\mathsf{Succ}_{\mathcal{R}} \wedge (\neg\mathsf{Unity} \wedge \neg\mathsf{SolveEq})] \\
&= \Pr[\mathsf{Succ}_{\mathcal{R}} \wedge \mathsf{Unity}] + \Pr[\mathsf{Succ}_{\mathcal{R}} \wedge \neg\mathsf{Unity}] \\
&= \Pr[\mathsf{Succ}_{\mathcal{R}}] \\
&\geq \epsilon_0.
\end{aligned}$$

Thus, $\mathcal{M}$ $\epsilon_0$-breaks SRSA. □

# Minkowski Sum Based Lattice Construction for Multivariate Simultaneous Coppersmith's Technique and Applications to RSA

Yoshinori Aono

National Institute of Information and Communications Technology
aono@nict.go.jp

**Abstract.** We investigate a lattice construction method for the Coppersmith technique for finding small solutions of a modular equation. We consider its variant for simultaneous equations and propose a method to construct a lattice by combining lattices for solving single equations. As applications, we consider a new RSA cryptanalysis. Our algorithm can factor an RSA modulus from $\ell \geq 2$ pairs of RSA public exponents with the common modulus corresponding to secret exponents smaller than $N^{(9\ell-5)/(12\ell+4)}$, which improves on the previously best known result by Sarkar and Maitra. For partial key exposure situation, we also can factor the modulus if $\beta - \delta/2 + 1/4 < (3\ell-1)(3\ell+1)$, where $\beta$ and $\delta$ are bit-lengths $/\log N$ of the secret exponent and its exposed LSBs, respectively. Due to the spacing limit, some arguments are omitted; see the full-version [1].

## 1 Introduction

Since the RSA cryptosystem [31] was proposed, its security has been intensively investigated. In particular, polynomial-time algorithms for recovering short secret exponents have been studied [34,3]. There are two main strategies for recovering a secret exponent in this situation: The continued fraction algorithm was used in this approach [34,19,17] and the Coppersmith technique based approach [3,5,11]. We consider the latter technique.

Using the Coppersmith technique for finding small roots of a modular equation, Boneh and Durfee [3] proposed an algorithm for recovering a small secret exponent from the corresponding public key pair. Under several acceptable assumptions, the attack is guaranteed to work when the secret exponent is smaller than $N^{0.292}$.

Although the original Coppersmith technique was designed to treat a single modular equation, the method can be extended to multivariate simultaneous equations [8,32,33,15]. Their approaches first construct a single multivariate modular equation whose solutions are also those of the simultaneous equations, and apply the standard Coppersmith technique. This may not be a better strategy from the viewpoint of lattice construction because it does not consider individual equations. May and Ritzenhofen [26] proposed an approach based on the Chinese remainder theorem to solve simultaneous univariate modular equations. In this paper, we study an extension of the Coppersmith technique that directly treats the original simultaneous multivariate equations. We expect that our algorithm will improve several lattice based attacks.

**Related Works on Lattice Construction for the Coppersmith Technique**: For a single modular equation, it has been widely studied. The first work by Coppersmith [9] gave a good lattice construction for any univariate modular equation. Recently, Aono et al. [2] has proven the optimality of this construction. Blömer and May [6] proposed a construction method for bivariate equations, and Jochemsz and May [20] improved this to a method for treating general multivariate equations. Another viewpoint was given by Kunihiro [22], who proposed a method for converting a lattice for an $n$-variable equation $f(x_1, \ldots, x_n) \equiv 0 \pmod{W}$ into a lattice for a new $(n+1)$-variable equation of the form $x_0 f(x_1, \ldots, x_n) + C \equiv 0 \pmod{W}$ where $C$ is a constant. For simultaneous modular equations, May and Ritzenhofen [26] considered a Chinese remainder theorem based approach. They proposed a method for constructing a lattice in the univariate case and gave an application to RSA. Recently, Ritzenhofen [30] improved this approach to multivariate simultaneous equations and proposed a lattice construction method for equations with the common modulus. However, the case for coprime moduli was not solved (see [30, Section 5.4]). We consider this problem.

## 1.1 Contributions of This Work

**Minkowski Sum Based Lattice Construction**: We propose a method to construct a lattice for the Coppersmith technique for simultaneous modular equations. We consider simultaneous equations such as $F_1(x_1, y) \equiv 0 \pmod{W_1}$ and $F_2(x_2, y) \equiv 0 \pmod{W_2}$. Assume that we have lattices spanned by the sets of polynomials $\{g_1^{(1)}, \ldots, g_{c_1}^{(1)}\}$ and $\{g_1^{(2)}, \ldots, g_{c_2}^{(2)}\}$ for the equations, respectively. Then, we propose the *Minkowski sum based lattice construction*, which is a method for generating a lattice basis for solving the simultaneous equations, as a set of polynomials of the form $\sum a_\lambda g_\lambda^{(1)} \cdot g_{\lambda'}^{(2)}$. Our method defines the range of suffixes $(\lambda, \lambda')$ and the coefficients $a_\lambda$ of the combination.

**Cryptanalysis of Multiple RSA Short Secret Exponents**: The above construction method can easily be extended to multivariate and multi-equation situations. By this, we improve the cryptanalysis of RSA with short secret exponents studied in [19,17,32,33]. In this situation, the attacker has $\ell$ pairs of RSA public keys $(e_k, N)$ with the common modulus, which correspond to secret exponents smaller than $N^\beta$ for some $\beta \in (0,1)$. Then, we prove that the RSA modulus is efficiently factored if

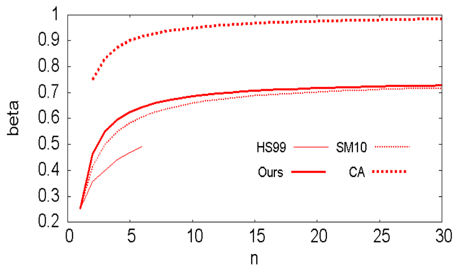
$$\beta < (9\ell - 5)/(12\ell + 4).$$

Here, we assumed that all $e_k$'s are full-sized i.e., they have the same bit sizes. This improves on the previously known best result by Sarkar and Maitra [33], which achieved $\beta < (3\ell - 1)/(4\ell + 4)$. For large $\ell$, both values converge to 3/4.

Noting that Howgrave-Graham and Seifert [19] had given an extended version of Wiener's continued fraction attack and obtain the bound

$$\beta < \frac{(2\ell+1)\cdot 2^\ell - (2\ell+1)\binom{\ell}{\ell/2}}{(2\ell-2)\cdot 2^\ell + (4\ell+2)\binom{\ell}{\ell/2}} \text{ if } 2|\ell \text{ and } \beta < \frac{(2\ell+1)\cdot 2^\ell - 4\ell\binom{\ell-1}{(\ell-1)/2}}{(2\ell-2)\cdot 2^\ell + 8\ell\binom{\ell-1}{(\ell-1)/2}} \text{ if } 2 \not| \ell.$$

However, Hinek and Lam [17] observed that the attack does not recover the secret exponents if the bound exceeds 0.5, i.e., $\ell > 7$. Hence, our result is the best one. These results are compared in Figure 1. `HS99` is the result by Howgrave-Graham and Seifert [19] for $\ell \leq 6$. `SM10` is Sarkar and Maitra [33]. `Ours` shows our result. `CA` indicates the heuristic bound by the counting argument in Section 4.1.



| $\ell$ | 2 | 3 | 4 |
|---|---|---|---|
| HS99: [19] | 0.357 | 0.400 | 0.441 |
| SM10: [33] | 0.416 | 0.500 | 0.550 |
| Our result | 0.464 | 0.550 | 0.596 |
| CA | 0.750 | 0.833 | 0.875 |

**Fig. 1.** Comparison of previous results

**Extension for the Partial Key Exposure Situation**: We then extend the attack to a situation studied in [4,11], in which the attacker has $\ell$ tuples $(e_k, N, \widetilde{d_k})$ where $e_k$ and $N$ are RSA public keys, and each $\widetilde{d_k}$ is $\delta n$ LSBs (least significant bits) of the corresponding secret exponent smaller than $N^\beta$. Then, we prove that the RSA modulus is efficiently factored if

$$\beta - \frac{\delta}{2} + \frac{1}{4} < \frac{3\ell-1}{3\ell+1} \Leftrightarrow \delta > 2\beta + \frac{1}{2} - \frac{2(3\ell-1)}{3\ell+1}.$$

**Computer Experiments**: We perform our computer experiments of the applications for RSA and the partial key exposure situation. Our experiments work well. Interestingly, in the partial key exposure situation, the range of $\beta$ and $\delta$ that we can factor $N$ is slightly larger than that derived by theory.

**Organization of this Paper**: Section 2 gives necessary definitions, lemmas, and an outline of the Coppersmith technique. In Section 3, we consider the Coppersmith technique for the simultaneous equations and propose our Minkowski sum based lattice construction. Sections 4 and 5 give applications to cryptanalysis of RSA Section 6 gives experimental results to verify our lattice construction. In Section 7, we suggest and discuss several open problems.

## 2 Preliminaries

Here we introduce necessary definitions and technical lemmas. For any positive integers $a$ and $b$, let $[a]$ and $[a : b]$ be the set $\{1, \dots, a\}$ and $\{a, a+1, \dots, b-1, b\}$, respectively. For natural numbers $x$, $A$ and $N$, the notation $|x| < A \pmod{N}$ means that $0 \le x < A$ or $N - A < x < N$ holds.

We use $\prec$ to denote the lexicographic order between integer tuples. For example, consider two 2-tuples, $(i_1, i_2)$ and $(i_1', i_2')$, then $(i_1, i_2) \prec (i_1', i_2')$ means that $i_1 < i_1'$ or $[i_1 = i_1'$ and $i_2 < i_2']$ holds. We also use this to order monomials; e.g., $x_1^{i_1}x_2^{i_2} \prec x_1^{i_1'}x_2^{i_2'} \Leftrightarrow (i_1, i_2) \prec (i_1', i_2')$. Here, we neglect the coefficients. These notations are used for general $n$-tuples and $n$-variable monomials. We use $x_1, x_2, \dots, x_{n-1}$ and $y$ to denote the variables, and fix the priority of variables as $y \prec x_{n-1} \prec \cdots \prec x_1$ to order the $n$-variable monomials. For example, consider four variables, $x_1, x_2, x_3, y$, and monomials $3x_2^2x_3$ and $x_1^2x_2^3y$. Then, $3x_2^2x_3 \prec x_1^2x_2^3y$ holds since the corresponding tuples are $(0, 2, 1, 0)$ and $(2, 3, 0, 1)$, respectively. Note that for any integer tuples $T_1, T_2, S_1, S_2$ of the same dimension, $T_1 \prec S_1$ and $T_2 \prec S_2$ implies that $T_1 + T_2 \prec S_1 + S_2$.

With respect to the above order, we can define the maximum element in a polynomial $f(x_1, \dots, x_\ell, y)$. Let $ax_1^{i_1} \cdots x_\ell^{i_\ell}y^j$ be the non-zero maximum monomial in $f$. Then, we call it the *head term* of $f$ and denote it by $\mathrm{HT}(f)$. We also call $a$, $x_1^{i_1} \cdots x_\ell^{i_\ell}y^j$ and $(i_1, \dots, i_\ell, j)$ *head coefficient*, *head monomial* and *head index*, and denote them by $\mathrm{HC}(f)$, $\mathrm{HM}(f)$ and $\mathrm{HI}(f)$, respectively.

Let $A$ and $B$ be finite subsets of $\mathbb{Z}^n$, then their Minkowski sum is defined by $A \boxplus B = \{(a_1 + b_1, \dots, a_n + b_n) : (a_1, \dots, a_n) \in A, (b_1, \dots, b_n) \in B\}$. Note that the sum of three or more sets is defined recursively.

### 2.1 Overview of the Coppersmith Technique

We introduce the Coppersmith technique [9,10] with necessary definitions and lemmas. Our formulation is due to Howgrave-Graham [16] and Aono et al. [2].

Fix a polynomial $F(x, y) \in \mathbb{Z}[x, y]$ and $X, Y, W \in \mathbb{N}$. Then consider the problem of finding all integer solutions of

$$F(x, y) \equiv 0 \pmod{W} \tag{1}$$

satisfying $|x| < X$ and $|y| < Y$. While this is generally not easy, the Coppersmith technique efficiently solves it if $X$ and $Y$ are much smaller than $W$.

The Coppersmith technique first fix an integer $m \ge 2$ and consider a set $L$ of polynomials $g(x, y) \in \mathbb{Z}[x, y]$ satisfying

$$\forall x, y \in \mathbb{Z}\ [F(x, y) \equiv 0 \pmod{W} \Rightarrow g(x, y) \equiv 0 \pmod{W^m}]. \tag{2}$$

Note that $L$ forms a lattice, i.e., it can easily see that $g_1, g_2 \in L \Rightarrow g_1 - g_2 \in L$. Next, find polynomials $g(x, y) \in L$ satisfying

$$\forall x, y \in \mathbb{Z}, |x| < X, |y| < Y\ [g(x, y) \equiv 0 \pmod{W^m} \Rightarrow g(x, y) = 0]. \tag{3}$$

Suppose two algebraically independent polynomials are found, then the original equation (1) can be converted to simultaneous equations over integers, which are easily solved by the resultant technique [14] or the Gröbner basis technique [7]. As we explain below, a polynomial with small coefficients satisfies (3). Our tasks are to construct a polynomial lattice $L$, and to find such polynomials in $L$.

Many algorithms to find small elements in a lattice exist; e.g., the LLL algorithm [23] is widely used. Unfortunately, most of them are designed for treating lattices in Euclidean spaces $\mathbb{R}^n$ w.r.t. the standard Euclidean norms. To use them as a subroutine, a polynomial lattice needs to be converted.

**Converting Polynomials to Vectors**: For a polynomial $g(x,y) = \sum_{i,j} a_{i,j}x^i y^j$ and parameters $X$ and $Y$, define the *vectorization* of the polynomial by

$$\mathcal{V}(g; X, Y) = (a_{0,0}, a_{1,0}X, \ldots, a_{i_w,j_w}X^{i_w}Y^{i_w}).$$

Thus, it maps each term $a_{i,j}x^i y^j$ to each coordinate $a_{i,j}X^iY^j$, respectively. It is a linear mapping with respect to $g$. Note that the sequence of tuples $\{(i_k, j_k)\}_{k=1}^{w}$ is taken so that it covers all non-zero terms in $g(x,y)$. We define the *polynomial norm* w.r.t. the parameters $X, Y$ by $|\mathcal{V}(g; X, Y)|$. W.r.t. this norm, the following lemma holds.

**Lemma 1. (Howgrave-Graham [16], generalized in [20])** *Fix $X, Y, W \in \mathbb{N}$. Let $g(x,y) \in \mathbb{Z}[x,y]$ be a polynomial consisting of $w$ non-zero terms, and $|\mathcal{V}(g; X, Y)| < W/\sqrt{w}$ holds. Then we have*

$$\forall x, y \in \mathbb{Z}, |x| < X, |y| < Y \;\; [g(x,y) \equiv 0 \pmod{W} \Leftrightarrow g(x,y) = 0].$$

Hence, if a polynomial lattice $L$ is given, our task is to find independent polynomials satisfying the above lemma, which is performed by finding short vectors in a Euclidean lattice converted from $L$ using certain parameters. To achieve this, we use a lattice reduction algorithm.

**Euclidean Lattices**: Consider a sequence of linearly independent vectors $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_c\}$ in $\mathbb{Z}^{\tilde{c}}$ where $\tilde{c} \geq c$. Then, the *Euclidean lattice* spanned by them is defined by $L(\mathbf{B}) = \{a_1\mathbf{b}_1 + \cdots + a_c\mathbf{b}_c \; : \; a_k \in \mathbb{Z} \text{ for } k \in [c]\}$. We call $\mathbf{b}_1, \ldots, \mathbf{b}_c$ the basis vectors. Following many papers, we assume that a lattice is represented by its basis vectors.

To find short vectors in a lattice, we use the LLL algorithm [23] which computes an LLL-reduced basis from a given basis. The following theorem bounds the lengths of first vectors in such bases.

**Theorem 1.** *[5] Let $L$ be a Euclidean lattice and $\mathbf{v}_1, \ldots, \mathbf{v}_c$ be its LLL-reduced basis. Then, the following inequality holds for $k \in [c]$.*

$$||\mathbf{v}_k|| \leq 2^{\{(c(c-1)+(k-1)(k-2)\}/4(c-k+1)} |\det(L)|^{1/(c-k+1)} \tag{4}$$

Here, $\det(L)$ is the *lattice determinant* that is defined by using the Gram-Schmidt orthogonal basis $\mathbf{v}_1^*, \ldots, \mathbf{v}_c^*$ as $\det(L) = \prod_{i=1}^{c} ||\mathbf{v}_i^*||$.

**Polynomial Lattices**: Let $\mathbf{G} = \{g_1, \ldots, g_c\}$ be a sequence of linearly independent polynomials in $\mathbb{Z}[x, y]$. Then, the *polynomial lattice* spanned by them is defined by $L(\mathbf{G}) = L(g_1, \ldots, g_c) = \{a_1 g_1 + \cdots + a_c g_c \; : \; a_i \in \mathbb{Z} \text{ for } k \in [c]\}$. We also consider the vectorization of polynomial lattices; i.e., for a basis $\mathbf{G} = \{g_1, \ldots, g_c\}$, consider their vectorization $\mathcal{V}(g_1; X, Y), \ldots, \mathcal{V}(g_c; X, Y)$ w.r.t. parameters $X$ and $Y$. Here, the tuple sequence is assumed to be fixed. Then, define the vectorization of $L(\mathbf{G})$ by the Euclidean lattice spanned by these vectors, and let it be $L(\mathbf{G}; X, Y)$. We use $\det(\mathbf{G}; X, Y)$ to denote the determinant of $L(\mathbf{G}; X, Y)$.

**Outline and a Working Condition for the Coppersmith Technique**: For fixed $X$ and $Y$, suppose we have a polynomial lattice $L(\mathbf{G})$ spanned by $c$ polynomials satisfying (2), and it holds that

$$2^{c/4} \det(\mathbf{G}; X, Y)^{1/c} < N^m / w. \tag{5}$$

Here, $w$ is the length of tuple sequence used at vectorization, which is equal to the Euclidean dimension of $L(\mathbf{G}; X, Y)$, and upper bounds the number of terms of any polynomials in $L(\mathbf{G})$. Then, compute the LLL-reduced basis of $L(\mathbf{G}; X, Y)$. By Theorem 1, the first two vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ in the reduced basis are shorter than $N^m/w$. Hence, the corresponding polynomials, i.e., $h_k(x, y)$ satisfying $\mathbf{v}_k = \mathcal{V}(h_k; X, Y)$ for $k = 1, 2$, also satisfy $|\mathcal{V}(h_k; X, Y)| \leq N^m/w$. Thus, by Lemma 1, these polynomials satisfy

$$\forall x, y \in \mathbb{Z}, |x| < X, |y| < Y \; [F(x, y) \equiv 0 \pmod{W} \Rightarrow h_k(x, y) = 0].$$

Finally, finding small integer solutions of $h_1(x, y) = h_2(x, y) = 0$, we obtain the desired solutions.

As in many previous works, we regard the following simplified condition as a working condition.

$$\det(\mathbf{G}; X, Y)^{1/c} < N^m \tag{6}$$

In many applications, the crucial problem is to construct a lattice $\mathbf{G}$ satisfying (6) for $X$ and $Y$ as large as possible.

The algebraic independence of polynomials $h_k(x, y)$ is necessary to solve the final simultaneous equations over the integers. Unfortunately, this is generally not guaranteed. In this paper, again following previous works, we assume this algebraic independence and justify it by computer experiments.

## 3 Coppersmith Technique for Simultaneous Equations

We consider a variant of the Coppersmith technique for the simultaneous equations, and propose a new method to construct polynomial lattices. For readability, we consider the following three variable simultaneous equations with two equations having the shared variable $y$:

$$F_1(x_1, y) \equiv 0 \pmod{W_1} \text{ and } F_2(x_2, y) \equiv 0 \pmod{W_2} \tag{7}$$

Here, if no variable is shared, the simultaneous equations have no meaning.

Our objective is to find all integer solutions within the range of $|x_1| < X_1$, $|x_2| < X_2$ and $|y| < Y$. Fix the above equations, given ranges, and parameters $c$ and $m$. Then we consider a lattice consisting of three variable polynomials $g_i(x_1, x_2, y)$ such that satisfies

$$\forall x_1, x_2, y \in \mathbb{Z}, \left[ \begin{matrix} F_1(x_1, y) \equiv 0 \pmod{W_1} \\ F_2(x_2, y) \equiv 0 \pmod{W_2} \end{matrix} \Rightarrow g_i(x_1, x_2, y) \equiv 0 \pmod{(W_1W_2)^m} \right]. \tag{8}$$

For a lattice $L(\mathbf{G})$ with basis $\mathbf{G} = \{g_1, \ldots, g_c\}$, compute the LLL-reduced basis of $L(\mathbf{G}; X_1, X_2, Y)$. By the same argument as that in Section 2.1, we can prove the technique works if $\det(\mathbf{G}; X_1, X_2, Y)^{1/c} < (W_1W_2)^m$. The problem is also finding the means of constructing better polynomial lattices.

### 3.1 Minkowski Sum Based Lattice Construction

We give a method for constructing a lattice for the simultaneous equations (7), by combining lattices for solving single equations.

For $k = 1, 2$, let $L(\mathbf{G}_k)$ be a polynomial lattice for solving $F_k(x_k, y) \equiv 0 \pmod{W_k}$ and its basis be $\mathbf{G}_k = \{g_1^{(k)}, \ldots, g_{c_k}^{(k)}\}$. Here we assume that the parameter $m$ is fixed. Then for any $\ell_1 \in [c_1]$ and $\ell_2 \in [c_2]$, the polynomial $g_{\ell_1}^{(1)} \cdot g_{\ell_2}^{(2)}$ satisfies (8). Hence, the set

$$\mathcal{A} = \left\{ \sum_{\ell_1 \in [c_1], \ell_2 \in [c_2]} a_{\ell_1,\ell_2} g_{\ell_1}^{(1)} g_{\ell_2}^{(2)} : a_{\ell_1,\ell_2} \in \mathbb{Z} \right\}$$

forms a polynomial lattice for solving the simultaneous equations. Unfortunately, since the polynomials $\{g_{\ell_1}^{(1)} g_{\ell_2}^{(2)}\}_{\ell_1,\ell_2}$ are not generally independent over the integers, it cannot explicitly obtain the basis of $\mathcal{A}$ and its determinant. Instead, we consider a sublattice of $\mathcal{A}$ and define its basis by using the Minkowski sum of indices.

We can assume that each basis $\mathbf{G}_k$ has a strictly increasing degree order, i.e., $\mathrm{HM}(g_1^{(k)}) \prec \cdots \prec \mathrm{HM}(g_{c_k}^{(k)})$ holds for $k = 1, 2$. If this is not true, an equivalent basis having this property can be efficiently computed by multiplying a unimodular matrix; the computation is performed by a Gaussian elimination-like algorithm, see [2]. Then, for each $k$, consider the set of indices $I_k = \{\mathrm{HI}(g_\ell^{(k)}) : \ell \in [c_k]\} \subset \mathbb{Z}^3$ and let their Minkowski sum be $I_+$. Noting that the elements of $I_1$ and $I_2$ have the form $(i_1, 0, j)$ and $(0, i_2, j)$, respectively. For every $(i_1, i_2, j) \in I_+$, define the polynomial $g_{i_1,i_2,j}^+$ to be

$$g_{i_1,i_2,j}^+ = \sum_{(*)} a_\lambda g_\lambda^{(1)} g_{\lambda'}^{(2)}. \tag{9}$$

Here, the range of sum $(*)$ is over all suffix pairs $(\lambda, \lambda')$ satisfying $\mathrm{HM}(g_\lambda^{(1)} g_{\lambda'}^{(2)}) = x_1^{i_1} x_2^{i_2} y^j$ and the coefficients $a_\lambda$ are defined so that

$$\mathrm{HC}(g_{i_1,i_2,j}^+) = \underset{(*)}{\mathrm{GCD}} \left(\mathrm{HC}(g_\lambda^{(1)} g_{\lambda'}^{(2)})\right), \tag{10}$$

that is, the greatest common divisor of all head coefficients within the range. It is easy to see that the polynomial satisfies (8). We define the polynomial basis by $\mathbf{G}_+ = \{g^+_{(i_1,i_2,j)} : (i_1, i_2, j) \in I_+\}$. Here, it is clear that the basis polynomials are linearly independent since the head monomials are distinct. We call the polynomial lattice $L(\mathbf{G}_+)$ the *Minkowski sum lattice* of $L(\mathbf{G}_1)$ and $L(\mathbf{G}_2)$. Clearly, $L(\mathbf{G}_+) \subset \mathcal{A}$ holds.

The basic strategy of this construction is to minimize the head coefficient of $g^+_{i_1,i_2,j}$ over all the possible integer combinations. It can be expected that the determinant of the combined lattice is reduced. Note that a combination of $a_\lambda$ that attains (10) is generally not unique. Hence, care needs to be taken regarding the determinant if the lattice is not triangular. If the lattice is lower triangular, the determinant, which is computed by $\prod |\mathrm{HC}(g^+_{i_1,i_2,j})| X_1^{i_1} X_2^{i_2} Y^j$, is not changed for any allowed combination of $a_\lambda$.

### 3.2 Minkowski Sum of Lower Triangular Lattices

Suppose the lattices for single equations are lower triangular, that is, there exist sequences of tuples $\{(i_1(\ell), j_1(\ell))\}_{\ell=1}^{c_1}$ and $\{(i_2(\ell), j_2(\ell))\}_{\ell=1}^{c_2}$, the polynomials in bases $\mathbf{G}_k$ can be written as

$$g^{(1)}_\ell = \sum_{\ell'=1}^{\ell} a_{\ell,\ell'} x_1^{i_1(\ell')} y^{j_1(\ell')} \text{ and } g^{(2)}_\ell = \sum_{\ell'=1}^{\ell} b_{\ell,\ell'} x_2^{i_2(\ell')} y^{j_2(\ell')}$$

where $a_{\ell,\ell} \neq 0$ and $b_{\ell,\ell} \neq 0$. In this case, w.r.t. the above sequences of tuples, the Euclidean lattices $L(\mathbf{G}_k; X_k, Y)$ are lower triangular. We can show that the Minkowski sum lattice of them is also lower triangular; for the proof, see the full-version. Note that the situation of three or more lattices, which is considered in our applications, can be proven by induction.

**Theorem 2.** *For $k = 1, 2$, assume that the polynomial lattice basis $\mathbf{G}_k = \{g_1^{(k)}, \dots$*
*$, g_{c_k}^{(k)}\}$ has a strictly increasing degree order, and that they are lower triangular. Then the Minkowski sum lattice $L(\mathbf{G}_+)$ is also lower triangular.*

## 4 Cryptanalysis of RSA with Short Secret Exponents

As an application of our Minkowski sum lattice construction, we analyze the RSA with multiple short secret exponents with a common modulus.

**Notations**: We use the standard notations for the RSA cryptography. That is, $p$ and $q$ are large primes, and let their product be the RSA modulus $N$. $e$ and $d$ are used to denote the public exponent and secret exponent, respectively. The basic relation $ed \equiv 1 \pmod{\varphi(N)}$ holds. Following [3], we assume that $e \approx N$ and $p + q < 3N^{0.5}$.

We consider the situation in which the attacker has $\ell$ pairs of public keys with a common modulus, let them be $(e_1, N), \dots, (e_\ell, N)$, which correspond to small secret exponents satisfying $d_1, \dots, d_\ell < N^\beta$ for some $\beta \in (0, 1)$. For simplicity, we assume that $e_i$ and $e_j$ are coprime to each other for $i \neq j$.

### 4.1 RSA Equation and Its Limit by a Counting Argument

Following the work of Sarkar and Maitra [32,33] (see Boneh and Durfee [3] for deriving single equation), it can prove that the simultaneous equations

$$F_i(x_i, y) = -1 + x_i(y + N) \equiv 0 \pmod{e_i} \text{ for } i = 1, \ldots, \ell \tag{11}$$

have a small solution $(x_1, \ldots, x_\ell, y)$ satisfying

$$|x_k| < N^\beta, \text{ for } k \in [\ell] \text{ and } |y| < 3N^{0.5}, \tag{12}$$

by which we can recover the secret exponents. Hence, our objective here is to find this solution by the Coppersmith technique.

On the other hand, if $\beta$ is not small, the solution within the range is not unique. In this situation, the number of solutions becomes exponential in $\log N$; thus, no polynomial-time algorithm exists. By a counting argument, we set the following heuristic assumption of bounding $\beta$; the detailed argument is given in the full version.

**Heuristic Assumption**: For a natural number $\ell$, assume that

$$\beta < (\ell - 0.5)/\ell. \tag{13}$$

Then, within the range of (12), the equation (11) has only one solution by which we can recover the corresponding secret keys $d_k$.

### 4.2 Our Lattice Construction and Bound

Here we give our polynomial lattice to solve the simultaneous equations (11) and a new security analysis of RSA. As mentioned in Section 3.1, assume that lattices for solving single equation $F_k(x_k, y) = -1 + x_k(y + N) \equiv 0 \pmod{e_k}$ are given. We follow the work of Boneh and Durfee [3], and employ their simple lower triangular lattice that achieves the bound $\beta < 0.25$. While they achieved $\beta < 0.292$ by their improved lattice, we did not use in this paper.

Fix an integer $m \geq 2$ and set

$$g_{i,j}^{(k)}(x_k, y) = x_k^{i-j} F_k(x_k, y) e_k^{m-j} \text{ and } \mathbf{G}_k = \{g_{i,j}^{(k)} : (i,j) \in \mathbb{Z}^2, 0 \leq j \leq i \leq m\} \tag{14}$$

for $k = 1, \ldots, \ell$. It is clear that $g_{i,j}^{(k)}(x_k, y)$ satisfies (2) w.r.t. $F_k(x_k, y) \equiv 0 \pmod{e_k}$ and $m$.

For each $k$, ordering its basis in the lexicographic order in suffixes $(i, j)$, the polynomial sequence has strictly increasing order since $\mathrm{HM}(g_{i,j}^{(k)}) = x_k^i y^j$ and $\mathrm{HI}(g_{i,j}^{(k)}) = (0, \ldots, 0, i, 0, \ldots, 0, j) \in \mathbb{Z}^{\ell+1}$ (the $k$-th and $\ell + 1$-th coordinates are $i$ and $j$, respectively). As shown in [3], the lattice $L(\mathbf{G}_k; X_k, Y)$ is lower triangular. Thus, these bases satisfy the assumption of Theorem 2 and the Minkowski sum lattice $L(\mathbf{G}_+)$ is also lower triangular.

We explicitly give the Minkowski sum lattice. The index set corresponding to $\mathbf{G}_k$ is given by $I_k = \{(0, \dots, 0, i, \dots, 0, j) : (i, j) \in \mathbb{Z},\ 0 \le j \le i \le m\}$ and their Minkowski sum is

$$I_+ = I_1 \boxplus \cdots \boxplus I_\ell = \{(i_1, \dots, i_\ell, j) : 0 \le i_1, \dots, i_\ell \le m \text{ and } 0 \le j \le i_1 + \cdots + i_\ell\}.$$

For each $(i_1, \dots, i_\ell, j) \in I_+$, a polynomial is written as by

$$g_{i_1,\dots,i_\ell,j} = \sum_{j_1,\dots,j_\ell} a_{j_1,\dots,j_\ell} \cdot g^{(1)}_{i_1,j_1} g^{(2)}_{i_2,j_2} \cdots g^{(\ell)}_{i_\ell,j_\ell}.$$

where the sum is over indices such that $\mathrm{HM}(g^{(1)}_{i_1,j_1} g^{(2)}_{i_2,j_2} \cdots g^{(\ell)}_{i_\ell,j_\ell}) = x_1^{i_1} \cdots x_\ell^{i_\ell} y^j$. In this situation, $i_k$ are fixed, and $(j_1, \dots, j_\ell)$ moves over all integer tuples subject to $0 \le j_k \le i_k$ and $j_1 + \cdots + j_\ell = j$. Next we consider the coefficients; again as mentioned in Section 3.1, the coefficients $a_{j_1,\dots,j_\ell}$ are selected so that

$$\mathrm{HC}(g_{i_1,\dots,i_\ell,j}) = \underset{j_1,\dots,j_\ell}{\mathrm{GCD}} \left( \mathrm{HC}(g^{(1)}_{i_1,j_1} g^{(2)}_{i_2,j_2} \cdots g^{(\ell)}_{i_\ell,j_\ell}) \right).$$

Note that $\mathrm{HC}(g^{(1)}_{i_1,j_1} g^{(2)}_{i_2,j_2} \cdots g^{(\ell)}_{i_\ell,j_\ell}) = e_1^{m-j_1} \cdots e_\ell^{m-j_\ell}$. Since $j_k$ can move from zero to $\min(i_k, j)$, the greatest common divisor is $e_1^{m-\min(i_1,j)} \cdots e_\ell^{m-\min(i_\ell,j)}$. Thus, we can take $a_{j_1,\dots,j_\ell}$ so that the head coefficient of $g_{i_1,\dots,i_\ell,j}$ is this value.

Then, we set the Minkowski sum lattice by $\mathbf{G}_+ = \{g_{i_1,\dots,i_\ell,j} : (i_1, \dots, i_\ell, j) \in I_+\}$ and the order is the lexicographic order of suffixes. By Theorem 2 (and its generalization), the converted lattice $L(\mathbf{G}_+; X_1, \dots, X_\ell, Y)$ is lower triangular. The diagonal element corresponding to $(i_1, \dots, i_\ell, j)$ is

$$\mathrm{HC}(g_{i_1,\dots,i_\ell,j}) \times X_1^{i_1} \cdots X_\ell^{i_\ell} Y^j = e_1^{m-\min(i_1,j)} \cdots e_\ell^{m-\min(i_\ell,j)} X_1^{i_1} \cdots X_\ell^{i_\ell} Y^j.$$

Therefore, the determinant is

$$\det(\mathbf{G}_+; X_1, \dots, X_\ell, Y) = \prod_{(i_1,\dots,i_\ell,j) \in I_+} \left[ e_1^{m-\min(i_1,j)} \cdots e_\ell^{m-\min(i_\ell,j)} X_1^{i_1} \cdots X_\ell^{i_\ell} Y^j \right].$$

As with the same argument in Section 2.1, the Coppersmith technique works if $\det(\mathbf{G}_+; X_1, \dots, X_\ell, Y)^{1/|I_+|} < (e_1 \cdots e_\ell)^m$, where $|I_+|$ denotes the number of elements in $I_+$. Using approximations $e_k \approx N$ for $k \in [\ell]$, $X_1 = \cdots = X_\ell = N^\beta$ and $Y \approx N^{0.5}$, the condition can be rewritten as

$$\sum_{(i_1,\dots,i_\ell,j) \in I_+} \left[ 0.5j + (i_1 + \cdots + i_\ell)\beta - \sum_{k=1}^{\ell} \min(i_k, j) \right] < 0. \tag{15}$$

By computing the left-hand side, we derive the condition

$$\left( -\frac{3}{16}\ell^2 + \frac{5}{48}\ell + \left( \frac{\ell^2}{4} + \frac{\ell}{12} \right) \beta \right) m^{\ell+2} + o(m^{\ell+2}) < 0.$$

Thus, when $m$ is sufficiently large, this condition is

$$\beta < (9\ell - 5)/(12\ell + 4). \tag{16}$$

**Heuristic Improvement of Lattice**: Suppose $\beta > 0.5$. We can construct a new lattice by removing polynomials whose indexes satisfy both $j > \max\{i_1, \ldots, i_\ell\}$ and $0.5j + (i_1 + \cdots + i_\ell)\beta - \sum_{k=1}^{\ell} \min(i_k, j) > 0$, which have negative contributions in the sigma (15). It can be shown that the new lattice is also lower triangular. However, we have never derived an explicit formula of the working condition; detailed construction and deriving numerical bounds are given in the full-version.

## 5 Application to Partial Key Exposure Attack on RSA

Assume that the attacker has $\ell$ pairs of RSA public keys $(e_1, N), \ldots, (e_\ell, N)$, and $\delta n$ LSBs of the corresponding $d_k$. Moreover, each $d_k$ is assumed to be smaller than $N^\beta$. Let $M = 2^{\lfloor \delta n \rfloor}$ and the exposed parts be $\widetilde{d_k}$ for $k \in [\ell]$. Then, following the derivation of the single equation for the situation that single $(e, N, \widetilde{d})$ is given [11], we consider the simultaneous equations

$$F_i(x_i, y) = e_i\widetilde{d_i} - 1 + x_i(y + N) \equiv 0 \pmod{e_i M} \text{ for } i = 1, \ldots, \ell. \tag{17}$$

By the counting argument, we can assume that if $\beta - \delta < (\ell - 0.5)/\ell$, then the solution satisfying $|x_1|, \ldots, |x_\ell| < N^\beta$ and $|y| < 3N^{0.5}$ is unique, and it can be used to factor $N$.

The basic lattice construction is the same as in the above section; i.e., we let

$$g_{i,j}^{(k)} = x_k^{i-j}(F_k(x_k, y))^j (e_k M)^{m-j} \text{ and } \mathbf{G}_k = \{g_{i,j}^{(k)} : (i,j) \in \mathbb{Z}^2, 0 \le j \le i \le m\}.$$

Note that only the constant terms and moduli differ between (11) and (17). Thus, $L(\mathbf{G}_k)$ for $k \in [\ell]$ and their Minkowski sum $L(\mathbf{G}_+)$ are also lower triangular. Moreover, the set of indices $I_1, \ldots, I_\ell$ and their Minkowski sum $I_+$ are also the same as in Section 4.2.

For each $(i_1, \ldots, i_\ell, j) \in I_+$, we give the polynomial $g_{i_1,\ldots,i_\ell,j}$. First note that

$$\mathrm{HC}(g_{i_1,j_1}^{(1)} g_{i_2,j_2}^{(2)} \cdots g_{i_\ell,j_\ell}^{(\ell)}) = e_1^{m-j_1} \cdots e_\ell^{m-j_\ell} M^{\ell m - j_1 - \cdots - j_\ell}.$$

Thus, as Section 4.2, each $j_k$ can move from zero to $\min(i_k, j)$, and we can take the coefficients in (9) so that

$$\mathrm{HT}(g_{i_1,\ldots,i_\ell,j}) e_1^{m-\min(i_1,j)} \cdots e_\ell^{m-\min(i_\ell,j)} M^{\ell m - j} x_1^{i_1} \cdots x_\ell^{i_\ell} y^j.$$

Hence, the determinant $\det(\mathbf{G}_+; X_1, \ldots, X_\ell, Y)$ is

$$\prod_{(i_1,\ldots,i_\ell,j) \in I_+} \left[ e_1^{m-\min(i_1,j)} \cdots e_\ell^{m-\min(i_\ell,j)} \times M^{\ell m - j} X_1^{i_1} \cdots X_\ell^{i_\ell} Y^j \right].$$

Params: $\ell$: Number of RSA keys; $n$: RSA bit length; $\beta$: ratio of secret keys to $n$

Step 1: (Generate a sample RSA instance) Randomly choose $\lfloor n/2 \rfloor$-bit pseudoprimes $p$ and $q$, and let $N = pq$. Randomly choose $\ell$ $\lfloor \beta n \rfloor$-bit odd integers $d_1, \ldots, d_\ell$ such that $\mathrm{GCD}(d_k, (p-1)(q-1)) = 1$ for all $k \in [\ell]$. Compute the corresponding $e_k$ by $d_k^{-1} \pmod{(p-1)(q-1)}$. For each $k \in [\ell]$, define the RSA polynomial $f_k(x_k, y) = -1 + x_k(N + y)$ and let the solutions $\bar{x_k} = (1 - e_k d_k)/(p-1)(q-1)$ and $\bar{y} = 1 - p - q$.

Step 2: Set the bounds $X_k = \lfloor N^\beta \rfloor$ and $Y = \lfloor 3N^{0.5} \rfloor$. Construct the polynomial lattice $L(\mathbf{G})$ in Section 4.2, and compute the Euclidean lattice $L(\mathbf{G}_+; X_1, \ldots, X_\ell, Y)$. Then, apply the LLL algorithm to $L(\mathbf{G}_+; X_1, \ldots, X_\ell, Y)$.

Step 3: From the reduced basis, pick the first $\ell + 1$ vectors $\mathbf{v}_1, \ldots, \mathbf{v}_{\ell+1}$. Then, compute the corresponding polynomials $h_k(x_1, \ldots, x_k, y)$, i.e., take polynomials so that $\mathbf{v}_k = \mathcal{V}(h_k; X_1, \ldots, X_k, Y)$ for $k \in [\ell + 1]$.

Step 4: First check $h_i(\bar{x_1}, \ldots, \bar{x_\ell}, \bar{y}) = 0$ for all $k \in [\ell+1]$. If it is not true, reject the instance. After the polynomials pass the first check, compute the resultant of polynomials modulo prime to check the algebraic independence. If the instance passes two checks, then we regard the experiment as successful.

**Fig. 2.** Procedure of our computer experiments

Plugging the approximations $e_k \approx N$, $X_k = N^\beta$, $Y \approx N^{0.5}$ and $M \approx N^\delta$, the working condition is

$$\sum_{(i_1,\ldots,i_\ell,j)\in I} \left[ (0.5 - \delta)j + (i_1 + \cdots + i_\ell)\beta - \sum_{k=1}^{\ell} \min(i_k, j) \right] < 0. \tag{18}$$

Calculating the left-hand side, when $m$ becomes large, the condition is

$$\beta - \frac{\delta}{2} + \frac{1}{4} < \frac{3\ell - 1}{3\ell + 1}. \tag{19}$$

## 6 Computer Experiments of our RSA Cryptanalysis

**Experimental Environment:** The experiments were conducted on a workstation with 16GB of RAM and two Intel Xeon X5675@3.07GHz. We wrote our experimental code in the C++ language using the following libraries. To compute the LLL reduced basis, we used Shoup's NTL library [28] version 5.5.2 compiled with the GMP library [13] version 5.0.4. The polynomial computation was performed using the GiNaC library [12] version 1.6.2. We compiled our source code using `g++` version 4.5.4 with the `-O3` option. We also used `Maple 15` to compute the resultant in $\mathbb{Z}_p$ in the final step of the experiments. We performed our experiments on the Windows 7 platform and ran our program in a single thread.

### 6.1 Experiments for Short RSA Secret Exponents

Figure 2 shows the procedure of our computer experiments. In Step 1, "pseudoprime" means an odd integer that passes the Euler-Jacobi primality testing

**Table 1.** Theoretical $\beta$ bound and lattice dimension for small $\ell$ and several $m$

| $\ell = 2$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $m$ | 2 | 3 | 4 | 5 | 6 | 7 | 10 | limit |
| $\beta$ | 0.386 | 0.405 | 0.416 | 0.424 | 0.430 | 0.434 | 0.442 | 0.464 |
| dim | 27 | 64 | 125 | 216 | 343 | 512 | 1331 | - |

| $\ell = 3$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $m$ | 2 | 3 | 4 | 5 | 6 | 7 | 10 | limit |
| $\beta$ | 0.464 | 0.486 | 0.500 | 0.508 | 0.514 | 0.519 | 0.527 | 0.550 |
| dim | 108 | 352 | 875 | 1836 | 3430 | 5888 | 21296 | - |

**Table 2.** Experimental results for short secret exponents

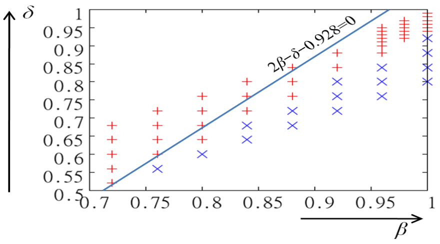
| $\ell$ | $m$ | $n$ | $\beta_{\text{thm}}$ | dim | $\beta_{\text{exp}}$ | LLL-time |
|---|---|---|---|---|---|---|
| 2 | 2 | 512 | 0.386 | 27 | 0.386 | 3.2 sec |
| | | 1024 | | | 0.386 | 10.55 sec. |
| 2 | 3 | 512 | 0.405 | 64 | 0.406 | 5 min. 33 sec |
| | | 1024 | | | 0.406 | 30 min. 44 sec. |
| 2 | 4 | 512 | 0.414 | 125 | 0.416 | 3 hrs. 50 min. |
| | | 1024 | | | 0.414 | 20hrs. 26min. |
| 3 | 2 | 512 | 0.464 | 108 | 0.464 | 41 min. 25 sec. |
| | | 1024 | | | 0.464 | 3 hrs. 17 min. |

for bases 2, 3, 5 and 7. In Step 2, we use the command `LLL_XD(L,0.99,0,0,1)`. In the second-half of Step 4, we first generate a random $0.5n$ bit prime number $P$. Then, we erase the variable $x_1$ by computing $r_k = \mathrm{Res}_{x_1}(h_1, h_k) \bmod P$ for $k = 2, \ldots, \ell+1$, and next we compute $\mathrm{Res}_{x_2}(r_2, r_k) \bmod P$ for $k = 3, \ldots, \ell+1$ modulo $P$, and repeat this process. Finally, we obtain a univariate polynomial $R(y)$ and check $R(\bar{y}) \equiv 0 \pmod{P}$. We repeat this check for three distinct pseudoprime numbers via `Maple 15`.

**Parameters and Results:** Note first that if $m$ and $\ell$ are fixed, condition (15) is written in a linear function w.r.t. $\beta$, and the maximum $\beta$ satisfying the inequality is easily computed. This $\beta$ is a theoretical bound when $N$ becomes large along with neglecting several factors as described in Section 2.1. For each $m$ and $\ell$ we compute the maximum $\beta$ and the dimension of lattice. They are shown in Table 1. The column "limit" indicates the right-hand side of (16).

We carried out our experiments to search for the practical bound of $\beta$ for several choices of $\ell$, $m$ and $n$. We executed our procedure for each $\beta$ at intervals of 0.002. Table 2 shows the experimental results. The column "$\beta_{\text{exp}}$" indicates the experimental bound of $\beta$ for parameters $(l, m, n)$; that is, the instance passed the final test at that $\beta$ and failed at $\beta + 0.002$. The columns "$\beta_{\text{thm}}$" and "dim" are the theoretical bound of $\beta$ and the lattice dimension, respectively; which are the same as shown in Table 1. The running time of the LLL algorithm for processing $L(\mathbf{G}_+)$ is given in the column "LLL-time."

We note that for $\ell = 3$ and $m = 2$, the second half of Step 4 is not finished due to computational time. More precisely, `Maple` computed two bivariate polynomials, $r_1(x_3, y)$ and $r_2(x_3, y)$ from $h_1, \ldots, h_4$. It took over 120 hours to compute $\mathrm{Res}_{x_3}(r_1, r_2)$, and we stopped the computation. However, we can observe that $h_1, \ldots, h_4$ are algebraically independent since they are reduced to the bivariate polynomials, and can expect that the final resultant will be computed if more



| $\beta$ | $\delta$ | LLL-time | result |
|---|---|---|---|
| 1.00 | 0.96 | 18 hrs. 59 min. | + (passed) |
| 1.00 | 0.94 | 16 hrs. 13 min. | + (passed) |
| 1.00 | 0.92 | 15 hrs. 46 min. | × (fault) |
| 0.96 | 0.90 | 15 hrs. 42 min. | + (passed) |
| 0.96 | 0.88 | 16 hrs. 9 min. | + (passed) |
| 0.96 | 0.84 | 14 hrs. 46 min. | × (fault) |

**Fig. 3.** Experimental results for partial key exposure situation

time is permitted [1]. Hence, we regard the experiment as a success. From the observation, we conclude our method works well.

### 6.2 Experiments for Partial Key Exposure Situation

We conducted our experiments on the partial key exposure situation. The experimental procedure is similar to in Figure 2. Different points are the definition of $F_k(x_k, y)$, and that $M = 2^{\lfloor \delta n \rfloor}$ and $\widetilde{d_k} = d \bmod M$ are added in Step 1.

We fixed the parameters $\ell = 3$ and $m = 2$ since it could be taken $\beta$ close to one. Unfortunately, for this $\ell$, only the lattice constructed with $m = 2$ can be reduced in reasonable time. The lattice dimension is 108 as in the above subsection. For several choices of $\beta$ and $\delta$, we generated 1024-bit RSA sample instances and tested them.

Figure 3 shows the result. In the figure, the horizontal and vertical axes are $\beta$ and $\delta$, respectively. Each mark represents one experiment $(\beta, \delta)$ at the point. The marks "+" and "×" mean that the instance passed and was a fault, respectively. The left table in Figure 3 indicates the running time of the LLL algorithm and experimental results for several $\beta$ and $\delta$ close to $\beta = 1$. Again, note that the final resultant computation was not finished and regard that the experiment is successful if `Maple` computes two bivariate polynomials.

## 7 Discussion and Open Problems

**Minkowski Sum Lattice Construction:** Although our lattice construction works well, it is not optimal. That is, in Section 3.1, $L(\mathbf{G}_+)$ is a sublattice of $\mathcal{A}$ that spanned by all possible combination of polynomials. Providing a method to extract the lattice basis of $\mathcal{A}$, and deriving the condition so that $L(\mathbf{G}_+)$ and $\mathcal{A}$ are equivalent are open problems.

**Cryptanalysis of RSA with Small Secret Exponents:** Both our bound (16) and that by Sarkar and Maitra converge to $N^{0.75}$ when $\ell$ becomes large, whereas the limit by the counting argument is $N$. Filling this gap is an interesting problem. We expect that our heuristic improvement shown in Appendix D in the full-verstion achieves this goal, though this is not proven.

[1] An ACISP reviewer proposed to use the Gröbner basis instead, and use more polynomials since the LLL algorithm usually finds more small vectors than required.

**Cryptanalysis of RSA in Other Situations:** The proposed Minkowski sum based lattice construction can be applied to other situations of cryptanalysis of RSA including revealed MSBs [11], RSA-CRT [20], Takagi's RSA [21], small $e$ [4,5,24], unbalanced $p$ and $q$ situation [25], and special settings of $e$ [27]. For more information, see [29, Chap. 10].

**Acknowledgments.** I would like to thank Noboru Kunihiro and Shiho Moriai for helpful discussion and comments on an earlier version of this paper. I also would like to thank the anonymous reviewers for insightful comments.

## References

1. Aono, Y.: Minkowski sum based lattice construction for multivariate simultaneous Coppersmith's technique and applications to RSA, Cryptology ePrint Archive, 2012/675 (2012)
2. Aono, Y., Agrawal, M., Satoh, T., Watanabe, O.: On the Optimality of Lattices for the Coppersmith Technique. In: Susilo, W., Mu, Y., Seberry, J. (eds.) ACISP 2012. LNCS, vol. 7372, pp. 376–389. Springer, Heidelberg (2012); The full-version is available online at Cryptology ePrint Archive, 2012/134
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., Frankel, Y.: An attack on RSA given a small fraction of the private key bits. In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 25–34. Springer, Heidelberg (1998)
5. 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)
6. 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)
7. Cox, D., Little, J., O'Shea, D.: Ideals, varieties, and algorithms: An introduction to computational algebraic geometry and commutative algebra. Springer, New York (2007)
8. Coron, J.-S., Naccache, D., Tibouchi, M.: Fault Attacks Against EMV Signatures. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 208–220. Springer, Heidelberg (2010)
9. 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)
10. 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)
11. 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)
12. GiNaC is Not a CAS, `http://www.ginac.de/`
13. The GNU MP Bignum Library, `http://gmplib.org/`
14. Healy, A.D.: Resultants, Resolvents and the Computation of Galois Groups, `http://www.alexhealy.net/papers/math250a.pdf`

15. Herrmann, M.: Improved cryptanalysis of the multi-prime $\Phi$-hiding assumption. In: Nitaj, A., Pointcheval, D. (eds.) AFRICACRYPT 2011. LNCS, vol. 6737, pp. 92–99. Springer, Heidelberg (2011)
16. Howgrave-Graham, N.: Finding small roots of univariate modular equations revisited. In: Darnell, M. (ed.) Cryptography and Coding 1997. LNCS, vol. 1355, pp. 131–142. Springer, Heidelberg (1997)
17. Hinek, M.J., Lam, C.C.Y.: Common modulus attacks on small private exponent RSA and some fast variants (in practice). Journal of Mathematical Cryptology 4(1), 58–93 (2010)
18. Herrmann, M., May, A.: Attacking power generators using unravelled linearization: When do we output too much? In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 487–504. Springer, Heidelberg (2009)
19. Howgrave-Graham, N., Seifert, J.-P.: Extending Wiener's attack in the presence of many decrypting exponents. In: Baumgart, R. (ed.) CQRE 1999. LNCS, vol. 1740, pp. 153–166. Springer, Heidelberg (1999)
20. 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)
21. Kunihiro, N., Kurosawa, K.: Deterministic polynomial time equivalence between factoring and key-recovery attack on Takagi's RSA. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 412–425. Springer, Heidelberg (2007)
22. Kunihiro, N.: Solving generalized small inverse problems. In: Steinfeld, R., Hawkes, P. (eds.) ACISP 2010. LNCS, vol. 6168, pp. 248–263. Springer, Heidelberg (2010)
23. Lenstra, A.K., Lenstra Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261, 515–534 (1982)
24. Luo, P., Zhou, H.-J., Wang, D.-S., Dai, Y.-Q.: Cryptanalysis of RSA for a special case with $d > e$. Science in China Series F: Information Sciences 52(4), 609–616 (2009)
25. May, A.: Cryptanalysis of unbalanced RSA with small CRT-exponent. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 242–256. Springer, Heidelberg (2002)
26. May, A., Ritzenhofen, M.: Solving systems of modular equations in one variable: How many RSA-encrypted messages does Eve need to know? In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 37–46. Springer, Heidelberg (2008)
27. Maitra, S., Sarkar, S.: A New Class of Weak Encryption Exponents in RSA. In: Chowdhury, D.R., Rijmen, V., Das, A. (eds.) INDOCRYPT 2008. LNCS, vol. 5365, pp. 337–349. Springer, Heidelberg (2008)
28. Shoup, V.: NTL: A Library for doing Number Theory, `http://www.shoup.net/ntl/index.html`
29. Nguyen, P.Q., Vallée, B.: The LLL algorithm: Survey and applications. Springer, Berlin (2009)
30. Ritzenhofen, M.: On efficiently calculating small solutions of systems of polynomial equations: lattice-based methods and applications to cryptography, Ph.D. thesis, Ruhr University Bochum, `http://www-brs.ub.ruhr-uni-bochum.de/netahtml/HSS/Diss/RitzenhofenMaike/diss.pdf`
31. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptsystems. Communications of the ACM 21(2), 120–128 (1978)
32. Sarkar, S., Maitra, S.: Cryptanalysis of RSA with two decryption exponents. Information Processing Letter 110, 178–181 (2010)
33. Sarkar, S., Maitra, S.: Cryptanalysis of RSA with more than one decryption exponent. Information Processing Letter 110, 336–340 (2010)
34. Wiener, M.J.: Cryptanalysis of short RSA secret exponents. IEEE Transactions on Information Theory 36(3), 553–558 (1990)

# Adaptive Precision Floating Point LLL*

Thomas Plantard, Willy Susilo, and Zhenfei Zhang

Centre for Computer and Information Security Research
School of Computer Science & Software Engineering (SCSSE)
University of Wollongong, Australia
{thomaspl,wsusilo,zz920}@uow.edu.au

**Abstract.** The LLL algorithm is one of the most studied lattice basis reduction algorithms in the literature. Among all of its variants, the floating point version, also known as $\mathrm{L}^2$, is the most popular one, due to its efficiency and its practicality. In its classic setting, the floating point precision is a fixed value, determined by the dimension of the input basis at the initiation of the algorithm. We observe that a fixed precision overkills the problem, since one does not require a huge precision to handle the process at the beginning of the reduction. In this paper, we propose an adaptive way to handle the precision, where the precision is adaptive during the procedure. Although this optimization does not change the worst-case complexity, it reduces the average-case complexity by a constant factor. In practice, we observe an average 20% acceleration in our implementation.

**Keywords:** Lattice Theory, Lattice Reduction, LLL, floating point arithmetic.

## 1 Introduction

A lattice $\mathcal{L}$ is a discrete subgroup of $\mathbb{R}^n$. It is usually represented by a set of integer linear combinations of vectors $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_d)$, $\mathbf{b}_i \in \mathbb{R}^n$, $d \leq n$. $\mathbf{B}$ is a basis of $\mathcal{L}$, if $\mathbf{b}_i$-s are linearly independent, and $d$ is known as the dimension of the $\mathcal{L}$. For a given lattice $\mathcal{L}$, there exists an infinite number of bases for $d \geq 2$. Given a "bad" basis (i.e., a basis with large coefficients), to find a short vector of the lattice (a vector with small coefficients), or a "good" basis (i.e., a basis with small coefficients and the vectors are almost orthogonal), is known as the *lattice reduction.*

The LLL algorithm, named after its inventors, Lenstra, Lenstra and Lovász [11], is a polynomial time lattice reduction algorithm. For a basis $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_d)$, the LLL algorithm is proven to terminate within $O(d^6\beta^3)$ time, where $\beta$ is the maximum bit length of all the Euclidean norm of input vectors. This algorithm is of great importance in cryptanalysis, since finding vectors with exponential approximation (in $d$) to the shortest non-zero vector will break some of the cryptosystems, such as the Coppersmith's method [6,5] against RSA cryptosystem

* This work is supported by ARC Future Fellowship FT0991397.

[21]. For this reason, one of the working direction is to increase the time efficiency of the algorithm.

One of the greatest achievements towards this end was due to [16,18], which incorporates the floating point arithmetics. It is the first algorithm that achieves quadratic complexity in terms of $\beta$, hence it was named L$^2$. In practice, there exist two main versions, a standard version (referred to as "L2") that delivers the proven complexity and a heuristic version (referred to as "FP") that adopts some heuristics to boost the reduction.

L$^2$ uses floating points instead of integers, where the precision is set to $O(d)$ to deliver error free arithmetics. The precision was determined at the beginning of the procedure. However, we notice that this setting indeed is an overkill. LLL deals with vectors progressively. During the process, when $k < d$ vectors are involved, it only requires $O(k)$ bit precision to deliver correct reduction. In fact, the only time that the algorithm requires $O(d)$ precision is when all the vectors are involved. This inspires us to propose the adaptive precision floating point LLL algorithm.

*Our Contribution.* We present an adaptive precision floating point LLL algorithm, the *ap-fplll*. We consider both the proven version, L2, and the most efficient version, FP. We test our *ap-fplll* with random lattices. In practice, we are always faster than the standard version of L$^2$. When the dimension and/or determinant are sufficiently large, we are also faster than the fastest implementation of L$^2$. In general, we accelerate the reduction by 20%.

*Roadmap.* In the next section, we will discuss the background knowledge and terminology required throughout the paper and briefly recall the L$^2$ algorithm. In Section 3, we show our adaptive precision floating point LLL algorithm and analyze its performance. In Section 4, we show the implementation result of our proposed algorithm. Finally, Section 5 concludes the paper.

## 2 Background

It this section, we briefly review the related area. We refer readers to [12,14] for a more complex account of lattice theory, and [4,19] for the LLL algorithm.

### 2.1 Lattice Basics

The lattice theory, also known as the geometry of numbers, was introduced by Minkowski in 1896 [15].

**Definition 1 (Lattice).** *A lattice $\mathcal{L}$ is a discrete sub-group of $\mathbb{R}^n$, or equivalently the set of all the integral combinations of $d \leq n$ linearly independent vectors over $\mathbb{R}$.*

$$\mathcal{L} = \mathbb{Z}\mathbf{b}_1 + \mathbb{Z}\mathbf{b}_2 + \cdots + \mathbb{Z}\mathbf{b}_d, \mathbf{b}_i \in \mathbb{R}^n$$

$\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_d)$ *is called a basis of $\mathcal{L}$ and $d$ is the dimension of $\mathcal{L}$, denoted as* $\dim(\mathcal{L})$. *$\mathcal{L}$ is a full rank lattice if $d$ equals to $n$.*

**Definition 2 (Successive Minima).** *Let $\mathcal{L}$ be an integer lattice of dimension $d$. Let $i$ be a positive integer. The $i$-th successive minima with respect to $\mathcal{L}$, denoted by $\lambda_i$, is the smallest real number, such that there exist $i$ non-zero linearly independent vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_i \in \mathcal{L}$ with*

$$\|\mathbf{v}_1\|, \|\mathbf{v}_2\|, \ldots, \|\mathbf{v}_i\| \leq \lambda_i.$$

The norm of $i$-th minima of a random lattice is estimated by

$$\lambda_i(\mathcal{L}) \sim \sqrt{\frac{d}{2\pi e}} \det(\mathcal{L})^{\frac{1}{d}}. \tag{1}$$

**Definition 3 (Gram-Schmidt Orthogonalization).** *Let $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_d)$ be a basis of $\mathcal{L}$. The Gram-Schmidt Orthogonalization (GSO) of $\mathbf{B}$ is the following basis $\mathbf{B}^* = (\mathbf{b}_1^*, \ldots, \mathbf{b}_d^*)$*

$$\begin{aligned}
\mathbf{b}_1^* &= \mathbf{b}_1, \\
\mathbf{b}_i^* &= \mathbf{b}_i - \sum_{j=1}^{i-1} \mu_{i,j} \mathbf{b}_j^*, \qquad (2 \leq i \leq d), \\
\mu_{i,j} &= \frac{\mathbf{b}_i \cdot \mathbf{b}_j^*}{\mathbf{b}_j^* \cdot \mathbf{b}_j^*}.
\end{aligned}$$

**Definition 4 (Gram determinants).** *Let $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_d)$ be a basis of $\mathcal{L}$. Let $\mathbf{B}^* = (\mathbf{b}_1^*, \ldots, \mathbf{b}_d^*)$ be the corresponding GSO. The Gram determinants of $\mathbf{B}$, noted $\{\Delta_1^*, \ldots, \Delta_d^*\}$ is defined as*

$$\Delta_i^* = \det(\mathbf{b}_1^*, \ldots, \mathbf{b}_i^*).$$

The loop invariant is defined as the product of all Gram determinants as: $D = \prod_{i=1}^{d-1} \Delta_i^*$. For any basis, $D$ is upper-bounded by $2^{\beta d(d-1)}$ [4].

## 2.2 The LLL Algorithm

The LLL algorithm was proposed by Lenstra, Lenstra and Lovasz [11] in 1982. We briefly sketch the algorithm as in Algorithm 1 and 2. LLL produces a $(\delta, 0.5)$-reduced basis for a given basis (see definitions below).

**Definition 5 ($\eta$-size reduced[18]).** *Let $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_d)$ be a basis of $\mathcal{L}$. $\mathbf{B}$ is $\eta$-size reduced, if $|\mu_{i,j}| \leq \eta$ for $1 \leq j < i \leq d$. $\eta \geq 0.5$ is the reduction parameter.*

**Definition 6 ($(\delta, \eta)$-reduced basis[18]).** *Let $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_d)$ be a basis of $\mathcal{L}$. $\mathbf{B}$ is $(\delta, \eta)$-reduced, if the basis is $\eta$-size reduced and it satisfies Lovász condition as follows: $\delta\|\mathbf{b}_{i-1}^*\|^2 \leq \|\mathbf{b}_i^* + \mu_{i,i-1}^2 \mathbf{b}_{i-1}^*\|^2$ for $2 \leq i \leq d$. $\frac{1}{4} < \delta \leq 1$ and $\frac{1}{2} \leq \eta < \sqrt{\delta}$ are two reduction parameters.*

For a lattice $\mathcal{L}$ with dimension $d$, and a basis $\mathbf{B}$, where the Euclidean norm of all spanning vectors in $\mathbf{B}$ is $\leq 2^{\beta}$, the worst-case time complexity is polynomial $O(d^6\beta^3)$.

This complexity comes from the following. Firstly, there exists $O(d^2\beta)$ loop iterations. It has been shown that the loop invariant $D$ is not changed except during the swap procedure, while during the swap, $D$ is decreased by a factor of $\delta$. Hence, the total number of swaps is upper bounded by the absolute value of $\frac{\beta d(d-1)}{\log_2 \delta}$. Therefore there are maximum $O(d^2\beta)$ loop iterations. As a result, the total number of size reduction is $O(d^2\beta)$. Secondly, for each size reduction, there are $O(d^2)$ arithmetic operations. Finally, each operation involves integer multiplications with a cost of $\mathcal{M}(d\beta)$ due to rational arithmetics.[1] Hence, the original LLL algorithm terminates in polynomial time $O(d^6\beta^3)$.

**Algorithm 1.** Size Reduction

```
Input: $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_d)$, its GSO and an index $k$.
Output: A new basis $\mathbf{B}$, where $\mathbf{b}_k$ is size reduced, and the updated GSO.
  for $i = (k-1) \to 1$ do
    $\mathbf{b}_k \leftarrow \mathbf{b}_k - \lceil \mu_{k,i} \rfloor \cdot \mathbf{b}_i$.
    Update GSO
  end for
  return $\mathbf{B}$.
```

Note that for the bases of a random lattices that are using in our analysis and implementation, the number of loop iterations is $O(d\beta)$ instead of $O(d^2\beta)$ (see Remark 3, [16]). So the complexity will be reduced by $O(d)$.

## 2.3 Floating Point LLL Algorithm

The most costly part in an LLL procedure is the size reduction. When one performs a size reduction, the GSO needs to be regularly updated. During the update, the classic LLL needs to operate on integers with length of $O(d\beta)$. As a result, the multiplication of those integers incurs a cost of $O(\mathcal{M}(d\beta))$.

**The First Floating Point LLL Algorithm.** In [22] and [23], Schnorr showed that using floating points instead of integers for LLL can reduce the cost of multiplications from $\mathcal{M}(d\beta)$ to $\mathcal{M}(d+\beta)$. To the best of our knowledge, this is the first time where floating points make a significant difference in the LLL algorithm. However, it is obseved that the hidden constant in the bit complexity remains huge.

[1] $\mathcal{M}(d)$ represents the cost of multiplication between two $O(d)$ length integers. In this paper, we follow the LLL algorithm by using $\mathcal{M}(d)$ to be $O(d^2)$ assuming a naive integer multiplication. One can replace it with $O(d^{1+\varepsilon})$ to obtain the exact bit complexity in practice.

**Algorithm 2.** LLL

```
Input: $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_d)$ and a reduction parameter $\eta$
Output: A $(\delta, 0.5)$-reduced basis $\mathbf{B}$.
  Compute GSO.
  $k \leftarrow 2$.
  while $k \leq d$ do
    size reduce $(\mathbf{B}, k)$;
    if $\delta \|\mathbf{b}^*_{k-1}\|^2 \leq \|\mathbf{b}_k\|^2 + \mu^2_{k,k-1} \|\mathbf{b}^*_{k-1}\|^2$ (Lovász condition) then
      $k \leftarrow k + 1$
    else
      swap $\mathbf{b}_k$ and $\mathbf{b}_{k-1}$;
      $k \leftarrow \max(k-1, 2)$;
      Update GSO;
    end if
  end while
  return $\mathbf{B}$.
```

**The L$^2$ Algorithm.** To further improve the efficiency, the L$^2$ algorithm was proposed by Nguyen and Stehlé [16] in 2005. It is the first variant whose worst-case time complexity is quadratic with respect to $\beta$. It uses a worst-case time complexity of $O(d^5\beta^2 + d^6\beta)$ to produce a $(\delta, \eta)$-reduced basis for $\frac{1}{4} < \delta \leq 1$ and $\frac{1}{2} < \eta < \sqrt{\delta}$.

The L$^2$ algorithm incorporates the *lazy reduction* as follows: firstly, the size reduction consists of many floating point reductions (*fp*-reduction). Then, within each *fp*-reduction, one works on floating point whose precision is $O(d)$. The factor within $O(\cdot)$ is influenced by the reduction parameters. A default setting in the *fplll* is approximately $1.6d$. As a consequence, the multiplication cost is reduced to $O(\mathcal{M}(d))$. However, the trade-off is that, each *fp*-reduction may be incomplete, and one is required to perform $O(1 + \frac{\beta}{d})$ *fp*-reductions to ensure that the vectors is size reduced.

**L$^2$ in Practice.** In practice, the *fplll* library [20] and MAGMA [3] are two well known implementations. Within both implementations, there exist two main versions, "L2" and "FP" (it is known as "LM_WRAPPER" in the *fplll*). The L2 is described as above. It is the proved version of L$^2$. Meanwhile, in practice, one can further improve the average performance with some heuristics. To the best of our knowledge, the most efficient implementation of L$^2$ is FP. As far as the floating point is concerned, FP is L2 plus some early reductions.

In FP, the basis is early reduced as follows: the algorithm will choose several fixed precisions subject to the following conditions:

– The arithmetics are fast with those precisions, for instance, 53 for C double precision.
– Reductions with those precisions are likely to produce a correct basis, for instance, $d$ rather than $1.6d$ (see Remark 4, [17]).

Reductions with above precisions are cheaper, while they produce somewhat reduced bases. So the algorithm will try all early reductions with different fixed precisions, and finally perform an L2 to ensure the quality of reduction. We note that those early reductions do not change the overall complexity, since in theory the last L2 is still the most costly one. Nevertheless, in practice, the early reductions are very effective to accelerate the whole procedure.

## 3 Our Adaptive Precision Floating Point LLL Algorithm

### 3.1 The Algorithm

The LLL algorithm uses a stepping method. For a basis $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_d)$, it starts with the first 2 vectors, and then adds 1 vector into the procedure during each step. We notice that, one does not require a floating point precision of $O(d)$ to reduce in the first $d-1$ steps. In fact, for any $k$ vectors, one only requires $O(k)$ precisions. Hence, a possible improvement is to adaptively select the precision according to the number of vectors that are involved. This leads to the adaptive precision floating point LLL algorithm (*ap-fplll*) as shown in Algorithm 3.

**Algorithm 3.** Adaptive precision floating point LLL algorithm

```
Input: $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_d)$, reduction parameters $(\delta, \eta)$ and a starting index $\gamma$.
Output: An $(\delta, \eta)$-reduced basis $\mathbf{B}$.
  $k \leftarrow 2$, $k_{max} \leftarrow \gamma$.
  SetPrecision($\gamma$) and Compute GSO accordingly.
  while $k \leq d$ do
    Size reduce $(\mathbf{B}, k)$;
    if $\delta \|\mathbf{b}^*_{k-1}\|^2 \leq \|\mathbf{b}_k + \mu_{k,k-1} \|\mathbf{b}^*_{k-1}\|^2$ (Lovász condition) then
      $k \leftarrow k + 1$;
      if $k > k_{max}$ then
        $k_{max} \leftarrow k$;
        if $k_{max} > \gamma$ then
          SetPrecision($k_{max}$) and Compute GSO accordingly.
        end if
      end if
    else
      Swap $\mathbf{b}_k$ and $\mathbf{b}_{k-1}$;
      $k \leftarrow \max(k-1, 2)$;
      Update GSO;
    end if
  end while
  return $\mathbf{B}$.
```

Algorithm 3 describes the L2 version of our *ap-fplll* algorithm. $k$ indicates the current vector the algorithm is working on, while $k_{max}$ indicates the maximum number of the vectors that are involved. When $k_{max}$ changes, one is required to reset the precision. To obtain the FP version of the algorithm, one conducts early reductions as in *fplll* when $k_{max}$ increases.

We also introduce an index parameter $\gamma$ due to implementation issue. For some of the library, there exists a minimum precision for floating point. If the required precision is smaller than this bound, the algorithm will automatically use the bound. In this case, the precision is not $O(d)$, rather it is a fixed value subject to the system. Hence, using an adaptive precision will not reduce the cost of multiplication, rather it will repetitively recompute the GSO. We set $\gamma$ such that when more than $\gamma$ vectors are involved, the algorithm will need to use a precision subject to the dimension.

*Remark 1.* In our algorithm, we follow the L2 by setting the precision to be the exact value that is required, i.e., $1.6d$, to deliver a fair comparison. Nevertheless, it is worth pointing out that the *mpfr* library [1] that the *fplll* depends on operates a floating number as a linked list of blocks of 32 bits (or 64 bits), therefore, it is possible that increasing precisions with respect to the size of the block (the actual size may be smaller than 32 or 64 due to the overhead of storing a floating number) may derive a better performance, since in this case, the GSO will be updated less often.

## 3.2 Worst-Case Complexity

In this part, we prove that our algorithm uses the same worst-case complexity with $L^2$.

The reduction part of $L^2$ algorithm can be seen as our algorithm with a fixed precision of $O(d)$. Therefore, during the reduction part we can never be more costly than $L^2$. However, our algorithm needs to recompute the GSO for each step, where the GSO is updated partially in $L^2$. On the worst-case, we can be more costly than $L^2$ by the cost of computing the GSO.

For each step, the cost of computing GSO is $O(d^2k^2\beta)$. This brings an overall cost of $O(d^5\beta)$, hence it will not affect the worst-case complexity of $O(d^6\beta + d^5\beta^2)$. As a result, the *ap-fplll* uses a same worst-case complexity with $L^2$.

## 3.3 Average Behaviors

We construct the random lattices as in [9]. There exist bases of those lattices that are of the following form:

$$\mathbf{B} = \begin{pmatrix} X_1 & 0 & 0 & \dots & 0 \\ X_2 & 1 & 0 & \dots & 0 \\ X_3 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ X_d & 0 & 0 & \dots & 1 \end{pmatrix},$$

where $X_1$ is a large prime with $\beta$ bits. $X_i$-s ($i \neq 1$) are chosen uniformly between 0 and $p$.

These bases are somewhat standard to analyze lattice reductions. They are believed to leak the least information for the corresponding lattice, since it can be obtained by any lattice basis within polynomial time. They are adopted in [8,17] where the LLL behavior is widely analyzed. Further, when setting $\beta \sim 10d$, these bases are also used for the shortest vector problem (SVP) challanges in [2].

We analyze the average case complexity of our algorithm with the above bases. Since the lattice is a random one, then its minimas $\lambda_i$ follow Equation 1.

$$\mathbf{B}_k = \begin{pmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,k} & 0 & \dots & 0 \\ x_{2,1} & x_{2,2} & \dots & x_{2,k} & 0 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots & \vdots & \dots & \vdots \\ x_{k,1} & x_{k,2} & \dots & x_{k,k} & 0 & \dots & 0 \\ X_{k+1} & 0 & \dots & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots & \vdots & \dots & \vdots \\ X_d & 0 & \dots & 0 & 0 & \dots & 1 \end{pmatrix}$$

For the $k$-th step ($k > 2$), the basis is shown as above, where $\|\mathbf{b}_i\| \lesssim 2^{\frac{k-1}{2}} 2^{\frac{\beta}{k-1}}$ for $i < k$ and $\|\mathbf{b}_k\| \lesssim 2^{\frac{k-2}{2}} 2^{\frac{\beta}{k-2}}$. Hence, the loop invariant for the current step $D_k$ is then bounded by

$$\prod_{i=1}^{k} \|\mathbf{b}_i\|^{2(k-i+1)} = 2^{k(k-1)^2-1} 2^{2\beta k + \frac{\beta}{(k-1)(k-2)}}.$$

When the $k$-th step terminates, $\mathbf{b}_i$ will be reduced to $2^{\frac{k}{2}} 2^{\frac{\beta}{k}}$ for $i \le k$. Hence, one obtains $O(\beta)$ loop iterations on average cases. We note that this observation is quite natural, since there are $O(d\beta)$ loop iterations in total, hence, on average there are $O(\beta)$ loop iterations for each $k$.

Let $l$ be the precision to be used in the algorithms. Then for each loop iteration, one needs to perform $O(1 + \frac{\beta}{l})$ floating point reductions, each at a cost of $O(d^2\mathcal{M}(l))$. Since $l = O(k)$, so it will cost $O(d^2\beta \sum_{i=\gamma}^{d}(1 + \frac{\beta}{i})\mathcal{M}(i)))$ that is $\frac{1}{6}c_1 d^5\beta + \frac{1}{2}c_2 d^4\beta^2$ for some constants $c_1$ and $c_2$, if we assume $\mathcal{M}(d) = O(d^2)$.

For comparison, we also show the complexity of L$^2$: $O(d^2\beta \sum_{i=1}^{d}(1+\frac{\beta}{d})\mathcal{M}(d))$ which is $c_1 d^5\beta + c_2 d^4\beta^2$ for the same constants.

It is straightforward to see that our algorithm uses the same bit complexity with L$^2$. Further, our algorithm wins on both terms. However, the factor $\frac{1}{6}$ on the first term does not make a difference, which is due to the following. Firstly, in this case, $\beta < d$ which indicates that for each lazy reduction, only requires $O(1)$ *fp*-reductions, while our advantage is in fact a faster *fp*-reduction. Hence, our advantage diminishes. Secondly, the cost of recompute the GSO is also $O(d^5\beta)$ on worst cases as well.

Nevertheless, when $\beta > d$, we anticipate a lot of reductions. In this case, we should be able to accelerate the reduction by a factor between 0 and 50% for L2 (due to the fact that in practice $\mathcal{M}(d) \le O(d^{1+\varepsilon})$).

As for FP, in practice, it is possible that the early reduction already produces a good basis. It happens a lot when the dimension is small and $\frac{\beta}{d}$ is small. In this case, the adaptive precision will not boost the reduction, since our advantage works on the final procedure, while in the final procedure, the basis is already in a good shape. Nevertheless, when one increases the dimension and/or the $\frac{\beta}{d}$, the adaptive precision will still accelerate the reduction.

## 4 Implementation

In this section we show the implementation results of our algorithm. The tests were conducted with *fplll* library version 4.0 on Xeon E5640 CPUs @ 2.66GHz. The memory was always sufficient since the algorithm only requires a polynomial space. We used the random lattice basis as shown in the last section. We set the dimension to 64 and increase it by 32 each time. For each dimension, we set $\beta = 10d$, $20d$, $\cdots$, and generated the bases accordingly. For each dimension/determinant, we tested 10 different bases where the random numbers are generated from different seeds $0 \sim 9$ using the pseudo-random generator of the NTL library [24].

We set the index $\gamma = 40$ so that the required precision is strictly greater than 53. In deed, one can change $\gamma$ to improve *ap-fplll*. However, to show a more fair comparison, we use a same value for all the tests. The reduction parameter pair is set to $(0.99, 0.51)$ as the default value of *fplll*. This results in a very strongly reduced basis which is in general most useful for cryptanalysis.

We show the implementation results as follows. As one can see from Table 1, one can merely observe any difference between two algorithms at the beginning of the tests, although *ap-fplll*-L2 is slightly faster than *fplll*-L2. We believe the reason is that the cost of recompute the GSO is more or less the same of the advantage of using smaller precisions. However, when the dimension grows, the reductions influence the total complexity more importantly compared with the GSO computation, and as a result, the *ap-fplll* starts to be a lot faster. Figure 1 illustrated the winning percentage of *ap-fplll*-L2 versus *fplll*-L2. When $d = 64$, we accelerate the reduction by 10%, since it is closer to the starting index $\gamma = 40$. When $d \geq 96$, the influence of $\gamma$ diminishes, and we start to see the phenomenon where the dimension and/or determinant grow, the advantage increases as well. When the dimension and the determinant are sufficiently large, we can expect an advantage up to 40%. Overall, our algorithm is always faster in all cases, and we anticipate a boost of over 20% in general.

The results for the FP version are shown in Table 2. The results are not as stable as L2 due to the early reductions. As we anticipated, with small determinant/dimension, i.e., $\beta = 10d$, our algorithm does not accelerate the reduction. The early reduction technique works extremely efficient in those cases. Nevertheless, the disadvantage is still acceptable considering that even in dimension 448, the disadvantage is less than several of minutes.

Meanwhile, for the other cases, when the dimension grows, we start to observe advantages. Further, the advantage rises with the increase of dimension

**Table 1.** Test Results: *ap-fplll*-L2 vs *fplll*-L2

| | $\beta$ | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 10$d$ | | 20$d$ | | 30$d$ | | 40$d$ | |
| $d$ | *ap-fplll*-L2 | *fplll*-L2 | *ap-fplll*-L2 | *fplll*-L2 | *ap-fplll*-L2 | *fplll*-L2 | *ap-fplll*-L2 | *fplll*-L2 |
| 64 | 5.298 | 5.742 | 11.015 | 12.319 | 17.719 | 20.069 | 23.941 | 27.212 |
| 96 | 29.488 | 30.607 | 64.643 | 69.336 | 104.553 | 113.513 | 144.9 | 158.222 |
| 128 | 95.445 | 99.326 | 221.722 | 237.136 | 368.633 | 409.15 | 516.012 | 577.822 |
| 160 | 234.241 | 253.711 | 575.6 | 636.703 | 971.459 | 1149.22 | 1417.34 | 1718.84 |
| 192 | 470.986 | 522.537 | 1241.9 | 1416.41 | 2278.79 | 2876.82 | 3248.4 | 4184.2 |
| 224 | 838.059 | 1003.08 | 2385.81 | 2944.07 | 4386.86 | 6062.06 | 6604.18 | 9238.31 |
| 256 | 1349 | 1702.95 | 4221.74 | 5452.48 | 8235.8 | 11684.3 | 11942.6 | 17102.9 |
| 288 | 2033.29 | 2561.16 | 6979.11 | 9080.97 | 13481.2 | 19231.7 | | |
| 320 | 2772.15 | 3702.22 | 11007.3 | 15048.3 | | | | |
| 352 | 3686.8 | 5181.06 | | | | | | |
| 384 | 4772.02 | 7175.78 | | | | | | |
| 416 | 6087.31 | 9476.52 | | | | | | |
| 448 | 7641.25 | 12563.9 | | | | | | |

**Table 2.** Test Results: *ap-fplll*-FP vs *fplll*-FP

| | | $\beta$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $10d$ | | $20d$ | | $30d$ | | $40d$ | | $50d$ | |
| | | *ap-fplll*-FP | *fplll*-FP | *ap-fplll*-FP | *fplll*-FP | *ap-fplll*-FP | *fplll*-FP | *ap-fplll*-FP | *fplll*-FP | *ap-fplll*-FP | *fplll*-FP |
| | 64 | 1.251 | 0.968 | 2.19 | 1.941 | 3.302 | 3.1 | 4.484 | 4.275 | 5.707 | 5.536 |
| | 96 | 6.195 | 4.833 | 12.206 | 11.488 | 19.283 | 18.919 | 26.769 | 27.424 | 34.075 | 35.028 |
| | 128 | 19.426 | 16.934 | 40.471 | 40.672 | 66.898 | 70.512 | 93.99 | 102.083 | 123.823 | 134.324 |
| | 160 | 48.422 | 42.133 | 109.613 | 113.167 | 185.293 | 201.169 | 268.095 | 298.629 | 357.412 | 418.395 |
| | 192 | 108.432 | 95.222 | 264.037 | 266.864 | 476.426 | 533.024 | 699.073 | 777.995 | 982.345 | 1076.97 |
| | 224 | 220.045 | 201.6 | 543.712 | 626.538 | 1160.49 | 1719.1 | 1922.35 | 2589.31 | 2537.65 | 3705.78 |
| $d$ | 256 | 564.462 | 605.925 | 1862.08 | 2829.1 | 3979.16 | 5243.41 | 6013.42 | 8036.89 | 8105.54 | 10185.1 |
| | 288 | 1127.6 | 1175.76 | 4557.49 | 5576.77 | 8451.64 | 10784.1 | 12722.3 | 16226 | 16787.2 | 21073.3 |
| | 320 | 1879.03 | 1868.19 | 7826.52 | 9341.18 | 15175.8 | 18896 | | | | |
| | 352 | 2800.87 | 2896.21 | 12445.3 | 14664.5 | | | | | | |
| | 384 | 4136.41 | 4123.88 | | | | | | | | |
| | 416 | 6223.48 | 6150.35 | | | | | | | | |
| | 448 | 8821.49 | 8641.27 | | | | | | | | |

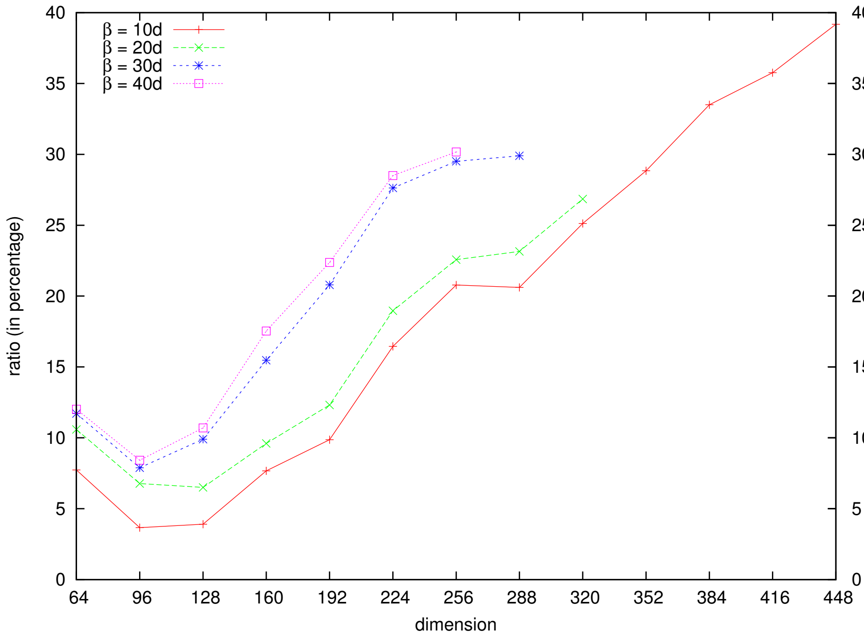


**Fig. 1.** Test Results: winning percentage of *ap-fplll* vs *fplll* using L2

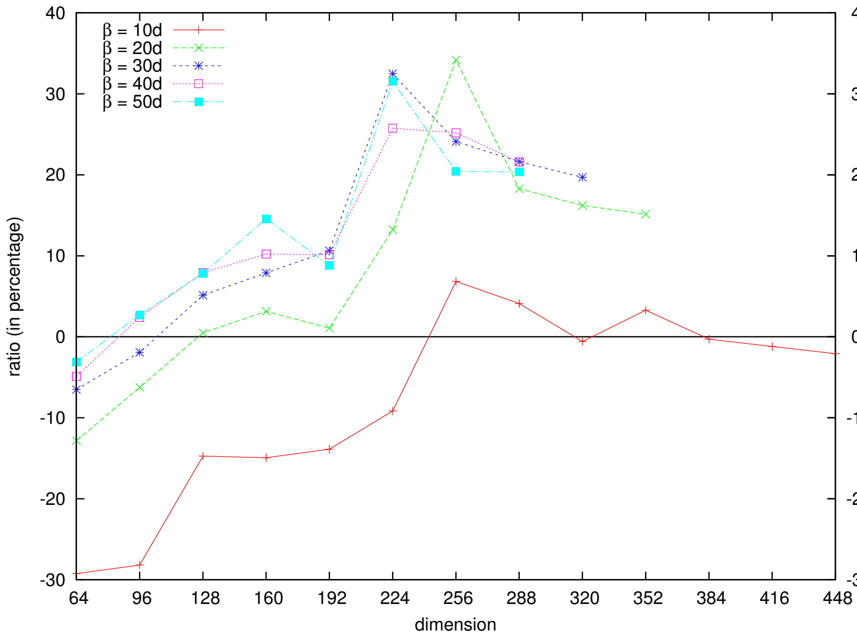


**Fig. 2.** Test Results: winning percentage of *ap-fplll* vs *fplll* using FP

and determinant, just like L2. However, we notice the advantage is not stable. This is because the early reduction affects differently for different dimensions. Overall, as shown in Figure 2, with dimension grows, we accelerate the reduction by approximately 20% for $\beta \geq 20d$. In cryptanalysis, one usually needs to deal with lattice with massive dimension and/or determinants, for instance, the Coppersmith-Shamir's technique [7] against an NTRU cryptosystem [10], so it is still helpful to use adaptive precisions when $d \geq 128$ and $\beta \geq 20d$.

## 5 Conclusion

In this paper, we presented an adaptive floating point precision LLL algorithm. The cost of reduction relies heavily on the precision of the floating point, and the precision used in $L^2$ in fact overkills the problem. Therefore, we presented an approach that adaptively handles the precision. In practice, our algorithm is always faster than the proved version of $L^2$. It also out-performs the fastest implementation so far in most cases.

## References

1. mpfr library, `http://www.mpfr.org/`
2. SVP CHALLENGE, `http://www.latticechallenge.org/svp-challenge/index.php`
3. Bosma, W., Cannon, J., Playoust, C.: The Magma algebra system. I. the user language. J. Symbolic Comput. 24(3-4), 235–265 (1997)
4. Bremner, M.: Lattice Basis Reduction: An Introduction to the LLL Algorithm and Its Applications. Pure and Applied Mathematics. CRC Press Inc. (2012)
5. Coppersmith, D.: Finding a small root of a bivariate integer equation; factoring with high bits known. In: Maurer (ed.) [13], pp. 178–189
6. Coppersmith, D.: Finding a small root of a univariate modular equation. In: Maurer (ed.) [13], pp. 155–165
7. Coppersmith, D., Shamir, A.: Lattice attacks on NTRU. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 52–61. Springer, Heidelberg (1997)
8. Gama, N., Nguyen, P.Q.: Predicting lattice reduction. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 31–51. Springer, Heidelberg (2008)
9. Goldstein, D., Mayer, A.: On the equidistribution of hecke points. Forum Mathematicum 15, 165–189 (2006)
10. 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)
11. Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261, 513–534 (1982)
12. Lovász, L.: An Algorithmic Theory of Numbers, Graphs and Convexity. CBMS-NSF Regional Conference Series in Applied Mathematics, vol. 50. SIAM Publications (1986)
13. Maurer, U.M. (ed.): EUROCRYPT 1996. LNCS, vol. 1070. Springer, Heidelberg (1996)

14. Micciancio, D., Goldwasser, S.: Complexity of Lattice Problems, A Cryptographic Perspective. Kluwer Academic Publishers (2002)
15. Minkowski, H.: Geometrie der Zahlen. B. G. Teubner, Leipzig (1896)
16. Nguen, P.Q., Stehlé, D.: Floating-point LLL revisited. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 215–233. Springer, Heidelberg (2005)
17. Nguyen, 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)
18. Nguyen, P.Q., Stehlé, D.: An lll algorithm with quadratic complexity. SIAM J. Comput. 39(3), 874–903 (2009)
19. Nguyen, P.Q., Valle, B.: The LLL Algorithm: Survey and Applications, 1st edn. Springer Publishing Company, Incorporated (2009)
20. Pujol, X., Stehlé, D., Cade, D.: fplll library, `http://perso.ens-lyon.fr/xavier.pujol/fplll/`
21. Rivest, R.L., Shamir, A., Adleman, L.M.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM 21(2), 120–126 (1978)
22. Schnorr, C.-P.: A more efficient algorithm for lattice basis reduction. J. Algorithms 9(1), 47–62 (1988)
23. Schnorr, C.-P., Euchner, M.: Lattice basis reduction: Improved practical algorithms and solving subset sum problems. Math. Program. 66, 181–199 (1994)
24. Shoup, V.: NTL - A Library for Doing Number Theory, `http://www.shoup.net/ntl/index.html`

# Better Lattice Constructions for Solving Multivariate Linear Equations Modulo Unknown Divisors

Atsushi Takayasu and Noboru Kunihiro

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

**Abstract.** At CaLC 2001, Howgrave-Graham proposed the polynomial time algorithm for solving univariate linear equations modulo an unknown divisor of a known composite integer, the so-called partially approximate common divisor problem. So far, two forms of multivariate generalizations of the problem have been considered in the context of cryptanalysis. The first is simultaneous modular univariate linear equations, whose polynomial time algorithm was proposed at ANTS 2012 by Cohn and Heninger. The second is modular multivariate linear equations, whose polynomial time algorithm was proposed at Asiacrypt 2008 by Herrmann and May. Both algorithms cover Howgrave-Graham's algorithm for univariate cases. On the other hand, both multivariate problems also become identical to Howgrave-Graham's problem in the asymptotic cases of root bounds. However, former algorithms do not cover Howgrave-Graham's algorithm in such cases. In this paper, we introduce the strategy for natural algorithm constructions that take into account the sizes of the root bounds. We work out the selection of polynomials in constructing lattices. Our algorithms are superior to all known attacks that solve the multivariate equations and can generalize to the case of arbitrary number of variables. Our algorithms achieve better cryptanalytic bounds for some applications that relate to RSA cryptosystems.

**Keywords:** Lattices, Coppersmith's method, small roots, implicit factorization, Multi-Prime $\Phi$-Hiding Assumption, Fault Attacks, Digital Signatures, RSA.

## 1 Introduction

In 1996, Coppersmith introduced lattice-based methods for solving modular equations that had small solutions [5]. So far, several results of RSA vulnerabilities have been reported using the method [2,6,20,21]. Howgrave-Graham's reformulation [16] describes the method as the reduction of solving modular equations to finding small roots of polynomials over the integers. It is crucial to select polynomials carefully in constructing lattices to maximize the solvable root bounds. From a theoretical point of view, it is interesting to consider the strategy for selecting polynomials to achieve optimal root bounds. In the context of cryptanalysis, there are a number of cases in which modular equations need to be solved. Therefore, also from a practical point of view, it is important to construct the efficient algorithm to solve modular equations.

At CaLC 2001, Howgrave-Graham [17] proposed the polynomial time algorithm to solve the modular univariate linear equations,

$$a + x = 0 \pmod{p}.$$

Although we do not know the moduli $p$, we know its multiple $N$ where $p \geq N^{\beta}$. The size of root is bounded, $|x| < N^{\gamma}$. Howgrave-Graham's algorithm can solve the problem provided that $\gamma < \beta^2$. The running time is polynomial in $\log N$. Howgrave-Graham called the problem the partially approximate common divisor problem. We call the problem $(1, 1)$-ME throughout the paper since there is one equation which has one variable.

So far, two forms of multivariate modular linear equations have been considered in the context of cryptanalysis. Both equations can be considered as multivariate generalizations of $(1, 1)$-ME. The first generalization is the simultaneous modular univariate linear equations,

$$\begin{cases} a_1 + x_1 = 0 \pmod{p}, \\ \qquad \vdots \\ a_n + x_n = 0 \pmod{p}. \end{cases}$$

We call this problem $(n, 1)$-ME since there are $n$ equations each of which has one variable. At Eurocrypt 2010, van Dijk, Gentry, Halevi and Vaikuntanathan introduced fully homomorphic encryption over the integers [12]. The security lies in the difficulty of solving $(n, 1)$-ME. So far, the variants of the scheme have been proposed [8,9,11]. Sarkar and Maitra used $(n, 1)$-ME in solving relaxed factorization problem [24], the so-called implicit factorization that was introduced by May and Ritzenhofen at PKC 2009 [22]. At CHES 2012, Fouque et al. introduced fault attacks on CRT-RSA signatures by solving $(n, 1)$-ME [13]. The attack is applicable to any padding functions even for RSA-PSS [1]. The second generalization is the modular multivariate linear equations,

$$a_0 + a_1x_1 + \cdots + a_nx_n = 0 \pmod{p}.$$

We call this problem $(1, n)$-ME since there is one equation which has $n$ variables. The problem $(1, n)$-ME has been used in factoring with known bits to arbitrary blocks [15]. At Crypto 2010, Kiltz, O'Neil and Smith introduced the Multi-Prime $\Phi$-Hiding Assumption [18]. After that, some cryptanalysises of the Multi-Prime $\Phi$-Hiding Assumption have been considered via $(1, n)$-ME [14,18,23,25]. At CHES 2009, Coron et al. introduced fault attacks on CRT-RSA signatures by solving $(1, n)$-ME [7]. The attacks are applicable to randomized padding functions. Coron, Naccache and Tibouchi applied the attacks to EMV signatures at CT-RSA 2010 [10].

In both problems, as with $(1,1)$-ME, we do not know the moduli $p$, and we know its multiple $N$ where $p \geq N^{\beta}$. The sizes of all roots are bounded, $|x_1| < N^{\gamma_1}, \dots, |x_n| < N^{\gamma_n}$. Needless to say, both $(n,1)$-ME and $(1,n)$-ME are the same as $(1,1)$-ME with $n = 1$. Additionally, in the asymptotic cases $\gamma_2, \gamma_3, \dots, \gamma_n \to \beta$ for $(n,1)$-ME and $\gamma_2, \gamma_3, \dots, \gamma_n \to 0$ for $(1,n)$-ME, the equations become identical to $(1,1)$-ME.

Several algorithms to solve $(n,1)$-ME have been proposed [3,4,9,24]. As a natural generalization of Howgrave-Graham's algorithm, Cohn and Heninger proposed the algorithm to solve $(n,1)$-ME at ANTS 2012 [4]. Cohn and Heninger's algorithm can solve $(n,1)$-ME provided that $(\gamma_1 + \cdots + \gamma_n)/n < \beta^{(n+1)/n}$. The running time of the algorithm is polynomial in $\log N$ and exponential in $n$. With $n = 1$, this condition covers Howgrave-Graham's condition. This algorithm was constructed considering different root bounds. Since Cohn and Heninger bounded the sizes of unknowns by individual value $X_1, \dots, X_n$ in $(n,1)$-ME and the same value $X$ in other problems in [4]. However, in the asymptotic case $\gamma_2, \gamma_3, \dots, \gamma_n \to \beta$, Cohn and Heninger's condition is worse than Howgrave-Graham's condition. The $(n,1)$-ME algorithm can be improved to overcome the issue.

At Asiacrypt 2008, Herrmann and May proposed the algorithm to solve $(1,n)$-ME [15]. Herrmann and May's algorithm can solve $(1,n)$-ME provided that $\sum_{i=1}^{n} \gamma_i < 1 - (n+1)(1-\beta) + n(1-\beta)^{(n+1)/n}$. The running time of the algorithm is polynomial in $\log N$ and exponential in $n$. As in the case with $(n,1)$-ME, with $n = 1$, this condition covers Howgrave-Graham's condition. However, in the asymptotic case $\gamma_2, \gamma_3, \dots, \gamma_n \to 0$, Herrmann and May's condition is worse than Howgrave-Graham's condition. The $(1,n)$-ME algorithm can be improved to overcome the issue. Herrmann and May pointed out the issue, and claimed that their $(1,n)$-ME algorithm with imbalanced root bounds had the room to be improved.

The improved algorithm to solve $(1,2)$-ME with imbalanced root bounds was proposed by Sarkar [23]. Sarkar's algorithm is superior to Herrmann and May's algorithm with extremely large or small root bounds. Sarkar's algorithm covers Howgrave-Graham's algorithm in the asymptotic case, though no proof is written in [23]. The algorithm can achieve the best cryptanalytic bounds for the Multi-Prime $\Phi$-Hiding Assumption. However, the property of the algorithm was not analyzed in detail. The condition of Sarkar's algorithm has two parameters that are not optimized. So, we do not know the relation between Sarkar's algorithm and Herrmann and May's algorithm; how large improvement is achieved and when Sarkar's algorithm is superior to Herrmann and May's algorithm. Furthermore, the algorithm was not extended to general $n$-variate cases.

The forms of $(n,1)$-ME and $(1,n)$-ME seem to be very simple. However, there are no optimal algorithms to solve even these simple multivariate equations when each root bound becomes large or small. To construct more efficient algorithms are theoretically important and strongly motivated. In the case that we analyze the more complicated forms of modular equations, the strategy to construct $(n,1)$-ME and $(1,n)$-ME algorithms has to be useful. From a practical point of view, such improved algorithms have to provide better cryptanalytic bounds for some applications.

**Our Contributions.** In previous works, the sizes of root bounds were not considered in the phase of selecting polynomials. Therefore, although the former algorithms [4,15] may be optimal when each root bound is the same or similar, they are not optimal in general cases. This is why the former algorithms cover Howgrave-Graham's algorithm with $n = 1$, but become worse in the asymptotic cases.

As a solution, we take into account the sizes of root bounds $\gamma_1, \dots, \gamma_n$ in the phase of selecting polynomials. We carefully decide if each polynomial is *helpful* or not. The diagonals of helpful polynomials in the basis matrix is smaller than modulo, this criterion relates to the value $\gamma_1, \dots, \gamma_n$. Helpful polynomials contribute to the condition for modular equations to be solved, and vice versa. So, we select as many helpful polynomials as possible.

Based on this simple strategy, we propose the algorithms to solve $(n, 1)$-ME and $(1, n)$-ME. Our algorithms work under better conditions than all known algorithms since previous works selected less helpful polynomials in constructing lattices. In addition, our algorithms cover Howgrave-Graham's algorithm in the asymptotic cases. Though our algorithms are heuristic, they work well in our numerical experiments.

Cohn and Heninger's algorithm can solve $(n, 1)$-ME provided that

$$\frac{\gamma_1 + \cdots + \gamma_n}{n} < \beta^{(n+1)/n}.$$

Our algorithm can solve $(n, 1)$-ME provided that

$$\sqrt[n]{\gamma_1 \cdots \gamma_n} < \beta^{(n+1)/n}.$$

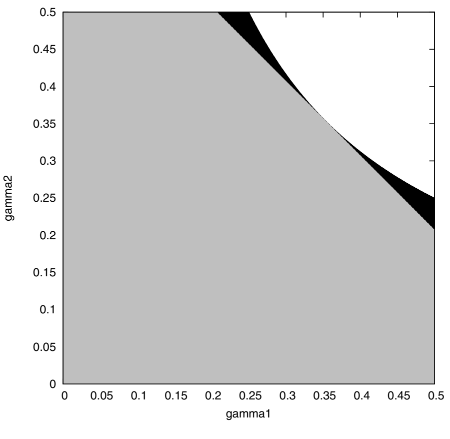


**Fig. 1.** The comparison of the bounds for $(2, 1)$-ME with $\beta = 0.5$ to be solved

Our algorithm becomes identical to Cohn and Heninger's algorithm if and only if all $\gamma_1, \dots, \gamma_n$ are the same values. However, our algorithm provides better bounds in different cases. It is clear that our algorithm covers Howgrave-Graham's algorithm in the asymptotic case $\gamma_2, \gamma_3, \dots, \gamma_n \to \beta$. Figure 1 compares the condition for our algorithm and Cohn and Heninger's algorithm for $(2, 1)$-ME with $\beta = 0.5$ to be solved. Cohn and Heninger's algorithm can work in the grey area. Our algorithm provides better results in the black area.

In the case of $(1, n)$-ME, we assume $\gamma_1 \geq \gamma_2 \geq \cdots \geq \gamma_n$ without loss of generality. Based on our strategy for polynomials selection, we show that Herrmann and May's algorithm can be improved if and only if $\gamma_1 > \beta(1 - \sqrt[n]{1 - \beta})$.

**Table 1.** The comparison of the bounds for $(1,2)$-ME with $\beta = 0.5$ to be solved

| $\gamma_2$ | Herrmann and May's $\gamma_1$ | Sarkar's $\gamma_1$ | Our $\gamma_1$ |
|---|---|---|---|
| 0 | 0.207107 | 0.25 | 0.25 |
| 0.01 | 0.197107 | 0.209391 | 0.209446 |
| 0.02 | 0.187107 | 0.192456 | 0.192513 |
| 0.03 | 0.177107 | 0.179215 | 0.179256 |
| 0.04 | 0.167107 | 0.16773 | 0.167749 |
| 0.05 | 0.157107 | 0.157193 | 0.157196 |
| 0.06 | 0.147107 | 0.147107 | 0.147107 |

Our algorithm and Sarkar's algorithm become identical to Herrmann and May's algorithm with smaller $\gamma_1$. The conditions for $(1,2)$-ME with $\gamma_1 > \beta(1-\sqrt{1-\beta})$ to be solved by all known algorithms can be written as

$$\gamma_1 + \gamma_2 < \tau^3 - 3\tau^2 + 3\beta\tau + \varDelta_2. \qquad (1)$$

Herrmann and May's algorithm can solve $(1,2)$-ME provided that the condition (1) holds where

$$\varDelta_2 = 0,\ \tau = 1 - \sqrt{1-\beta}.$$

Our strategy enables us to optimize the parameters of Sarkar's algorithm, too.

Our algorithm can solve $(1,2)$-ME provided that the condition (1) holds where

$$\varDelta_2 = \frac{(\gamma_1 - \beta\tau)^3}{\gamma_1(\gamma_1 - \gamma_2)},\ \tau = \frac{\gamma_2}{\beta - \sqrt{\gamma_1 - \gamma_2}}.$$

These algorithms provide better bounds by the existence of positive $\varDelta_2$. Furthermore, our algorithm can extend to general $n$-variate cases unlike Sarkar's work. We prove that our algorithm covers Howgrave-Graham's algorithm in the asymptotic case $\gamma_2, \gamma_3, \ldots, \gamma_n \to 0$. Table 1 compares the bounds for these algorithms for $(1,2)$-ME with $\beta = 0.5, \gamma_1 > \beta(1-\sqrt{1-\beta})$ to be solved.

**Organization of the Paper.** In Section 2, we introduce Howgrave-Graham's lemma and the LLL algorithm to use Coppersmith's method and define the problems. In Section 3, we explain our strategy for selecting polynomials. In Section 4, we give the analysis of $(n,1)$-ME. In Section 5, we give the analysis of $(1,n)$-ME. In Section 6, we implement our algorithms.

## 2 Preliminaries

Using the lattice-based Coppersmith's method [5], we can find small solutions of modular equations. Here, we introduce the reformulation of Howgrave-Graham [16]. For $n$-variate polynomials $h(x_1,\ldots,x_n) = \sum h_{i_1,\ldots,i_n} x_1^{i_1} \cdots x_n^{i_n}$, define the norm of polynomials $\|h(x_1,\ldots,x_n)\| := \sqrt{\sum h^2_{i_1,\ldots,i_n}}$. Howgrave-Graham proposed the following lemma that reduces modular equations into integer polynomials [16].

**Lemma 1 (Howgrave-Graham's lemma [16]).** *Let $h(x_1,\ldots,x_n) \in \mathbb{Z}[x_1,\ldots,x_n]$ be a polynomial, which consists of at most $w$ monomials. Let $\phi, m, X_1,\ldots,X_n$ be positive integers. Consider the case that the polynomial $h(x_1,\ldots,x_n)$ satisfies*
*1. $h(\bar{x}_1,\ldots,\bar{x}_n) = 0 \pmod{\phi^m}$, where $|\bar{x}_1| < X_1,\ldots,|\bar{x}_n| < X_n$,*
*2. $\|h(x_1X_1,\ldots,x_nX_n)\| < \phi^m/\sqrt{w}$.*
*Then $h(\bar{x}_1,\ldots,\bar{x}_n) = 0$ holds over the integers.*

To find new polynomials that satisfy Howgrave-Graham's lemma, we use the LLL algorithm. Let $\{\mathbf{b}_1,\ldots,\mathbf{b}_d\}$ be linearly independent $w$-dimensional vectors. The lattice $L(\mathbf{b}_1,\ldots,\mathbf{b}_d)$ spanned by the basis vectors $\{\mathbf{b}_1,\ldots,\mathbf{b}_d\}$ is defined as $L(\mathbf{b}_1,\ldots,\mathbf{b}_d) = \{\sum_{j=1}^d x_j\mathbf{b}_j : x_j \in \mathbb{Z}\}$. We call $d$ the rank of the lattice, and $w$ the dimension of the lattice. With $d = w$, the lattice is described as full-rank. The basis matrix of the lattice $B$ is defined as the $d \times w$ matrix that has basis vectors $\{\mathbf{b}_1,\ldots,\mathbf{b}_d\}$ in each row. In this paper, we use only full-rank lattices. The volume of the full-rank lattice is computed by $\mathrm{vol}(L(B)) = |\det(B)|$.

In 1982, Lenstra, Lenstra and Lovász proposed the LLL algorithm [19] that can find vectors in polynomial time whose norm is small enough to satisfy the following condition [20].

**Proposition 1 (LLL algorithm [19]).** *Given a lattice spanned by $w$-dimensional basis vectors $\{\mathbf{b}_1,\ldots,\mathbf{b}_d\}$, the LLL algorithm finds new reduced bases $\{\mathbf{b}'_1,\ldots,\mathbf{b}'_n\}$ that satisfy*

$$\|\mathbf{b}'_1\| \le 2^{(d-1)/4}(\mathrm{vol}(L))^{1/d},\ \|\mathbf{b}'_n\| \le 2^{d(d-1)/4(d-n+1)}(\mathrm{vol}(L))^{1/(d-n+1)}.$$

*These norms are all Euclidean norms. The running time of the LLL algorithm is $O(d^5w(\log B)^3)$ where $\log B = \log \max_j |\mathbf{b_j}|$.*

This condition is the worst case proved in [19]. It is widely known that the LLL algorithm tends to output the vectors whose norms are much smaller than theoretically predicted.

We should mention that the Coppersmith's method requires a heuristic argument in solving multivariate equations. This is because each new polynomial whose coefficients consist of the LLL-reduced bases has no assurance of algebraically independence. In this paper, we assume that the polynomials whose coefficients consist of LLL-reduced bases are algebraically independent and the resultant of these polynomials will not vanish.

We define the problems introduced in Section 1.

**Definition 1 ($(n,1)$-ME).** *Given positive integers $N = pq, a_1,\ldots,a_n$, and given $(0,1)$ real numbers $\gamma_1,\ldots,\gamma_n,\beta$, we want to find all integers $r_1,\ldots,r_n$ such that $|r_1| \le N^{\gamma_1},\ldots,|r_n| \le N^{\gamma_n}, p \ge N^{\beta}$, and*

$$\begin{cases} a_1 + r_1 = 0 \pmod{p}, \\ \vdots \\ a_n + r_n = 0 \pmod{p}. \end{cases}$$

**Definition 2** (**$(1,n)$-ME**)**.** *Given positive integers $N = pq, a_0, a_1, \dots, a_n$, and given* $(0,1)$ *real numbers $\gamma_1, \dots, \gamma_n, \beta$, we want to find all integers $r_1, \dots, r_n$ such that $|r_1| \le N^{\gamma_1}, \dots, |r_n| \le N^{\gamma_n}, p \ge N^{\beta}$, and*

$$a_0 + a_1 r_1 + \cdots + a_n r_n = 0 \pmod{p}.$$

We write $X_j := N^{\gamma_j}$ throughout the paper. In the case of $(1,n)$-ME, we assume $\gamma_1 \ge \gamma_2 \ge \cdots \ge \gamma_n$ and $a_1 = 1$ without loss of generality. If $a_1 \ne 1$, we multiply with the inverse of $a_1$ modulo $N$, or we find a non-trivial factorization of $N$. For both problems, $N$ is large enough not to factorize.

## 3 The Strategy for Lattice Constructions

We briefly explain the Coppersmith's method to solve the equations modulo $\phi$. We put lattice bases the coefficients vectors of polynomials that have the same roots as the original equations modulo $\phi^m$ with positive integer $m$. We have new polynomials whose coefficients are the elements of LLL-reduced bases. The new polynomials have the same roots as the original equations modulo $\phi^m$, and the norms of polynomials are small. We can solve the $n$-variate equations if new $n$ polynomials satisfy Howgrave-Graham's lemma. This condition can be written as

$$2^{d(d-1)/4(d-n+1)}(\det(B))^{1/(d-n+1)} < \frac{\phi^m}{\sqrt{d}}.$$

As in previous works, we neglect small terms[1] and rewrite the condition as

$$(\det(B))^{1/d} < \phi^m. \tag{2}$$

Our strategy for lattice construction to improve the algorithms is very simple. In previous works [4,15], the basis matrixes are triangular. The left side of the condition (2) can be computed as the geometric mean of all diagonals of the basis matrix. We call polynomials whose diagonals are less than $\phi^m$ *helpful* as in [21]. Conversely, we call polynomials whose diagonals are larger than $\phi^m$ non-helpful. Helpful polynomials contribute to satisfy the condition (2). The more we add helpful polynomials to the lattice bases, the better the conditions for modular equations to be solved become. We should select as many helpful polynomials

[1] This approximation can be used if $\beta \gg 1/\sqrt{\log N}$. In the context of fully homomorphic encryption over the integers, we can not neglect the small terms (e.g. see [4]).

as possible and select as few non-helpful polynomials as possible in constructing lattices under the constraint for the basis matrix to be triangular.

There are some cases that we cannot hold the basis matrix triangular when adding a helpful polynomial, but we can hold the basis matrix triangular when adding $w'$ polynomials. If the geometric mean of the diagonals of the $w'$ polynomials is less than $\phi^m$, these polynomials contribute to satisfy the condition (2). We call these polynomials *consecutive helpful polynomials*. As is the case for helpful polynomials, we should select as many consecutive helpful polynomials as possible in constructing lattices. Based on this strategy, we can optimize the parameters before calculating the condition (2) unlike previous works.

## 4 Improved Algorithm for $(n, 1)$-ME

To solve $(n, 1)$-ME, we use the shift-polynomials as Cohn and Heninger with positive integer $m$,

$$f_{[i_1,\ldots,i_n]}(x_1,\ldots,x_n) = (a_1 + x_1)^{i_1} \cdots (a_n + x_n)^{i_n} N^{\max\{m-\sum_{j=1}^n i_j, 0\}}.$$

If all indexes $i_1, \ldots, i_n$ are non-negative integers, the shift-polynomials modulo $p^m$ have the same solutions as the original equations for all $x_1, x_2, \ldots, x_n$; that is, $f_{[i_1,\ldots,i_n]}(r_1,\ldots,r_n) = 0 \pmod{p^m}$. We span the lattice generated by the coefficient vectors of shift-polynomials $f_{[i_1,\ldots,i_n]}(x_1X_1,\ldots,x_nX_n)$.

Cohn and Heninger selected the shift-polynomials that satisfy $0 \le \sum_{j=1}^n i_j \le \tau m$. Considering the condition (2), Cohn and Heninger optimized $\tau = \beta^{-1/n}$ and concluded that $(n, 1)$-ME can be solved provided that $(\gamma_1 + \cdots + \gamma_n)/n < \beta^{(n+1)/n}$.

We change the shift-polynomials to be selected and improve the condition for $(n, 1)$-ME to be solved. Each diagonal of the shift-polynomial is $N^{\sum_{j=1}^n \gamma_j i_j + \max\{m-\sum_{j=1}^n i_j, 0\}}$. To select as many helpful polynomials as possible under the constraint for the basis matrix to be triangular, we select the shift-polynomials that satisfy

$$0 \le \sum_{j=1}^{n} \gamma_j i_j \le \beta m.$$

Since we select more helpful polynomials and less non-helpful polynomials than Cohn and Heninger's lattice, our algorithm is expected to improve the algorithm. Figure 2 describes an example of the basis matrix for $(2, 1)$-ME with $\gamma_1 = 0.4, \gamma_2 = 0.1, \beta = 0.3, m = 2$. We define the polynomial order $\prec$ as $x_1^{i_1} x_2^{i_2} \cdots x_n^{i_n} \prec x_1^{i_1'} x_2^{i_2'} \cdots x_n^{i_n'}$ if

$$\sum_{j=1}^{n} i_j < \sum_{j=1}^{n} i_j' \quad \text{or} \quad \sum_{j=1}^{n} i_j = \sum_{j=1}^{n} i_j',\ i_j = i_j' (j = 1, 2, \ldots, t),\ i_{t+1} < i_{t+1}'.$$

Ordered in this way, the basis matrixes become triangular in general.

| $f_{[i_1,i_2]}$ | $1$ | $x_1$ | $x_2$ | $x_1x_2$ | $x_2^2$ | $x_1x_2^2$ | $x_2^3$ | $x_2^4$ | $x_2^5$ | $x_2^6$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $f_{[0,0]}$ | $N^2$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ |
| $f_{[1,0]}$ | $Na_1$ | $NX_1$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ |
| $f_{[0,1]}$ | $Na_2$ | $0$ | $NX_2$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ |
| $f_{[1,1]}$ | $a_1a_2$ | $a_2X_1$ | $a_1X_2$ | $X_1X_2$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ |
| $f_{[0,2]}$ | $a_2^2$ | $0$ | $2a_2X_2$ | $0$ | $X_2^2$ | $0$ | $0$ | $0$ | $0$ | $0$ |
| $f_{[1,2]}$ | $a_1a_2^2$ | $a_2^2X_1$ | $2a_1a_2X_2$ | $2a_2X_1X_2$ | $a_1X_2^2$ | $X_1X_2^2$ | $0$ | $0$ | $0$ | $0$ |
| $f_{[0,3]}$ | $a_2^3$ | $0$ | $3a_2^2X_2$ | $0$ | $3a_2X_2^2$ | $0$ | $X_2^3$ | $0$ | $0$ | $0$ |
| $f_{[0,4]}$ | $a_2^4$ | $0$ | $4a_2^3X_2$ | $0$ | $6a_2^2X_2^2$ | $0$ | $4a_2X_2^3$ | $X_2^4$ | $0$ | $0$ |
| $f_{[0,5]}$ | $a_2^5$ | $0$ | $5a_2^4X_2$ | $0$ | $10a_2^3X_2^2$ | $0$ | $10a_2^2X_2^3$ | $5a_2X_2^4$ | $X_2^5$ | $0$ |
| $f_{[0,6]}$ | $a_2^6$ | $0$ | $6a_2^5X_2$ | $0$ | $15a_2^4X_2^2$ | $0$ | $20a_2^3X_2^3$ | $15a_2^2X_2^4$ | $6a_2X_2^5$ | $X_2^6$ |

**Fig. 2.** The example of basis matrixes to solve $(2,1)$-ME with $\gamma_1 = 0.4, \gamma_2 = 0.1, \beta = 0.3, m = 2$

To derive the condition for $(n,1)$-ME to be solved, we compute the dimension as $d$, where

$$d = \sum_{\gamma_1 i_1 + \cdots + \gamma_n i_n = 0}^{\beta m} 1 = \frac{m^n}{n!} \frac{\beta^n}{\gamma_1 \cdots \gamma_n} + o(m^n),$$

and the volume as $\det(B) = N^{s_N} {X_1}^{s_{X_1}} \cdots {X_n}^{s_{X_n}}$, where

$$s_N = \sum_{i_1 + \cdots + i_n = 0}^{m} \binom{\sum_{j=1}^n i_j + n - 1}{n-1} (m - \sum_{j=1}^n i_j) = \frac{m^{n+1}}{(n+1)!} + o(m^{n+1}),$$

$$s_{X_j} = \sum_{\gamma_1 i_1 + \cdots + \gamma_n i_n = 0}^{\beta m} i_j = \frac{m^{n+1}}{(n+1)!} \frac{\beta^{n+1}}{\gamma_1 \cdots \gamma_{j-1} \gamma_j^2 \gamma_{j+1} \cdots \gamma_n} + o(m^{n+1}),$$

for each $s_{X_1}, s_{X_2}, \ldots, s_{X_n}$. Ignoring low order terms of $m$, we have the following result considering condition (2), that is, $(s_N + \sum_{j=1}^n \gamma_j s_{X_j})/d < \beta m$.

**Theorem 1 ($(n,1)$-ME algorithm).** *$(n,1)$-ME can be solved provided that*

$$\sqrt[n]{\gamma_1 \cdots \gamma_n} < \beta^{(n+1)/n}.$$

*The running time of the algorithm is polynomial in $\log N$ and exponential in $n$.*

As the application of the algorithm, we analyze implicit factorization in the full version. Since our $(n,1)$-ME algorithm is superior to that of [24], we achieve better cryptanalytic bounds.

Cohn and Heninger also considered the degrees generalization of $(n,1)$-ME [4]. Consider the simultaneous modular univariate equations,

$$\begin{cases} h_1(x_1) = 0 \pmod{p}, \\ \qquad \vdots \\ h_n(x_n) = 0 \pmod{p}. \end{cases}$$

There are $n$ equations and each equation $h_j(x_j)$ has one variable of degrees $\delta_j$. We call the problem $(n,1)^\delta$-ME. In [20], May analyzed $(1,1)^\delta$-ME. May's algorithm can solve $(1,1)^\delta$-ME provided that $\delta_1\gamma_1 < \beta^2$. Cohn and Heninger [4] proposed an algorithm to solve $(n,1)^\delta$-ME provided that $(\delta_1\gamma_1+\cdots+\delta_n\gamma_n)/n < \beta^{(n+1)/n}$. As the case of $(n,1)$-ME, our method can improve $(n,1)^\delta$-ME algorithm.

**Theorem 2 ($(n,1)^\delta$-ME algorithm).** *$(n,1)^\delta$-ME can be solved in polynomial time provided that*

$$\sqrt[n]{\delta_1\gamma_1\cdots\delta_n\gamma_n} < \beta^{(n+1)/n}.$$

*The running time of the algorithm is polynomial in $\log N$ and exponential in $n, \delta_1, \ldots, \delta_n$.*

The proof is written in the full version.

## 5 Improved Algorithm for $(1,n)$-ME

### 5.1 Improved Algorithm for $(1,2)$-ME

First, we analyze $(1,2)$-ME. To solve $(1,2)$-ME, we use the shift-polynomials as previous works with positive integers $m, t$ and the parameter $\tau = t/m > 0$ to be optimized later,

$$g_{[i_1,i_2]}(x_1,x_2) = x_2^{i_2}(a_0 + x_1 + a_2x_2)^{i_1} N^{\max\{t-i_1,0\}}.$$

If both indexes $i_1, i_2$ are non-negative integers, the shift-polynomials modulo $p^t$ have the same solutions as the original equations for both $x_1, x_2$; that is, $g_{[i_1,i_2]}(r_1,r_2) = 0 \pmod{p^t}$. We span the lattice generated by the coefficient vectors of shift-polynomials $g_{[i_1,i_2]}(x_1X_1, x_2X_2)$. Each diagonal of the shift-polynomial is $N^{\gamma_1 i_1+\gamma_2 i_2+\max\{t-i_1,0\}}$. To select as many helpful polynomials as possible under the constraint for the basis matrix to be triangular, we select the shift-polynomials that satisfy

$$0 \le i_1 + i_2 \le m \quad \text{and} \quad 0 \le \gamma_1 i_1 + \gamma_2 i_2 \le \beta t. \tag{3}$$

Herrmann and May selected the shift-polynomials that satisfy only the first inequality of (3). They optimized $\tau = 1 - \sqrt{1-\beta}$ and concluded that $(1,2)$-ME can be solved provided that $\gamma_1 + \gamma_2 < 3\beta - 2 + 2(1-\beta)^{3/2}$. Since the second inequality of (3) eliminates non-helpful polynomials, our algorithm is expected to improve the algorithm. Sarkar selected the shift-polynomials that satisfy the first inequality of (3) and $0 \le i_1 \le k$ with another parameter $\eta = k/m > 0$. For our condition (3) can select more helpful polynomials than Sarkar, our algorithm is expected to improve the algorithm. Figure 3 describes an example of the basis matrix for $(1,2)$-ME with $\gamma_1 = 0.2, \gamma_2 = 0.1, \beta = 0.5, m = 4, \tau = 0.25$.

| $g_{[i_1,i_2]}$ | $1$ | $x_2$ | $x_1$ | $x_2^2$ | $x_1x_2$ | $x_1^2$ | $x_2^3$ | $x_1x_2^2$ | $x_2^4$ | $x_1x_2^3$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $g_{[0,0]}$ | $N$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ |
| $g_{[0,1]}$ | $0$ | $NX_2$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ |
| $g_{[1,0]}$ | $a_0$ | $a_2X_2$ | $X_1$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ |
| $g_{[0,2]}$ | $0$ | $0$ | $0$ | $NX_2^2$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ |
| $g_{[1,1]}$ | $0$ | $a_0X_2$ | $0$ | $a_2X_2^2$ | $X_1X_2$ | $0$ | $0$ | $0$ | $0$ | $0$ |
| $g_{[2,0]}$ | $a_0^2$ | $2a_0a_2X_2$ | $2a_0X_1$ | $a_2^2X_2^2$ | $2a_2X_1X_2$ | $X_1^2$ | $0$ | $0$ | $0$ | $0$ |
| $g_{[0,3]}$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $NX_2^3$ | $0$ | $0$ | $0$ |
| $g_{[1,2]}$ | $0$ | $0$ | $0$ | $a_0X_2^2$ | $0$ | $0$ | $a_2X_2^3$ | $X_1X_2^2$ | $0$ | $0$ |
| $g_{[0,4]}$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $NX_2^4$ | $0$ |
| $g_{[1,3]}$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $a_0X_2^3$ | $0$ | $a_2X_2^4$ | $X_1X_2^3$ |

**Fig. 3.** The example of basis matrixes to solve $(1,2)$-ME with $\gamma_1 = 0.2, \gamma_2 = 0.1, \beta = 0.5, m = 4, \tau = 0.25$

We should consider the case that the sets of indexes $(i_1, i_2)$ that satisfy the second inequality of (3) always satisfy the first inequality. In this case, our lattices become identical to Herrmann and May's lattices. Such situations occur provided that $\gamma_1 \leq \beta\tau$. According to the result of Herrmann and May, our algorithm achieves better bounds than Herrmann and May's algorithm if and only if $\gamma_1 > \beta(1-\sqrt{1-\beta})$. We analyze the case $\gamma_1 > \beta(1-\sqrt{1-\beta}) > \gamma_2$. In this case, we can eliminate non-helpful polynomials from the lattices and improve the algorithm.

We consider the optimization of $\tau$, $\gamma_1 > \beta\tau > \gamma_2$. The condition (3) can be rewritten as

$$0 \leq i_1 \leq \frac{\beta\tau - \gamma_2}{\gamma_1 - \gamma_2} m;\ 0 \leq i_2 \leq m - i_1,$$
$$\frac{\beta\tau - \gamma_2}{\gamma_1 - \gamma_2} m < i_1 \leq \frac{\beta\tau}{\gamma_1} m;\ 0 \leq i_2 \leq \frac{\beta\tau m - \gamma_1 i_1}{\gamma_2}.$$

The diagonals of the shift-polynomials that satisfy $(\beta\tau - \gamma_2)m/(\gamma_1 - \gamma_2) < i_1 \leq \beta\tau m/\gamma_1$ and $i_2 = (\beta\tau m - \gamma_1 i_1)/\gamma_2$ are all $N^{\beta\tau m}$. That is, with $(\beta\tau - \gamma_2)m/(\gamma_1 - \gamma_2) < i_1 \leq \beta\tau m/\gamma_1$, we select all helpful-polynomials and no non-helpful polynomials.

We consider the shift-polynomials that satisfy $0 \leq i_1 \leq (\beta\tau - \gamma_2)m/(\gamma_1 - \gamma_2)$ in the same way. In this case, we should consider the constraint for the basis matrix to be triangular. For the shift-polynomials that satisfy $i_2 = m - i_1$, we do not consider the shift-polynomials individually but consider whether all shift-polynomials are consecutive helpful polynomials or not. We should optimize $\tau$ for the geometric mean of the diagonals of those shift-polynomials to be $N^{\beta\tau m}$. In this case, with $0 \leq i_1 \leq (\beta\tau - \gamma_2)m/(\gamma_1 - \gamma_2)$, we select all consecutive helpful-polynomials and no consecutive non-helpful polynomials. We compute the number of shift-polynomials that satisfy $0 \leq i_1 \leq (\beta\tau - \gamma_2)m/(\gamma_1 - \gamma_2), i_2 = m - i_1$ as $d'$, where

$$d' = \sum_{i_1=0}^{(\beta\tau-\gamma_2)m/(\gamma_1-\gamma_2)} 1 = \frac{\beta\tau-\gamma_2}{\gamma_1-\gamma_2}m + o(m),$$

and the product of diagonals of such shift-polynomials as $N^{s'_N} X_1^{s'_{X_1}} X_2^{s'_{X_2}}$, where

$$\begin{aligned}
s'_N &= \sum_{i_1=0}^{\tau m}(\tau m - i_1) = \frac{m^2}{2}\tau^2 + o(m^2),\\
s'_{X_1} &= \sum_{i_1=0}^{(\beta\tau-\gamma_2)m/(\gamma_1-\gamma_2)} i_1 = \frac{m^2}{2}(\frac{\beta\tau-\gamma_2}{\gamma_1-\gamma_2})^2 + o(m^2),\\
s'_{X_2} &= \sum_{i_1=0}^{(\beta\tau-\gamma_2)m/(\gamma_1-\gamma_2)} (m-i_1) = \frac{m^2}{2}\left(1-(\frac{\gamma_1-\beta\tau}{\gamma_1-\gamma_2})^2\right) + o(m^2).
\end{aligned}$$

For the geometric mean of the diagonals of those shift-polynomials to be $N^{\beta\tau m}$, $(s'_N + \gamma_1 s'_{X_1} + \gamma_2 s'_{X_2})/d' = \beta m$ should hold. Substituting the value and we have the equation $(\beta^2 - \gamma_1 + \gamma_2)\tau^2 - 2\gamma_2\beta\tau + \gamma_2^2 = 0$. Solving the equation, and we have the value $\tau = \gamma_2/(\beta - \sqrt{\gamma_1-\gamma_2})$. The other solution $\tau = \gamma_2/(\beta + \sqrt{\gamma_1-\gamma_2})$ does not satisfy $\gamma_2 < \beta\tau$. This discussion can also be applied to Sarkar's algorithm. We optimize the parameters of Sarkar's algorithm in the full version.

To derive the condition for $(1,2)$-ME to be solved, we compute the dimension as $d$, where

$$\begin{aligned}
d &= \sum_{i_1=0}^{(\beta\tau-\gamma_2)m/(\gamma_1-\gamma_2)} \sum_{i_2=0}^{m-i_1} 1 + \sum_{i_1=(\beta\tau-\gamma_2)m/(\gamma_1-\gamma_2)}^{\beta\tau m/\gamma_1} \sum_{i_2=0}^{(\beta\tau m-\gamma_1 i_1)/\gamma_2} 1\\
&= \frac{m^2}{2}\left(1 - \frac{(\gamma_1-\beta\tau)^2}{\gamma_1(\gamma_1-\gamma_2)}\right) + o(m^2),
\end{aligned}$$

and the volume as $\det(B) = N^{s_N} X_1^{s_{X_1}} X_2^{s_{X_2}}$, where

$$\begin{aligned}
s_N &= \sum_{i_1=0}^{\tau m}(m+1-i_1)(\tau m - i_1) = \frac{m^3}{6}\left(-\tau^3 + 3\tau^2\right) + o(m^3),\\
s_{X_1} &= \sum_{i_1=0}^{(\beta\tau-\gamma_2)m/(\gamma_1-\gamma_2)} \sum_{i_2=0}^{m-i_1} i_1 + \sum_{i_1=(\beta\tau-\gamma_2)m/(\gamma_1-\gamma_2)}^{\beta\tau m/\gamma_1} \sum_{i_2=0}^{(\beta\tau m-\gamma_1 i_1)/\gamma_2} i_1\\
&= \frac{m^3}{6}\left(1 - 3(\frac{\gamma_1-\beta\tau}{\gamma_1-\gamma_2})^2(1 - \frac{3\gamma_2\beta\tau + 2(\gamma_1+\gamma_2)(\gamma_1-\beta\tau)}{3\gamma_1^2})\right) + o(m^3),\\
s_{X_2} &= \sum_{i_1=0}^{(\beta\tau-\gamma_2)m/(\gamma_1-\gamma_2)} \sum_{i_2=0}^{m-i_1} i_2 + \sum_{i_1=(\beta\tau-\gamma_2)m/(\gamma_1-\gamma_2)}^{\beta\tau m/\gamma_1} \sum_{i_2=0}^{(\beta\tau m-\gamma_1 i_1)/\gamma_2} i_2\\
&= \frac{m^3}{6}\left(1 - \frac{(\gamma_1-\beta\tau)^3}{\gamma_1(\gamma_1-\gamma_2)^2}\right) + o(m^3).
\end{aligned}$$

Ignoring low order terms of $m$ and considering condition (2), that is, $(s_N + \gamma_1 s_{X_1} + \gamma_2 s_{X_2})/d < \beta m$ and we have

$$\gamma_1 + \gamma_2 < \tau^3 - 3\tau^2 + 3\beta\tau + \frac{(\gamma_1 - \beta\tau)^3}{\gamma_1(\gamma_1 - \gamma_2)} \quad \text{where} \quad \tau = \frac{\gamma_2}{\beta - \sqrt{\gamma_1 - \gamma_2}}.$$

We substitute the value of $\tau$ and have the following result.

**Theorem 3 ((1, 2)-ME algorithm).** *(1, 2)-ME with $\gamma_1 > \beta(1 - \sqrt{1-\beta})$ can be solved in polynomial time provided that*

$$\gamma_1(3\beta - \gamma_2 - 2\sqrt{\gamma_1 - \gamma_2}) < \beta^3.$$

*The running time of the algorithm is polynomial in* $\log N$.

Unlike Herrmann and May's algorithm, this algorithm satisfies the following property.

**Proposition 2.** *With $\gamma_2 \to 0$, the algorithm of Theorem 3 can solve (1, 2)-ME provided that $\gamma_1 < \beta^2$.*

*Proof.* First, we show that the algorithm does not work if $\gamma_1 > \beta^2$. Consider the case $\gamma_1 < \beta'$. Here, $\beta'$ is a real number that satisfies $\beta' > \beta^2$. With a positive real number $\epsilon$, we rewrite $\gamma_1 = \beta' - \epsilon$. The parameter $\tau$ can be rewritten as $\tau = \gamma_2/(\beta - \sqrt{\beta' - \epsilon - \gamma_2})$. If $\epsilon \ll \beta^2$ and $\gamma_2$ is sufficiently small, $\tau < 0$. This is inconsistent with the definition of $\tau$. Therefore, the algorithm cannot solve (1, 2)-ME with $\gamma_1 > \beta^2$.

Next, we show that the algorithm works provided that $\gamma_1 < \beta^2$. With positive real numbers $\epsilon \ll \beta^2$ and sufficiently large $u$, we rewrite $\gamma_1 = \beta^2 - \epsilon, \gamma_2 = \epsilon^u$. We can calculate as

$$\begin{aligned}
&\gamma_1(3\beta - \gamma_2 - 2\sqrt{\gamma_1 - \gamma_2}) - \beta^3 \\
&= (\beta^2 - \epsilon)(3\beta - \epsilon^u - 2\sqrt{\beta^2 - \epsilon - \epsilon^u}) - \beta^3 \\
&= 2(\beta^2 - \epsilon)\frac{\epsilon + \epsilon^u}{\beta + \sqrt{\beta^2 - \epsilon - \epsilon^u}} - \beta\epsilon - (\beta^2 - \epsilon)\epsilon^u \\
&= \frac{2\beta^2\epsilon}{\beta + \sqrt{\beta^2 - \epsilon - \epsilon^u}} - \frac{2\epsilon(\epsilon + \epsilon^u)}{\beta + \sqrt{\beta^2 - \epsilon - \epsilon^u}} - \beta\epsilon \\
&+ \frac{2\beta^2\epsilon^u}{\beta + \sqrt{\beta^2 - \epsilon - \epsilon^u}} - (\beta^2 - \epsilon)\epsilon^u \\
&= -\frac{\epsilon(\epsilon + \epsilon^u)((\beta + 2\sqrt{\beta^2 - \epsilon - \epsilon^u})}{(\beta + \sqrt{\beta^2 - \epsilon - \epsilon^u})^2} + o(\epsilon^u) < 0.
\end{aligned}$$

Therefore, the condition of Theorem 3 holds with sufficiently large $u$. That is, Proposition 2 is proved. □

## 5.2 Extension to $(1, n)$-ME

To extend Theorem 3 to general $n$-variate cases, we use the shift-polynomials as Herrmann and May with positive integers $m, t$, and the parameter $\tau = t/m > 0$ to be optimized later,

$$g_{[i_1,\dots,i_n]}(x_1,\dots,x_n) = x_2^{i_2}\cdots x_n^{i_n}(a_0 + x_1 + a_2x_2 + \cdots + a_nx_n)^{i_1}N^{\max\{t-i_1,0\}}.$$

If all indexes $i_1,\dots,i_n$ are non-negative integers, these shift-polynomials modulo $p^t$ have the same solutions as the original equations for all $x_1, x_2, \dots, x_n$; that is, $g_{[i_1,\dots,i_n]}(r_1,\dots,r_n) = 0 \pmod{p^t}$. We span the lattice generated by the coefficient vectors of shift-polynomials $g_{[i_1,\dots,i_n]}(x_1X_1,\dots,x_nX_n)$. Each diagonal of the shift-polynomial is $N^{\sum_{j=1}^n \gamma_j i_j + \max\{t-i_1,0\}}$. To select as many helpful polynomials as possible under the constraint for the basis matrix to be triangular, we select the shift-polynomials that satisfy

$$0 \le \sum_{j=1}^n i_j \le m \quad \text{and} \quad 0 \le \sum_{j=1}^n \gamma_j i_j \le \beta t. \tag{4}$$

Herrmann and May selected the shift-polynomials that satisfy only the first inequality of (4). They optimized $\tau = 1 - \sqrt[n]{1-\beta}$ and concluded that $(1,n)$-ME can be solved provided that $\sum_{i=1}^n \gamma_i < 1 - (n+1)(1-\beta) + n(1-\beta)^{(n+1)/n}$. Because the second inequality of (4) can eliminate non-helpful polynomials, our algorithm is expected to improve the algorithm. We define the polynomial order $\prec$ for $n$-variate cases as $x_1^{i_1}x_2^{i_2}\cdots x_n^{i_n} \prec x_1^{i'_1}x_2^{i'_2}\cdots x_n^{i'_n}$ if

$$\sum_{j=1}^n i_j < \sum_{j=1}^n i'_j \quad \text{or} \quad \sum_{j=1}^n i_j = \sum_{j=1}^n i'_j,\ i_j = i'_j (j = t, t+1, \dots, n),\ i_{t-1} < i'_{t-1}.$$

Ordered in this way, the basis matrixes become triangular in general.

As the case of $(1,2)$-ME, our algorithms are superior to Herrmann and May's algorithm if and only if $\gamma_1 > \beta(1 - \sqrt[n]{1-\beta})$. We analyze the case $\gamma_1 > \beta(1 - \sqrt[n]{1-\beta}) > \gamma_2$. However, the calculation of the dimension and the volume of this lattice is too complicated. We relax the condition (4) and select the shift-polynomials that satisfy

$$0 \le \sum_{j=1}^n i_j \le m \quad \text{and} \quad 0 \le \gamma_1 i_1 + \bar{\gamma}\sum_{j=2}^n i_j \le \beta t. \tag{5}$$

The new parameter $\bar{\gamma}$ should be optimized later. This polynomials selection is the exact multivariate generalization of Theorem 3 if and only if $\gamma_2 = \cdots = \gamma_n$. However, this relaxed algorithm is superior to Herrmann and May's algorithm for the second inequality of (5) can eliminate non-helpful polynomials. Selecting polynomials according to the condition (5), we have the following result.

**Theorem 4 (relaxed $(1,n)$-ME algorithm).** *$(1,n)$-ME with $\gamma_1 > \beta(1-\sqrt[n]{1-\beta}) > \gamma_2$ can be solved in polynomial time provided that*

$$\sum_{j=1}^{n}\gamma_j < 1-(n+1)(1-\beta)\tau-(1-\tau)^{n+1}+\Delta_{n(1)},$$

*where $\Delta_{n(1)} = (\gamma_1-\beta\tau)^{n+1}/(\gamma_1(\gamma_1-\bar{\gamma})^{n-1}), \bar{\gamma} = \sum_{j=2}^{n}\gamma_j/(n-1)$, and $\tau$ is the solution of the equation*

$$(1-\tau)^n - \frac{(\gamma_1-\beta\tau)^n}{(\gamma_1-\bar{\gamma})^{n-1}} + n(1-\beta)\tau - 1 + \sum_{j=1}^{n}\gamma_j = 0,$$

*with $\gamma_1 > \beta\tau > \bar{\gamma}$. The running time of the algorithm is polynomial in $\log N$ and exponential in $n$.*

We show the proof in the full version. Theorem 4 covers Theorem 3 with $n=2$. We present $(1,n)$-ME algorithm under more general setting in the full version, that is, $\gamma_k > \beta(1-\sqrt[n]{1-\beta}) \geq \gamma_{k+1}$ for some positive integer $k$.

Like our $(1,2)$-ME algorithm and unlike Herrmann and May's algorithm, this algorithm satisfies the following property.

**Proposition 3.** *With $\gamma_2,\gamma_3,\ldots,\gamma_n \to 0$, the algorithm of Theorem 4 can solve $(1,n)$-ME provided that $\gamma_1 < \beta^2$.*

*Proof.* We can get the value of parameter $\tau$ by solving the equation

$$(1-\tau)^n - \frac{(\gamma_1-\beta\tau)^n}{(\gamma_1-\bar{\gamma})^{n-1}} + n(1-\beta)\tau - 1 + \sum_{j=1}^{n}\gamma_j = 0.$$

We can rewrite the equation as

$$\sum_{k=2}^{n}\binom{n}{k}((\gamma_1-\bar{\gamma})^{n-1}-\gamma_1^{n-k}\beta^k)(-\tau)^k - n\beta\tau\sum_{l=1}^{n-1}\binom{n-1}{l}(-\bar{\gamma})^l\gamma_1^{n-1-l}$$
$$+ (\gamma_1-\bar{\gamma})^{n-1}\sum_{j=1}^{n}\gamma_j - \gamma_1^n = 0.$$

With positive real numbers $\epsilon \ll 1$ and sufficiently large $u$, we rewrite $\gamma_1 = \beta^2-\epsilon, \gamma_2 = \cdots = \gamma_n = \epsilon^u$ and assume $\tau = \Theta(\epsilon^v)$, the above equation can be rewritten as

$$-\frac{n(n-1)}{2}\gamma_1^{n-2}\epsilon\tau^2 + o(\epsilon^{2v+1}) + n(n-1)\beta\gamma_1^{n-2}\bar{\gamma}\tau + o(\epsilon^{u+v})$$
$$-\frac{n(n-1)}{2}\gamma_1^{n-2}\bar{\gamma}^2 + o(\epsilon^{2u}) = 0.$$

Consider the case that $v > u$, $\tau$ is the solution of the equation

$$-\frac{n(n-1)}{2}\gamma_1^{n-2}\bar{\gamma}^2 + o(\epsilon^{2u}) = 0.$$

This is the contradiction for there are no valid values of $\tau$.

Consider the case that $v = u$, $\tau$ is the solution of the equation

$$n(n-1)\beta\gamma_1^{n-2}\bar{\gamma}\tau - \frac{n(n-1)}{2}\gamma_1^{n-2}\bar{\gamma}^2 + o(\epsilon^{2u}) = 0.$$

Solving the equation, and we have $\tau = \bar{\gamma}/2\beta + o(\epsilon^{2u-1})$. This is the contradiction for $\beta\tau > \bar{\gamma}$ does not hold.

Consider the case that $u > v > u-1$, $\tau$ is the solution of the equation

$$-\frac{n(n-1)}{2}\gamma_1^{n-2}\bar{\gamma}^2 + o(\epsilon^{2u}) = 0.$$

This is the contradiction for there are no valid values of $\tau$.

Considering the case $v < u-1$, $\tau$ is the solution of the equation

$$-\frac{n(n-1)}{2}\gamma_1^{n-2}\epsilon\tau^2 + o(\epsilon^{2v+1}) = 0.$$

This is the contradiction for $\tau = \Theta(\epsilon^v)$ does not hold.

Considering the case $v = u-1$, $\tau$ is the solution of the equation

$$-\frac{n(n-1)}{2}\gamma_1^{n-2}\epsilon\tau^2 + n(n-1)\beta\gamma_1^{n-2}\bar{\gamma}\tau + o(\epsilon^{2u-1}) = 0.$$

Solving the equation, and we have $\tau = 2\beta\bar{\gamma}/\epsilon + o(\epsilon^{2u-1})$. Therefore, $\tau = \Theta(\epsilon^v) = \Theta(\epsilon^{u-1})$ holds and this is the valid parameter.

By the result of Theorem 4, $(1,n)$-ME can be solved provided that

$$\sum_{j=1}^{n}\gamma_j < 1 - (n+1)(1-\beta)\tau - (1-\tau)^{n+1} + \frac{(\gamma_1 - \beta\tau)^{n+1}}{\gamma_1(\gamma_1-\gamma_l)^{n-1}}.$$

We can rewrite the condition as

$$\begin{aligned}&\sum_{k=2}^{n+1}\binom{n+1}{k}(-\tau)^k(\gamma_1(\gamma_1-\bar{\gamma})^n - \beta^k\gamma_1^{n+1-k}(\gamma_1-\bar{\gamma}))\\ &-(n+1)\beta\gamma_1((\gamma_1-\bar{\gamma})^n - \gamma_1^{n-1}(\gamma_1-\bar{\gamma}))\tau\\ &+\gamma_1\left(\sum_{j=1}^{n}\gamma_j(\gamma_1^n - (\gamma_1-\bar{\gamma})^n) - n\gamma_1^n\bar{\gamma}\right) < 0.\end{aligned}$$

Consider the case that $\gamma_1 = \beta^2 - \epsilon, \gamma_2 = \cdots = \gamma_n = \epsilon^u$ and assume $\tau = \Theta(\epsilon^{u-1})$,

$$-\frac{(n+1)n}{2}\gamma_1^n\epsilon\tau^2 + (n+1)(n-1)\beta\gamma_1^n\bar{\gamma}\tau + o(\epsilon^{2u-1}) < 0.$$

Substitute the value $\bar{\gamma} = \epsilon^u, \tau = 2\beta\bar{\gamma}/\epsilon + o(\epsilon^{2u-1})$,

$$-2(n+1)\beta^2\gamma_1^n\epsilon^{2u-1} + o(\epsilon^{2u-1}) < 0.$$

Therefore, the condition of Theorem 4 holds with sufficiently large $u$.

As the same way, we can prove that our $(1,n)$-ME algorithm does not work if $\gamma_1 < \beta'$ with some $\beta' > \beta^2$. That is, Proposition 3 is proved. □

Our $(1, n)$-ME algorithm is superior to all known attacks with $\gamma_1 > \beta(1 - \sqrt[n]{1-\beta})$. Our $n$-variate algorithm can achieve some cryptanalytic improvements that can not be done by Sarkar's algorithm. In the full version, we show two examples, cryptanalysis of the Multi-Prime $\Phi$-Hiding Assumption if the RSA moduli have a number of prime factors, and improved fault attacks on RSA signatures with randomized paddings.

## 6 Implementation

We implemented our algorithms in Sage 5.3 over Mac OS X 10.7.5 CPU 3.06GHz Intel Core 2 Duo, 12 GB RAM. Table 2 shows the results. RSA moduli $N$ are 1024 bits and prime factors $p, q$ are 512 bits. Though our algorithms are heuristic, it works well in practice. We have new polynomials from LLL-reduced bases. We can easily recover small roots by Gröbner basis computation.

**Table 2.** Implementation of $(1, 2)$-ME algorithm

| | $\gamma_1$ | $\gamma_2$ | $d$ | LLL running time(sec.) |
|---|---|---|---|---|
| $(2,1)$-ME | 0.390625 | 0.246094 | 59 | 112.5 |
| $(2,1)$-ME | 0.488281 | 0.239258 | 100 | 332.6 |
| $(1,2)$-ME | 0.159180 | 0.007813 | 144 | 281.2 |
| $(1,2)$-ME | 0.144531 | 0.015625 | 159 | 232.7 |
| $(1,2)$-ME | 0.133789 | 0.023438 | 177 | 271.8 |

**Acknowledgement.** This work was supported by JSPS KAKENHI Grant Numbers 22700006 and 25280001.

## References

1. Bellare, M., Rogaway, P.: Probabilistic signature scheme. US Patent 6266771 (2001)
2. Boneh, D., Durfee, G.: Cryptanalysis of RSA with private key $d$ less than $N^{0.292}$. IEEE Trans. Inf. Theory 46(4), 1339–1349 (2000); Firstly appeared In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 1–11. Springer, Heidelberg (1999)
3. Chen, Y., Nguyen, P.Q.: Faster Algorithms for Approximate Common Divisors: Breaking Fully-Homomorphic-Encryption Challenges over the Integers. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 502–519. Springer, Heidelberg (2012)
4. Cohn, H., Heninger, N.: Approximate common divisors via lattices. Report 2011/437 in the Cryptology ePrint Archive (2011), `http://eprint.iacr.org/2011/437` (to appear at Proc. of ANTS-X)
5. 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)
6. Coppersmith, D.: Small solutions to polynomial equations, and low exponent RSA vulnerabilities. Journal of Cryptology 10(4), 233–260 (1997)
7. 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)

8. Coron, J.-S., Lepoint, T., Tibouchi, M.: Batch Fully Homomorphic Encryption over the Integers. Report 2013/036 in the Cryptology ePrint Archive (2013), `http://eprint.iacr.org/2013/036`
9. Coron, J.-S., Mandal, A., Naccache, D., Tibouchi, M.: Fully Homomorphic Encryption over the Integers with Shorter Public Keys. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 487–504. Springer, Heidelberg (2011)
10. Coron, J.-S., Naccache, D., Tibouchi, M.: Fault Attacks Against EMV Signatures. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 208–220. Springer, Heidelberg (2010)
11. Coron, J.-S., Naccache, D., Tibouchi, M.: Public Key Compression and Modulus Switching for Fully Homomorphic Encryption over the Integers. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 446–464. Springer, Heidelberg (2012)
12. van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 24–43. Springer, Heidelberg (2010)
13. Fouque, P.-A., Guillermin, N., Leresteux, D., Tibouchi, M., Zapalowicz, J.-C.: Attacking RSA-CRT Signatures with Faults on Montgomery Multiplication. In: Prouff, E., Schaumont, P. (eds.) CHES 2012. LNCS, vol. 7428, pp. 447–462. Springer, Heidelberg (2012)
14. Herrmann, M.: Improved Cryptanalysis of the Multi-Prime $\Phi$-Hiding Assumption. In: Nitaj, A., Pointcheval, D. (eds.) AFRICACRYPT 2011. LNCS, vol. 6737, pp. 92–99. Springer, Heidelberg (2011)
15. 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)
16. Howgrave-Graham, N.: Finding small roots of univariate modular equations revisited. In: Darnell, M. (ed.) Cryptography and Coding 1997. LNCS, vol. 1355, pp. 131–142. Springer, Heidelberg (1997)
17. Howgrave-Graham, N.: Approximate integer common divisors. In: Silverman, J.H. (ed.) CaLC 2001. LNCS, vol. 2146, pp. 51–66. Springer, Heidelberg (2001)
18. 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)
19. Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261, 515–534 (1982)
20. May, A.: New RSA Vulnerabilities Using Lattice Reduction Methods. PhD thesis, University of Paderborn (2003)
21. May, A.: Using LLL-reduction for solving RSA and factorization problems: A survey (2007), `http://www.cits.rub.de/permonen/may.html`
22. May, A., Ritzenhofen, M.: Implicit Factoring: On Polynomial Time Factoring Given Only an Implicit Hint. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 1–14. Springer, Heidelberg (2009)
23. Sarkar, S.: Reduction in Lossiness of RSA Trapdoor Permutation. In: Bogdanov, A., Sanadhya, S. (eds.) SPACE 2012. LNCS, vol. 7644, pp. 144–152. Springer, Heidelberg (2012)
24. Sarkar, S., Maitra, S.: Approximate Integer Common Divisor Problem relates to Implicit Factorization. IEEE Trans. Inf. Theory 57(4), 4002–4013 (2011)
25. Tosu, K., Kunihiro, N.: Optimal Bounds for Multi-Prime $\Phi$-Hiding Assumption. In: Susilo, W., Mu, Y., Seberry, J. (eds.) ACISP 2012. LNCS, vol. 7372, pp. 1–14. Springer, Heidelberg (2012)

# Key-Dependent Message Chosen-Ciphertext Security of the Cramer-Shoup Cryptosystem

Baodong Qin[1,2], Shengli Liu[1], and Zhengan Huang[1]

[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
[2] College of Computer Science and Technology, Southwest University of Science and Technology, Mianyang, China
{qinbaodong,slliu}@sjtu.edu.cn, zhahuang.sjtu@gmail.com

**Abstract.** The Key-Dependent Message (KDM) security requires that an encryption scheme remains secure, even if an adversary has access to encryptions of messages that depend on the secret key. In a multi-user surrounding, a key-dependent message can be any polynomial-time function $f(sk_1, sk_2, \ldots, sk_n)$ in the secret keys of the users. The Key-Dependent Message Chosen-Ciphertext (KDM-CCA2) security can be similarly defined if the adversary is also allowed to query a decryption oracle. To date, KDM security has been obtained by a few constructions. But most of them are limited $f(sk_1, sk_2, \ldots, sk_n)$ to affine functions. As to KDM-CCA2 security, there are only two constructions available. However, neither of them has comparable key sizes and reasonable efficiency, compared to the traditional KDM-free but CCA2 secure public key encryption schemes. This article defines a new function ensemble, and shows how to obtain KDM-CCA2 security with respect to this new ensemble from the traditional Cramer-Shoup (CS) cryptosystem. To obtain KDM security, the CS system has to be tailored for encryption of key-dependent messages. We present an efficient instantiation of the Cramer-Shoup public-key encryption (CS-PKE) scheme over the subgroup of quadratic residues in $\mathbb{Z}_p^*$, where $p$ is a safe prime, and prove the CS-PKE to be KDM-CCA2 secure with respect to the new function ensemble. We show that our proposed ensemble covers some affine functions, as well as other functions that are not contained in the affine ensemble. At the same time, the CS-PKE scheme with respect to our proposed function ensemble finds immediate application to anonymous credential systems. Compared to other KDM-CCA2 secure proposals, the CS scheme is the most practical one due to its short ciphertext size and computational efficiency.

**Keywords:** Key-dependent message security, adaptive chosen-ciphertext attack, Cramer-Shoup cryptosystem, public-key encryption.

## 1 Introduction

**KDM Security.** The well-accepted security notion for Public-Key Encryption (PKE) schemes [18, 28, 29] assumes that plaintext messages are chosen in a way independent of the secret key. Over the years, however, it was observed that in some situations [12, 8, 1] the messages do depend on the secret key. Security in this more demanding setting was formalized as Key-Dependent Message (KDM) security by Black et al. [6] in 2002 and circular security by Camenisch and Lysyanskaya [12] a little earlier. Some negative results [13] showed that KDM security does not follow from standard security. For this reason, in the past few years, KDM security has received much attention in various settings, including the symmetric-key [24, 3], public-key [8, 9], identity-based settings [2, 17]. For detailed history, results and applications of KDM security, we refer to the excellent survey by Malkin, Teranishi and Yung [27].

In this paper, we are interested in public-key encryption with KDM security. We will first recall the model of KDM security for public-key setting proposed by Black et al. [6] and then give an overview of the main results on KDM secure PKE schemes.

In the setting of public-key encryption, KDM security with respect to an efficiently computable (polynomial-time) function ensemble $\mathcal{F}$ is modelled as follows. An adversary is given $n$ public keys $(pk_i)_{i=1}^n$ and has access to an encryption oracle $\mathcal{O}_{\mathrm{kdm}}$ that returns the encryption of $f((sk_i)_{i=1}^n)$ under public key $pk_i$ for any function $f \in \mathcal{F}$ and any index $1 \leq i \leq n$ chosen by the adversary. The scheme is $\mathrm{KDM}^{(n)}[\mathcal{F}]$ secure if the adversary can not tell any difference when $\mathcal{O}_{\mathrm{kdm}}$ is replaced by an oracle that always returns an encryption of some constant symbol 0. However, there is no *efficient* constructions achieving KDM security when $f$ is polynomial-time but arbitrarily depends on the secret key. With the exception of [26] (which is KEM-secure with respect to the so-called Straight-Line Program (SLP) computable functions), the best result up to now is that the function ensemble $\mathcal{F}$ is limited to affine functions. In other words, the adversary has access only to affine functions of the secret keys. Nevertheless, KDM security with respect to affine functions covers the *circular security.* Circular security is defined in a multi-user setting when the adversary is provided with encryptions of arbitrary secret keys under arbitrary public key. And it is sufficient in scenarios like harddisk encryption, anonymous credential systems, etc.

Note that the function ensemble $\mathcal{F}$ plays an important role in KDM security. Different function ensembles may find applications in different scenarios. For example, when $\mathcal{F}$ is the set of all (trivial) constant functions $\mathcal{F} : \{f_m : (sk_j)_{j=1}^n \to m\}_{m\in\mathcal{M}}$ where $\mathcal{M}$ is the message space, then KDM security with respect to this ensemble just defines the traditional semantic security [18]. Another example of function ensemble is the set of all selector functions $\mathcal{F} : \{f_i : (sk_j)_{j=1}^n \to sk_i\}_{i\in[n]}$. KDM security with respect to this ensemble is called clique/circular security [8]. Circular security has the so-called "all-or-nothing" sharing property and can be used in an anonymous credential system [12] where a circular secure encryption is used to discourage delegation of credentials.

In 2008, Boneh, Halevi, Hamburg, and Ostrovsky [8] (henceforth BHHO) constructed the first PKE scheme that is KDM secure with respect to affine functions against Chosen-Plaintext Attacks (KDM-CPA secure) in the standard model (not relying on the random oracle idealization [5]), under the Decisional Diffie-Hellman (DDH) assumption. Though the BHHO scheme is quite inefficient, its approach has led to constructions of comparatively efficient KDM-CPA secure PKE schemes with respect to affine functions [9], rational functions over SLP [26] and division function [25] from other computational assumptions. In 2009, Camenisch, Chandran and Shoup [11] proposed the first PKE scheme that is KDM secure against adaptive Chosen-Ciphertext Attacks (KDM-CCA2 secure) in the standard model. They showed that by applying the Naor-Yung "double encryption" paradigm, one can combine any KDM-CPA secure scheme to any regular CCA2 secure scheme, along with a non-interactive zero-knowledge proof, to obtain a KDM-CCA2 secure scheme. Moreover, they gave a nearly practical instantiation using the BBHO KDM-CPA scheme, a $K$-linear version [30, 23] of the Cramer-Shoup CCA2 scheme [14], and a pairing-based NIZK proof system [19–21]. Very recently, Hofheinz [22] constructed a KDM-CCA2 secure PKE scheme with respect to selector functions. It is the first KDM-CCA2 secure scheme with compact ciphertexts. In this scheme, a new technical tool called Lossy Algebraic Filters (LAFs) was used to make a hybrid between the BHHO-like PKE schemes of Brakerski and Goldwasser [9], resp. Malkin et al. [26] to achieve KDM-CCA2 security. So far, only the constructions of Camenisch et al. [11] and Hofheinz [22] are known to be KDM-CCA2 secure. However, none of them are competitive with current KDM-free schemes in terms of parameters and efficiency.

Theoretically, it would be interesting to find a feasible KDM secure PKE scheme with respect to a function ensemble $\mathcal{F}$ that is as large as possible, regardless of its efficiency [10, 4]. But, in this article, we concentrates on designing *practical* KDM secure scheme with respect to a reasonably large $\mathcal{F}$.

## 1.1 Our Contribution

In this work, we prove the KDM-CCA2 security of the Cramer-Shoup cryptosystem by designing a reasonable function ensemble and show its application to credential systems.

- Let $\mathcal{K}$ be the secret key space and $q$ be a prime number of $\kappa$ (security parameter) bits. We define a function ensemble $\mathcal{F} = \{f : \mathcal{K}^n \rightarrow \mathbb{Z}_q\}$, where

$$f(sk_1, \ldots, sk_n) = \sum_{t=1}^{L} \prod_{i,j \in [n]} \alpha_{i,j,t}(sk_i - sk_j)^{a_{i,j,t}} \pmod{q}. \tag{1}$$

  Here $n, L \in \mathbb{N}, \alpha_{i,j,t} \in \mathbb{Z}_q$. When $\mathcal{K}$ is not contained in $\mathbb{Z}_q$, we assume that any secret key can be mapped into a vector space over $\mathbb{F}_q$ via an injective function. Then all the secret keys in Eq.(1) are replaced by their vector components.
- Our function ensemble covers the *difference circular encryptions*

$$CE = (\mathsf{Enc}(pk_1, sk_1 - sk_2), \mathsf{Enc}(pk_2, sk_2 - sk_3), \ldots, \mathsf{Enc}(pk_n, sk_n - sk_1)),$$

which provide the "all-or-nothing" property, i.e., anyone who knows one secret key can obtain all the secret keys from $CE$. Consequently, our result finds an immediate application to anonymous credential systems [12].

– We proved the Cramer-Shoup ($\mathsf{CS}$) scheme to be KDM-CCA2 secure with respect to the aforementioned function ensemble with an efficient instantiation. To encrypt the secret key, we will first find an injective function to map the secret key to elements of group $\mathbb{G}$ on which the $\mathsf{CS}$ scheme is based. Specifically, the instantiation is based on the subgroup $\mathbb{QR}_p$ of quadratic residues in $\mathbb{Z}_p^*$, where $p = 2q + 1$ is a safe prime. Obviously, $\mathbb{QR}_p$ is a cyclic group with prime order $q$. The DDH problem over $\mathbb{QR}_p$ is believed to be intractable from [7]. As noted in [15], in this group, elements of $\mathbb{Z}_q$ are easily encoded as elements of $\mathbb{QR}_p$. Thus, the secret key space (i.e. $\mathbb{Z}_q$) can be used as a message space. This algorithm guarantees our scheme the same efficiency as the original KDM-free Cramer-Shoup scheme.

### 1.2 Organization

The rest of this paper is organized as follows. Preliminaries are presented in Section 2. We give a detailed definition for a new function ensemble, and an application of the KDM-secure scheme with respect to this function ensemble in Section 3. We propose an instantiation of the Cramer-Shoup encryption scheme, and show its KDM-CCA2 security with respect to the new function ensemble in Section 4. Conclusion and further work are presented in Section 5.

## 2 Preliminaries

**Notation.** Let $\kappa \in \mathbb{N}$ denote the security parameter. For $n \in \mathbb{N}$, let $[n] = \{1, 2, \ldots, n\}$. For $i \in [n]$, let $[n \setminus i] = \{1, 2, \ldots, n\} \setminus \{i\}$. If $A$ is a set, then $|A|$ denotes its cardinality and $a \overset{\$}{\leftarrow} A$ denotes that $a$ is sampled uniformly from $A$. Let $|a|$ denote the bit-length of $a$. If $A$ is a probabilistic algorithm, then $a \overset{\$}{\leftarrow} A(x; R)$ denotes that $a$ is computed by $A$ on input $x$ and randomness $R$. We write $a \overset{\$}{\leftarrow} A(x)$ to denote $a \overset{\$}{\leftarrow} A(x; R)$ with uniformly chosen $R$.

### 2.1 Definitions of KDM-CCA2 Security

**Public-Key Encryption.** A public-key encryption scheme $\mathsf{PKE}$ consists of a tuple of (possibly probabilistic) polynomial-time algorithms $(\mathsf{Sys}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. The system parameter generation algorithm $\mathsf{Sys}(1^\kappa)$ takes as input a security parameter and outputs a public parameter $\textsc{pp}$. The public/secret key generation algorithm $\mathsf{Gen}(\textsc{pp})$ takes as input a public parameter $\textsc{pp}$ and outputs a public/secret key pair $(pk, sk)$. The probabilistic encryption algorithm $\mathsf{Enc}(\textsc{pp}, pk, m)$ encrypts a message $m$ from the message space $\mathcal{M}$ under public parameter $\textsc{pp}$ and public key $pk$ and outputs a ciphetext $c$. The deterministic decryption algorithm $\mathsf{Dec}(\textsc{pp}, sk, c)$ decrypts a ciphertext $c$ with secret key $sk$ and outputs a message $m$ or a special symbol $\perp$. For consistency, we require that for all $\textsc{pp} \xleftarrow{\$} \mathsf{Sys}(1^\kappa)$, all $(pk, sk) \xleftarrow{\$} \mathsf{Gen}(\textsc{pp})$, all $m \in \mathcal{M}$ and all $c \xleftarrow{\$} \mathsf{Enc}(\textsc{pp}, pk, m)$, we have $\mathsf{Dec}(\textsc{pp}, sk, c) = m$.

**Security.** We recall the definition of KDM-CCA2 security proposed by Camenisch et al. [11]. Let $\mathcal{K}$ be the space of secret keys output by $\mathsf{Gen}$. For $n \in \mathbb{N}$, let $\mathcal{F} := \{f : \mathcal{K}^n \to \mathcal{M}\}$ be a finite set of functions. KDM-CCA2 security with respect to function ensemble $\mathcal{F}$ is defined through the following experiment played between a challenger and an adversary $\mathcal{A}$. The experiment $\mathsf{Exp}_{\mathsf{PKE},\mathcal{A}}^{\text{KDM-CCA2}}(\kappa, b)$ (where $b = 0, 1$) is defined as

**Experiment** $\mathsf{Exp}_{\mathsf{PKE},\mathcal{A}}^{\text{KDM-CCA2}}(\kappa, b)$:

1. **Initialization Phase:** The challenger runs $\mathsf{Sys}(1^\kappa)$ to generate a public parameter $\textsc{pp}$ and then runs $\mathsf{Gen}(\textsc{pp})$ $n$ times to generate $n$ key pairs $(pk_i, sk_i)$, $i \in [n]$. It sends $\textsc{pp}$ and the $n$ public keys $pk_i$, $i \in [n]$ to $\mathcal{A}$. The challenger also initializes a list $\mathsf{CL} := \emptyset$ to an empty list. Suppose that $0^{|\mathcal{M}|}$ is a fixed message in $\mathcal{M}$.
2. **Query Phase:** $\mathcal{A}$ may adaptively query the challenger for two types of operations.
   (a) **Encryption Queries:** The adversary selects $(i, f) \in [n] \times \mathcal{F}$ and submits it to the challenger. The challenger computes $c = \mathsf{Enc}(\textsc{pp}, pk_i, m)$, where $m$ depends on the value of $b$. If $b = 0$, then $m = f(sk_1, \ldots, sk_n)$, else $m = 0^{|\mathcal{M}|}$. Then it appends $(i, c)$ to $\mathsf{CL}$. Finally, the challenger sends $c$ to the adversary.
   (b) **Decryption Queries:** The adversary submits a ciphertext $c$ together with an index $i \in [n]$ to the challenger. If $(i, c) \in \mathsf{CL}$, the challenger returns $\perp$, otherwise returns the output of $\mathsf{Dec}(\textsc{pp}, sk_i, c)$.
3. **Guess Phase:** Eventually, the adversary outputs a bit $b' \in \{0, 1\}$.

**Definition 1 (KDM-CCA2).** A public-key encryption scheme $\mathsf{PKE}$ is KDM-CCA2 secure with respect to $\mathcal{F}$ if for any probabilistic polynomial-time adversary $\mathcal{A}$, the following advantage:

$$\mathsf{Adv}_{\mathsf{PKE},\mathcal{A}}^{\text{KDM-CCA2}}(\kappa) = \Pr\left[\mathsf{Exp}_{\mathsf{PKE},\mathcal{A}}^{\text{KDM-CCA2}}(\kappa, 0) = 1\right] - \Pr\left[\mathsf{Exp}_{\mathsf{PKE},\mathcal{A}}^{\text{KDM-CCA2}}(\kappa, 1) = 1\right]$$

is negligible in $\kappa$.

Note that in the above experiment, if we let $n = 1$ and let $\mathcal{F}$ be the set of all constant functions on $\mathcal{K} \to \mathcal{M}$, then we obtain the standard notion of security against adaptive Chosen-Ciphertext Attacks (CCA2) [29].

## 2.2 Target-Collision Resistant Hashing

A hash function $\mathsf{H} : X \to Y$ is Target Collision-Resistant (TCR) if given a random pre-image $x \in X$, it is hard to find $x' \neq x$ with $\mathsf{H}(x') = \mathsf{H}(x)$. Below is the precise definition of TCR.

**Definition 2 (TCR hash function).** Let $\mathsf{H} : X \to Y$ be a hash function. Then $\mathsf{H}$ is target collision-resistant if for an PPT algorithm $\mathcal{B}$ the following advantage

$$\mathsf{Adv}^{\mathrm{TCR}}_{\mathsf{H},\mathcal{B}}(\kappa) := \Pr\left[x' \neq x \wedge \mathsf{H}(x') = \mathsf{H}(x) \mid x \xleftarrow{\$} X, x' \xleftarrow{\$} \mathcal{B}(x)\right]$$

is negligible in $\kappa$.

### 2.3 The DDH Assumption

Let $\mathbb{G}$ be a group of prime order $q$, and $g$ be a generator. The Decisional Diffie-Hellman (DDH) problem over the group $\mathbb{G}$ is to distinguish the two distributions $(g, g^a, g^b, g^{ab})$ and $(g, g^a, g^b, g^c)$ where $a, b, c$ are uniformly at random from $\mathbb{Z}_q$.

**Definition 3 (The DDH Assumption).** The Decisional Diffie-Hellman (DDH) assumption over group $\mathbb{G}$ states that for every PPT algorithm $\mathcal{D}$,

$$\mathrm{Adv}^{\mathrm{ddh}}_{\mathbb{G},\mathcal{D}}(\kappa) := \left|\Pr\left[\mathcal{D}(g, g^a, g^b, g^{ab}) = 1\right] - \Pr\left[\mathcal{D}(g, g^a, g^b, g^c) = 1\right]\right|$$

is negligible, where $g$ is a generator of $\mathbb{G}$ and $a, b, c \xleftarrow{\$} \mathbb{Z}_q$.

Let $p$ be a safe prime with $p = 2q + 1$, where $q$ is also a prime. Let $\mathbb{QR}_p$ denote the subgroup of quadratic residues in $\mathbb{Z}_p^*$. Then $\mathbb{QR}_p$ is a cyclic group of prime order $q$. The DDH problem over the subgroup $\mathbb{QR}_p$ is believed to be intractable (for details, see [7]).

## 3 New Function Ensemble and Application

### 3.1 New Function Ensemble

Let $L$ and $M$ be two positive integers, both of which are polynomial in $\kappa$. Let $q$ be a prime number. Let $\mathcal{X}$ be a finite set contained in $\mathbb{Z}_q$. We define a new ensemble of functions $\mathcal{F}^{L,M}_{q,n} : \mathcal{X}^n \to \mathbb{Z}_q$ over ring $\mathbb{Z}_q$. The functions in $\mathcal{F}^{L,M}_{q,n}$ have $n$ variables $X = (x_1, \ldots, x_n)$ with coefficients in ring $\mathbb{Z}_q$ and each function $f \in \mathcal{F}^{L,M}_{q,n}$ can be expressed as

$$f(x_1, \ldots, x_n) = \sum_{t=1}^{L} \prod_{i,j \in [n]} \alpha_{i,j,t}(x_i - x_j)^{a_{i,j,t}} \pmod{q},$$

where $\alpha_{i,j,t} \in \mathbb{Z}_q$ and $\sum_{i,j \in [n]} a_{i,j,t} \leq M$ for all $t$. Actually, function $f$ can also be considered as a multivariate polynomial with $\{x_i - x_j\}_{i,j \in [n]}$ being the $n^2$ arguments. In formula, Define $\mathcal{S}(n^2)$ be the set of $n^2$-variate polynomials over $\mathbb{Z}_q$. Then

$$\mathcal{F}^{L,M}_{q,n} = \{g : (x_i - x_j)_{i,j \in [n]} \to \mathbb{Z}_q \mid g \in \mathcal{S}(n^2), x_i \in \mathcal{X}, i \in [n]\}.$$

It is easy to see that $(x_1 - x_2), (x_2 - x_3), \ldots, (x_{n-1} - x_n), (x_n - x_1)$ are all contained in $\mathcal{F}^{L,M}_{q,n}$. We have

$$\begin{pmatrix} x_1 - x_2 \\ x_2 - x_3 \\ \vdots \\ x_{n-1} - x_n \\ x_n - x_1 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & -1 & 0 & \ldots 0 & 0 \\ 0 & 1 & -1 & \ldots 0 & 0 \\ \vdots & \vdots & \vdots & \ldots \vdots & \vdots \\ 0 & 0 & 0 & \ldots 1 & -1 \\ -1 & 0 & 0 & \ldots 0 & 1 \end{pmatrix}}_{A} \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}.$$

The matrix $A_{n\times n}$ has rank $n$. For any vector $\boldsymbol{b} = (b_1, b_2, \ldots, b_n)$ over $\mathbb{Z}_q$ and $\beta \in \mathbb{Z}_q$, the following

$$f(x_1, x_2, \ldots, x_n) = (b_1, b_2, \ldots, b_n) \cdot A \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} + \beta$$

is an affine function which is in $\mathcal{F}_{q,n}^{L,M}$.

Let $\mathcal{I} = \{\boldsymbol{b} \cdot A \cdot (x_1, x_2, \ldots, x_n)^T + \beta \mid \boldsymbol{b} \in \mathbb{Z}_q^n, \beta \in \mathbb{Z}_q\}$. Let $\varGamma$ be the ensemble of all affine functions from $\mathcal{X}^n$ to $\mathbb{Z}_q$. Then we have $\mathcal{I} = \mathcal{F}_{q,n}^{L,M} \cap \varGamma$. Therefore, the new function ensemble covers part of affine functions. On the other hand, when the degree of the multivariate polynomial $g$ is higher than 1, the new ensemble covers much more functions that are not contained in the affine function ensemble.

As for KDM security, set $\mathcal{X}$ is just the secret key space $\mathcal{K}$, and the function ensemble is expressed as $\mathcal{F}_{q,n}^{L,M} = \{g : (sk_i - sk_j)_{i,j\in[n]} \to \mathbb{Z}_q \mid g \in \mathcal{S}(n^2), sk_i \in \mathcal{K}, i \in [n]\}$. To evaluate the function, we implicitly assume that $\mathcal{K} \subseteq \mathbb{Z}_q$.

If $\mathcal{K} \nsubseteq \mathbb{Z}_q$, we assume that there is an efficient injective function to map $\mathcal{K}$ into a vector space over $\mathbb{Z}_q$. All the $sk_i$ will be replaced by any of its vector components over $\mathbb{Z}_q$.

## 3.2 Application to Anonymous Credential Systems

In 2001, Camenisch and Lysyanskaya [12] proposed a practical anonymous credential system. In the system, to prevent users from sharing their credentials, they introduced the primitive of circular encryption to achieve all-or-nothing sharing: a user who allows a friend to use one of her credentials once, gives this friend the ability to use all of her credentials. Circular encryption is a special kind of key-dependent message encryption. Specifically, a circular encryption for $n$ users consists of a tuple of ciphertexts:

$$\mathsf{Enc}(\textsc{pp}, pk_1, sk_2), \mathsf{Enc}(\textsc{pp}, pk_2, sk_3), \ldots, \mathsf{Enc}(\textsc{pp}, pk_n, sk_1).$$

Obviously, a friend who knows one secret key can obtain all of the remaining secret keys from these ciphertexts.

We briefly show that a KDM-secure PKE scheme with respect to functions $f \in \mathcal{F}_{q,n}^{L,M}$ suffices to achieve the so-called "all-or-nothing" property. Consider the following KDM ciphertexts

$$\mathsf{Enc}(\text{PP}, pk_1, sk_1 - sk_2), \mathsf{Enc}(\text{PP}, pk_2, sk_2 - sk_3), \ldots, \mathsf{Enc}(\text{PP}, pk_n, sk_n - sk_1).$$

It is clear that the above encryptions also have "all-or-nothing" sharing property. Thus, it can be used to discourage delegation of credentials in an anonymous credential system as well. We will see shortly (in Section 4) that such encryption can be constructed efficiently and enjoys KDM-CCA2-security in the standard model.

## 4 The CS Scheme Tailored for KDM-CCA2 Security

### 4.1 The Tailored CS Scheme

Recall that the traditional CS scheme has a message space $\mathbb{G}$ of order $q$, while the secret key space is $\mathbb{Z}_q^6$. To encrypt a message that depends on $sk$, it is necessary to tailor the CS scheme to encode $sk = (x_1, x_2, x_3, x_4, x_5, x_6)$ into six elements over $\mathbb{G}$. More precisely, we need an efficient injective encoding $\mathsf{encode} : \mathbb{Z}_q \to \mathbb{G}$ and efficient decoding $\mathsf{decode} : \mathbb{G} \to \mathbb{Z}_q$ such that $\mathsf{decode}(\mathsf{encode}(x)) = x$ for all $x \in \mathbb{Z}_q$.

Now we describe the tailored Cramer-Shoup encryption (denoted by $\mathsf{CS} = (\mathsf{CS.Sys}, \mathsf{CS.Gen}, \mathsf{CS.Enc}, \mathsf{CS.Dec})$ over the group $\mathbb{G}$ equipped with two more algorithm ($\mathsf{encode}, \mathsf{decode}$). Note now that the message space is $\mathbb{Z}_q$, compared to $\mathbb{G}$ in the original CSscheme.

$\mathsf{CS.Sys}(1^\kappa)$: Choose a group $\mathbb{G}$ of prime order $q$, where $q$ has $\kappa$ bits. Choose two random generators $g, \hat{g}$ of $\mathbb{G}$. In addition, pick a TCR hash function $\mathsf{H} : \mathbb{G}^3 \to \mathbb{Z}_q$. The public parameter is $\text{PP} = (q, \mathbb{G}, g, \hat{g}, \mathsf{H})$.

$\mathsf{CS.Gen}(\text{PP})$: Randomly and independently choose $x_i$ from $\mathbb{Z}_q$ for $i \in [6]$ and set

$$h_1 = g^{x_1}\hat{g}^{x_2}, \qquad h_2 = g^{x_3}\hat{g}^{x_4}, \qquad h_3 = g^{x_5}\hat{g}^{x_6}$$

The public and secret keys are $pk = (h_i)_{i\in[3]}$ resp. $sk = (x_i)_{i\in[6]}$

$\mathsf{CS.Enc}(\text{PP}, pk, m)$: To encrypt a message $m \in \mathbb{Z}_q$ with the public parameter $\text{PP} = (q, \mathbb{G}, g, \hat{g}, \mathsf{H})$ and the public key $pk = (h_1, h_2, h_3)$, choose a random $r \leftarrow \mathbb{Z}_q$ and then set

$$u = g^r, \qquad \hat{u} = \hat{g}^r, \qquad c = h_1^r \cdot \mathsf{encode}(m), \qquad v = (h_2^t h_3)^r$$

where $t = \mathsf{H}(u, \hat{u}, c)$. The ciphertext is $C = (u, \hat{u}, c, v)$.

$\mathsf{CS.Dec}(\text{PP}, sk, C)$: To decrypt a ciphertext $C = (u, \hat{u}, c, v)$ with the secret key $sk = (x_i)_{i=1}^6$, compute $t = \mathsf{H}(u, \hat{u}, c)$ and check that $u^{x_3 t + x_5} \cdot \hat{u}^{x_4 t + x_6} \overset{?}{=} v$ holds. If it does not, outputs $\perp$, else outputs $\mathsf{decode}(c/u^{x_1}\hat{u}^{x_2})$.

**Correctness.** If a ciphertext $C = (u, \hat{u}, c, v)$ is the output of $\mathsf{CS.Enc}(\textsc{pp}, pk, m)$ with a public key $pk$ generated by $\mathsf{CS.Gen}(\textsc{pp})$, then we will have

$$\begin{cases} u^{x_3 t + x_5} \cdot \hat{u}^{x_4 t + x_6} = (g^{x_3}\hat{g}^{x_4})^{tr} \cdot (g^{x_5}\hat{g}^{x_6})^r = (h_2^t h_3)^r = v \\ \mathsf{decode}\left(\frac{c}{u^{x_1}\hat{u}^{x_2}}\right) = \mathsf{decode}\left(\frac{h_1^r \cdot \mathsf{encode}(m)}{h_1^r}\right) = \mathsf{decode}(\mathsf{encode}(m)) = m \end{cases}$$

as required.

Now we present an instantiation of the above $\mathsf{CS}$ scheme with specific $\mathsf{encode}$ and $\mathsf{decode}$ algorithms, which were also suggested in [15].

**Group Generation:** $\mathsf{CS.Sys}(1^\kappa)$: Choose a safe prime $p = 2q + 1$. Let $\mathbb{QR}_p$ be the subgroup of quadratic residues in $\mathbb{Z}_p^*$. Choose two random generators $g, \hat{g}$ of $\mathbb{QR}_p$. In addition, pick a TCR hash function $\mathsf{H} : \mathbb{QR}_p^3 \to \mathbb{Z}_q$. The public parameter is $\textsc{pp} = (p, q, \mathbb{QR}_p, g, \hat{g}, \mathsf{H})$.

**Encode:** Let $\left(\frac{x}{p}\right)$ denote the Legendre symbol. The algorithm $\mathsf{encode} : \mathbb{Z}_q \to \mathbb{QR}_p$ is given by

$$\mathsf{encode}(x) := \begin{cases} x, & \text{if } \left(\frac{x}{p}\right) = 1 \\ p - x, & \text{if } \left(\frac{x}{p}\right) = -1 \end{cases}$$

**Decode:** The decoding algorithm $\mathsf{decode} : \mathbb{QR}_p \to \mathbb{Z}_q$ is described as follows:

$$\mathsf{decode}(x) := \begin{cases} x, & \text{if } 1 \le x \le q \\ p - x, & \text{if } q < x \le p - 1 \end{cases}$$

Note that, for each $x \in \mathbb{Z}_q$, if $\left(\frac{x}{p}\right) = 1$, then $x \in \mathbb{QR}_p$; if $\left(\frac{x}{p}\right) = -1$, then $\left(\frac{p-x}{p}\right) = 1$ and thus $p - x \in \mathbb{QR}_p$. Further, for distinct $x_1, x_2 \in \mathbb{Z}_q$, we have $\mathsf{encode}(x_1) \neq \mathsf{encode}(x_2)$. Thus, $\mathsf{encode}$ is a bijective mapping and $\mathsf{decode}$ is its inverse.

**CCA2 Security.** According to [15], we have the following theorem.

**Theorem 4.** *The above PKE encryption scheme is CCA2-secure if* $\mathsf{H}$ *is a TCR hash function and the DDH assumption holds in the subgroup* $\mathbb{QR}_p$. *More precisely, we have*

$$\mathsf{Adv}_{\mathsf{CS},\mathcal{A}}^{\mathrm{CCA2}}(\kappa) \le 2 \cdot \left(\mathsf{Adv}_{\mathbb{QR}_p,\mathcal{D}}^{\mathrm{DDH}}(\kappa) + \mathsf{Adv}_{\mathsf{H},\mathcal{B}}^{\mathrm{TCR}}(\kappa) + \frac{(Q_d + 4)}{q}\right)$$

*where* $Q_d$ *is the number of decryption queries* $\mathcal{A}$ *makes to its decryption oracle.*

## 4.2 KDM-CCA2 Security of the Tailored CS Scheme

**Secret Keys as Messages.** The tailored $\mathsf{CS}$ scheme has secret keys $sk = (x_s)_{s=1}^6 \in \mathbb{Z}_q^6$. In an $n$-user setting, let $sk_i = (x_{i,1}, x_{i,2}, x_{i,3}, x_{i,4}, x_{i,5}, x_{i,6})$ for $i = 1, 2, \ldots, n$. The function in ensemble $\mathcal{F}_{q,n}^{L,M}$ is defined by

$$f(sk_1, \ldots, sk_n) = f((x_{i,1}, x_{i,2}, x_{i,3}, x_{i,4}, x_{i,5}, x_{i,6})_{i=1}^n) \tag{2}$$
$$= \sum_{t=1}^{L} \prod_{i,j \in [n], s \in [6]} \alpha_{i,j,t}(x_{i,s} - x_{j,s})^{a_{i,j,t}} \pmod q, \tag{3}$$

where $\alpha_{i,j,t} \in \mathbb{Z}_q$, and $a_{i,j,t} \in \mathbb{N}$.

Next, we show that the CS-PKE scheme proposed in Section 4.1 achieves KDM-CCA2 security with respect to the function class $\mathcal{F}_{q,n}^{L,M}$. We have the following theorem.

**Theorem 5.** *For any positive integer $n \geq 2$, $L$ and $M$, both of which are polynomial in the security parameter $\kappa$, the tailored* CS *scheme is KDM-CCA2 secure with respect to the function class $\mathcal{F}_{q,n}^{L,M}$. More precisely, we have*

$$\mathsf{Adv}_{\mathsf{CS},\mathcal{A}}^{\text{KDM-CCA2}}(\kappa) \leq 2nQ_e \cdot \left( \mathsf{Adv}_{\mathbb{QR}_p,\mathcal{D}}^{\text{DDH}}(\kappa) + \mathsf{Adv}_{\mathsf{H},\mathcal{B}}^{\text{TCR}}(\kappa) + \frac{(Q_d + 4)}{q} \right)$$

*assuming that $\mathcal{A}$ makes at most $Q_e$ queries to the encryption oracle and makes at most $Q_d$ queries to the decryption oracle.*

Before the formal proof, we briefly explain why our scheme can achieve KDM-CCA2 security with respect to the aforementioned function ensemble. Let us first take a look at the structure of our scheme. The public key is given by $pk = (h_i)_{i=1}^3 = (g^{x_{2i-1}} \hat{g}^{x_{2i}})_{i=1}^3$ and the secret key is given by $sk = (x_i)_{i=1}^6$, where $g, \hat{g}$ are two random generators and $(x_i)_{i=1}^6$ are independently and uniformly chosen from $\mathbb{Z}_q$. To encrypt a message $m \in \mathbb{Z}_q$ under public key $pk$, we select a random $r \in \mathbb{Z}_q$ and then set the ciphertext as $C = (u, \hat{u}, c, v) = (g^r, \hat{g}^r, h_1^r \cdot \mathsf{encode}(m), h_2^{rt} h_3^r)$, where $t = \mathsf{H}(u, \hat{u}, c)$ and $\mathsf{H}$ is a suitable hash function. In the proof, the simulator starts from a public key $pk^*$ with respect to secret key $sk^*$. She chooses difference values $\widehat{sk_i}$ uniformly and independently from $\mathbb{Z}_q^6$, and then all of the other secret keys are implicitly defined based on the chosen differences from the unknown secret key. The public key $pk_i$ corresponding to the $i$-th secret key can also be correctly computed from $pk^*$ and the difference values $\widehat{sk_i}$. Though the simulator does not know $sk^*$, she can correctly compute the difference between $sk_i$ and $sk_j$ for all $i, j \in [n]$. Thus, the simulator can answer all the KDM queries perfectly. What remains is to answer decryption queries. Suppose that the queried ciphertext $C = (u, \hat{u}, c, v)$ is an encryption of message $m$ under public key $pk_i$. The simulator can prepare a new ciphertext $C^* = (u, \hat{u}, c, v^*)$ that is an encryption of some message $m^*$ under public key $pk^*$. The message $m^*$ can be used by the simulator to recover the real message $m$. So, the simulator only needs the decryption oracle with respect to $pk^*$ to get $m^*$ and then recover message $m$. The above analysis suggests that the CCA2 security of the CS scheme implies the KDM-CCA2 security of our proposal.

*Proof.* For any PPT adversary $\mathcal{A}$ attacking on CS's KDM-CCA2 security, Let $Q_e$ denote the number of queries to the encryption oracle and $Q_d$ the number of queries to the decryption oracle. The proof proceeds through a sequence of games. Denote by $\mathsf{out}_i$ the output of $\mathcal{A}$ in $\mathsf{Game}_i$.

$\mathsf{Game}_0$ is the original KDM-CCA2 security experiment for $b = 0$. Thus,

$$\Pr[\mathsf{out}_0 = 1] := \Pr\left[\mathsf{Exp}_{\mathsf{CS},\mathcal{A}}^{\text{KDM-CCA2}}(\kappa, 0) = 1\right]$$

$\mathsf{Game}_1$ is the same as $\mathsf{Game}_0$ except for the way the challenger answers the encryption queries of $\mathcal{A}$. The challenger first computes and stores in a list $\mathcal{L}$ all the differences between any pair of the $n$ secret keys.

$$\mathcal{L} = \{(i, j, sk_i - sk_j)_{i,j\in[n]} = (i, j, x_{i,1} - x_{j,1}, x_{i,2} - x_{j,2}, \ldots, x_{i,6} - x_{j,6})_{i,j\in[n]}\},$$

where all operations are over $\mathbb{F}_q$.
For each encryption query $(i_k, f_k)$, $k \in [Q_e]$, the challenger computes the key-dependent message

$$f_k(sk_1, \ldots, sk_n) = \sum_{t=1}^{L} \prod_{i,j\in[n],s\in[6]} \alpha_{i,j,t}(x_{i,s} - x_{j,s})^{a_{i,j,t}} \pmod q,$$

with the values $(x_{i,s} - x_{j,s})$ stored in $\mathcal{L}$, instead of the secret keys $sk_1, \ldots,$ $sk_n$. Then the challenger computes

$$C_k = \mathsf{CS.Enc}(\textsc{pp}, pk_{i_k}, f_k(sk_1, \ldots, sk_n))$$

and returns $C_k$ to $\mathcal{A}$.
This change does not affect $\mathcal{A}$'s view in $\mathsf{Game}_0$ at all. Thus,

$$\Pr[\mathsf{out}_1 = 1] = \Pr[\mathsf{out}_0 = 1].$$

$\mathsf{Game}_2$ is defined as the KDM-CCA2 security experiment for $b = 1$, i.e., the challenger answers all the responses of the encryption queries with encryptions of $0^{|\mathcal{M}|}$.
To show $\mathsf{Game}_1$ and $\mathsf{Game}_2$ are indistinguishable, it needs to prove that the following sub-games $\mathsf{Game}_{1.\ell}$ $(\ell = 0, 1, \ldots, Q_e)$ are indistinguishable.
Denote by $(i_k, f_k)$ the $k$-th encryption query by $\mathcal{A}$, $k = 1, 2, \ldots, Q_e$.
In $\mathsf{Game}_{1.\ell}$ $(\ell = 0, 1, \ldots, Q_e)$, the challenger responds the $k$-th encryption query $(i_k, f_k)$ with

$$C_k = \begin{cases} \mathsf{Enc}(\textsc{pp}, pk_{i_k}, 0^{|\mathcal{M}|}) & \text{if } \; k = 1, 2, \ldots, \ell \\ \mathsf{Enc}(\textsc{pp}, pk_{i_k}, f_k((sk_j)_{j\in[n]})) & \text{if } \; k = \ell + 1, \ell + 2, \ldots, Q_e \end{cases}$$

It is easy to see that $\mathsf{Game}_{1.0}$ is the same as $\mathsf{Game}_1$, $\mathsf{Game}_{1.\ell}$ is the same as $\mathsf{Game}_2$, and

$$|\Pr[\mathsf{out}_2 = 1] - \Pr[\mathsf{out}_1 = 1]| \leq \sum_{\ell=1}^{Q_e} |\Pr[\mathsf{out}_{1.\ell} = 1] - \Pr[\mathsf{out}_{1.\ell-1} = 1]|.$$

We postpone the proof of the following claim.

**Lemma 6.** *For all* $1 \leq \ell \leq Q_e$,

$$\Pr[\mathsf{out}_{1.\ell} = 1] - \Pr[\mathsf{out}_{1.\ell-1} = 1] \leq n\mathsf{Adv}^{\mathrm{CCA2}}_{\mathsf{CS},\mathcal{A}'}(\kappa)$$

*for a suitable adversary* $\mathcal{A}'$ *on* $\mathsf{CS}$*'s CCA2-security.*

Then, it follows that

$$\Pr[\mathsf{out}_1 = 1] - \Pr[\mathsf{out}_2 = 1] \leq nQ_e\mathsf{Adv}^{\mathrm{CCA2}}_{\mathsf{CS},\mathcal{A}'}(\kappa).$$

Finally, combining all the three games, we have

$$\mathsf{Adv}^{\mathrm{KDM\text{-}CCA2}}_{\mathsf{CS},\mathcal{A}}(\kappa) \leq nQ_e\mathsf{Adv}^{\mathrm{CCA2}}_{\mathsf{CS},\mathcal{A}'}(\kappa).$$

□

We now turn to prove Lemma 6.

*Proof (proof of Lemma 6).* We construct a PPT adversary $\mathcal{A}'$ who implements an attack on $\mathsf{CS}$'s CCA2-security using $\mathcal{A}$ as a subroutine. Remember in the IND-CCA2 game, $\mathcal{A}'$ is given public parameters $\textsc{pp} = (q, \mathbb{G}, g, \hat{g}, \mathsf{H})$ which is the output of $\mathsf{CS.Sys}(1^\kappa)$, a public key $pk^* = (h_1^*, h_2^*, h_3^*)$, which is generated via $(pk^*, sk^*) \leftarrow \mathsf{CS.Gen}(\mathsf{pp})$, and access to a decryption oracle $\mathcal{O}'_{sk^*}(\cdot)$ which answers $\mathcal{A}'$'s decryption query $C$ with $\mathsf{CS.Dec}(\textsc{pp}, sk^*, C)$.

In order to simulate $\mathsf{Game}_{1.\ell}$ (or $\mathsf{Game}_{1.\ell-1}$) for $\mathcal{A}$, $\mathcal{A}'$ prepares the public parameters and public keys, the encryption and decryption oracles for $\mathcal{A}$ as follows.

**Simulating Public Parameters and Public Keys:** Given public parameters $\textsc{pp} = (q, \mathbb{G}, g, \hat{g}, \mathsf{H})$ and the public key $pk^* = (h_1^*, h_2^*, h_3^*)$, $\mathcal{A}'$ computes $(pk_1, pk_2, \ldots, pk_n)$ as follows.

- Choose $i^* \xleftarrow{\$} [n]$;
- Choose $(x_{i,1}, x_{i,2}, x_{i,3}, x_{i,4}, x_{i,5}, x_{i,6}) \xleftarrow{\$} \mathbb{Z}_q^6$ for $i \in [n \setminus i^*]$;
- Compute

$$h_{i,1} = h_1^* \cdot g^{x_{i,1}}\hat{g}^{x_{i,2}}, \quad h_{i,2} = h_2^* \cdot g^{x_{i,3}}\hat{g}^{x_{i,4}}, \quad h_{i,3} = h_3^* \cdot g^{x_{i,5}}\hat{g}^{x_{i,6}}, \quad (4)$$

  for $i \in [n \setminus i^*]$;
- Set $h_{i^*,1} = h_1^*, \qquad h_{i^*,2} = h_2^*, \qquad h_{i^*,3} = h_3^*$;
- Set $pk_i = (h_{i,1}, h_{i,2}, h_{i,3})$ for $i \in [n]$.

Return $\textsc{pp} = (q, \mathbb{G}, g, \hat{g}, \mathsf{H})$ and $(pk_1, pk_2, \ldots, pk_n)$ to $\mathcal{A}$.

All the public keys are distributed exactly like the original KDM-CCA2 security game.

Note that $\mathcal{A}'$ does not know the values of the secret keys $(sk_1, sk_2, \ldots, sk_n)$ at all, but it does know all the differences between any pair of secret keys

$$sk_i - sk_j = \begin{cases} (x_{i,1} - x_{j,1}, x_{i,2} - x_{j,2}, \ldots, x_{i,6} - x_{j,6}) & \text{if } \ i, j \neq i^* \\ (x_{i,1}, x_{i,2}, \ldots, x_{i,6}) & \text{if } \ i \neq i^*, j = i^* \\ (-x_{j,1}, -x_{j,2}, \ldots, -x_{j,6}) & \text{if } \ i = i^*, j \neq i^* \end{cases} \quad (5)$$

for $i, j \in [n]$, and stores all the secret key differences in a list

$$\mathcal{L} = \{(i, j, sk_i - sk_j)_{i,j \in [n]}\}.$$

**Simulating Encryption Oracle:** There are totally $Q_e$ queries from the adversary. Let $(i_k, f_k)$ be the $k$-th query from $\mathcal{A}$.

– For $k = 1, \ldots, \ell - 1$, $\mathcal{A}'$ responds the query $(i_k, f_k)$ with

$$C_k = \mathsf{Enc}(\text{PP}, pk_{i_k}, 0^{|\mathcal{M}|}).$$

– For $k = \ell$, $\mathcal{A}'$ checks whether $i_\ell = i^*$. If $i_\ell \neq i^*$, $\mathcal{A}'$ aborts the game with $\perp$; Otherwise, $\mathcal{A}'$ submits $m_0 = 0^{|\mathcal{M}|}$ and $m_1 = f_\ell(sk_1, \ldots, sk_n) \pmod q$ to its own encryption oracle and gets the challenge ciphertext $C^*$. $\mathcal{A}'$ sets

$$C_k = C^*.$$

– For $k = \ell + 1, \ell + 2, \ldots, Q_e$, $\mathcal{A}$ responds the query $(i_k, f_k)$ with

$$C_k = \mathsf{Enc}(\text{PP}, pk_{i_k}, f_k((sk_j)_{j \in [n]})).$$

Since $\mathcal{A}'$ knows all the differences $sk_i - sk_j$ between any secret key pair, $\mathcal{A}'$ is able to compute $C_k$ without knowing secret keys.

Finally, $\mathcal{A}'$ stores $(i_1, C_1), (i_2, C_2), \ldots, (i_{Q_e}, C_{Q_e})$ in the ciphertext list $\mathsf{CL}$.

**Simulating Decryption Oracle:** Given the decryption oracle $\mathcal{O}'_{sk^*}(\cdot)$ to which $\mathcal{A}'$ has access in the IND-CCA2 security game, $\mathcal{A}'$ is able to provide decryption services to $\mathcal{A}$ with help of $\mathcal{O}'_{sk^*}(\cdot)$ .

Denote by $(i, C)$ the decryption query submitted by $\mathcal{A}$, then $\mathcal{A}'$ proceeds as follows.

1. If $(i, C) \in \mathsf{CL}$, then return $\perp$ to $\mathcal{A}$.
2. If $(i, C) \notin \mathsf{CL}$ and $i = i^*$, then $\mathcal{A}'$ submits $C$ to its own decryption oracle $\mathcal{O}'_{sk^*}(\cdot)$ and returns the output of $\mathcal{O}'_{sk^*}(\cdot)$ to $\mathcal{A}$.
3. If $(i, C) \notin \mathsf{CL}$ and $i \neq i^*$, parse $C$ as $(u, \hat{u}, c, v)$ and then compute

$$t = \mathsf{H}(u, \hat{u}, c) \text{ and } v' = \frac{v}{u^{x_{i,3}t + x_{i,5}} \cdot \hat{u}^{x_{i,4}t + x_{i,6}}}.$$

Next, $\mathcal{A}'$ submits the ciphertext $C' = (u, \hat{u}, c, v')$ to $\mathcal{O}'_{sk^*}(\cdot)$. Let $m'$ be the output of its decryption oracle $\mathcal{O}'_{sk^*}(\cdot)$. If $m' = \perp$, then $\mathcal{A}'$ returns $\perp$ to $\mathcal{A}$, otherwise $\mathcal{A}'$ computes

$$m = \mathsf{decode}\left(\frac{\mathsf{encode}(m')}{u^{x_{i,1}} \cdot \hat{u}^{x_{i,2}}}\right)$$

and returns this message (i.e. $m$) to $\mathcal{A}$.

The correctness of the decryption services provided by $\mathcal{A}'$ is justified by the the correctness of $\mathcal{O}'_{sk^*}(\cdot)$ and the following facts.

Suppose $sk^* = (x_1^*, x_2^*, x_3^*, x_4^*, x_5^*, x_6^*)$ be the corresponding secret key of $pk^* = (h_1^*, h_2^*, h_3^*)$. For $i \neq i^*$, according to Eq.(4),

$$pk_i = (h_{i,1}, h_{i,2}, h_{i,3}) = (h_1^* \cdot g^{x_{i,1}} \hat{g}^{x_{i,2}}, \quad h_2^* \cdot g^{x_{i,3}} \hat{g}^{x_{i,4}}, \quad h_3^* \cdot g^{x_{i,5}} \hat{g}^{x_{i,6}}),$$

and the secret key is

$$sk_i = (x_1^* + x_{i,1}, \quad x_2^* + x_{i,2}, \quad x_3^* + x_{i,3}, \quad x_4^* + x_{i,4}, \quad x_5^* + x_{i,5}, \quad x_6^* + x_{i,6}).$$

The ciphertext $C' = (u, \hat{u}, c, v')$ is consistent under $pk^* = (h_1^*, h_2^*, h_3^*)$ if

$$v' = u^{x_3^* t + x_5^*} \cdot \hat{u}^{x_4^* t + x_6^*}.$$

Meanwhile the ciphertext $C = (u, \hat{u}, c, v)$ is consistent under $pk_i = (h_{i,1}, h_{i,2}, h_{i,3})$ if

$$v = u^{(x_3^* + x_{i,3})t + (x_5^* + x_{i,5})} \cdot \hat{u}^{(x_4^* + x_{i,4})t + (x_6^* + x_{i,6})}.$$

Note that $v = v' \cdot u^{x_{i,3}t + x_{i,5}} \cdot \hat{u}^{x_{i,4}t + x_{i,6}}$, so $C' = (u, \hat{u}, c, v')$ is consistent under $pk^*$ iff $C = (u, \hat{u}, c, v)$ is consistent under $pk_i$.
For consistent ciphertext $C' = (u, \hat{u}, c, v')$ under $pk^*$, $\mathcal{O}'_{sk^*}(\cdot)$ returns $m' = \mathsf{decode}\left(\frac{c}{u^{x_1^*} \hat{u}^{x_2^*}}\right)$. For consistent ciphertext $C = (u, \hat{u}, c, v)$ under $pk_i$, the decryption result should be $m = \mathsf{decode}\left(\frac{c}{u^{x_1^* + x_{i,1}} \hat{u}^{x_2^* + x_{i,2}}}\right)$. Note that $m = \mathsf{decode}\left(\frac{\mathsf{encode}(m')}{u^{x_{i,1}} \cdot \hat{u}^{x_{i,2}}}\right)$.

As long as $\mathcal{A}'$ does not abort in the simulation, $\mathcal{A}'$ perfectly simulates $\mathsf{Game}_{1.\ell}$ if $C^*$ is an encryption of message $m_0 = 0^{|\mathcal{M}|}$ and $\mathsf{Game}_{1.\ell-1}$ if $C^*$ is an encryption of message $m_1 = f_\ell(sk_1, \dots, sk_n) \pmod q$.

Finally, $\mathcal{A}'$ outputs whatever $\mathcal{A}$ outputs.

Since $\Pr[\mathcal{A}'$ does not abort$] = \Pr[i_\ell = i^*] = 1/n$, we have

$$\Pr[\mathsf{out}_{1.\ell} = 1] - \Pr[\mathsf{out}_{1.\ell-1} = 1] \le n \mathsf{Adv}_{\mathsf{CS},\mathcal{A}'}^{\mathrm{CCA2}}(\kappa).$$

This completes the proof of Lemma 6. □

## 5 Conclusion and Further Work

This paper introduced a new function ensemble and showed the relation between this new ensemble with the ensemble of affine functions. A tailored Cramer-Shoup (CS) cryptosystem was proved to be key-dependent message chosen-ciphertext secure with respect to the new function ensemble. We also presented an efficient instantiation of the tailored CS cryptosystem. Though the new function ensemble does not cover all the affine functions over $\mathbb{Z}_q$, it suffices for some applications like the anonymous credential systems. Due to the efficiency of the CS system, our proposal is much more efficient than the known KDM-CCA2 secure schemes [11, 22]. Our result shows the elegance of the (CS) cryptosystem and promote the applications of KDM secure PKE in practice.

It may be of independent interest to exploit other applications of the new functions ensemble. Another interesting challenge is design *practical* scheme, as efficient as the CS system, with KDM-CCA2 security with respect to affine functions.

**Acknowledgements.** We are grateful to the anonymous ACISP 2013 reviewers for many helpful comments. This work was supported by the National Natural Science Foundation of China (Grant No. 61170229 and No. 61133014), the Specialized Research Fund for the Doctoral Program of Higher Education (Grant No. 20110073110016), and the Innovation Program of Shanghai Municipal Education Commission (Grant No. 12ZZ021).

## References

1. Adão, P., Bana, G., Herzog, J., Scedrov, A.: Soundness of formal encryption in the presence of key-cycles. In: De Capitani di Vimercati, S., Syverson, P., Gollmann, D. (eds.) ESORICS 2005. LNCS, vol. 3679, pp. 374–396. Springer, Heidelberg (2005)
2. Alperin-Sheriff, J., Peikert, C.: Circular and KDM security for identity-based encryption. In: Fischlin, et al. (eds.) [16], pp. 334–352
3. Backes, M., Pfitzmann, B., Scedrov, A.: Key-dependent message security under active attacks - BRSIM/UC-soundness of dolev-yao-style encryption with key cycles. Journal of Computer Security 16(5), 497–530 (2008)
4. Barak, B., Haitner, I., Hofheinz, D., Ishai, Y.: Bounded key-dependent message security. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 423–444. Springer, Heidelberg (2010)
5. 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.) ACM Conference on Computer and Communications Security, pp. 62–73. ACM (1993)
6. Black, J., Rogaway, P., Shrimpton, T.: Encryption-scheme security in the presence of key-dependent messages. In: Nyberg, K., Heys, H.M. (eds.) SAC 2002. LNCS, vol. 2595, pp. 62–75. Springer, Heidelberg (2003)
7. Boneh, D.: The decision Diffie-Hellman problem. In: Buhler, J.P. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 48–63. Springer, Heidelberg (1998)
8. Boneh, D., Halevi, S., Hamburg, M., Ostrovsky, R.: Circular-secure encryption from decision Diffie-Hellman. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 108–125. Springer, Heidelberg (2008)
9. Brakerski, Z., Goldwasser, S.: Circular and leakage resilient public-key encryption under subgroup indistinguishability - (or: Quadratic residuosity strikes back). In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 1–20. Springer, Heidelberg (2010)
10. Brakerski, Z., Goldwasser, S., Kalai, Y.T.: Black-box circular-secure encryption beyond affine functions. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 201–218. Springer, Heidelberg (2011)
11. Camenisch, J., Chandran, N., Shoup, V.: A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 351–368. Springer, Heidelberg (2009)
12. Camenisch, J., Lysyanskaya, A.: An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer, Heidelberg (2001)
13. Cash, D., Green, M., Hohenberger, S.: New definitions and separations for circular security. In: Fischlin, et al. (eds.) [16], pp. 540–557

14. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998)
15. 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 (2004), `http://dx.doi.org/10.1137/S0097539702403773`
16. Fischlin, M., Buchmann, J., Manulis, M. (eds.): PKC 2012. LNCS, vol. 7293. Springer, Heidelberg (2012)
17. Galindo, D., Herranz, J., Villar, J.: Identity-based encryption with master key-dependent message security and leakage-resilience. In: Foresti, S., Yung, M., Martinelli, F. (eds.) ESORICS 2012. LNCS, vol. 7459, pp. 627–642. Springer, Heidelberg (2012)
18. Goldwasser, S., Micali, S.: Probabilistic encryption. J. Comput. Syst. Sci. 28(2), 270–299 (1984)
19. Groth, J.: Simulation-sound NIZK proofs for a practical language and constant size group signatures. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 444–459. Springer, Heidelberg (2006)
20. Groth, J., Ostrovsky, R., Sahai, A.: Perfect non-interactive zero knowledge for NP. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 339–358. Springer, Heidelberg (2006)
21. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart (ed.) [31], pp. 415–432
22. Hofheinz, D.: Circular chosen-ciphertext security with compact ciphertexts. Cryptology ePrint Archive, Report 2012/150 (2012), `http://eprint.iacr.org/` (to appear, Eurocrypt 2013)
23. Hofheinz, D., Kiltz, E.: Secure hybrid encryption from weakened key encapsulation. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 553–571. Springer, Heidelberg (2007)
24. Hofheinz, D., Unruh, D.: Towards key-dependent message security in the standard model. In: Smart (ed.) [31], pp. 108–126
25. Lu, X., Li, B., Mei, Q., Xu, H.: Key-dependent message security for division function: Discouraging anonymous credential sharing. In: Boyen, X., Chen, X. (eds.) ProvSec 2011. LNCS, vol. 6980, pp. 297–308. Springer, Heidelberg (2011)
26. Malkin, T., Teranishi, I., Yung, M.: Efficient circuit-size independent public key encryption with KDM security. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 507–526. Springer, Heidelberg (2011)
27. Malkin, T., Teranishi, I., Yung, M.: Key dependent message security: recent results and applications. In: Sandhu, R.S., Bertino, E. (eds.) CODASPY, pp. 3–12. ACM (2011)
28. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: Ortiz, H. (ed.) STOC, pp. 427–437. ACM (1990)
29. 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)
30. Shacham, H.: A cramer-shoup encryption scheme from the linear assumption and from progressively weaker linear variants. Cryptology ePrint Archive, Report 2007/074 (2007), `http://eprint.iacr.org/`
31. Smart, N.P. (ed.): EUROCRYPT 2008. LNCS, vol. 4965. Springer, Heidelberg (2008)

# Black-Box Separations and Their Adaptability to the Non-uniform Model

Ahto Buldas[1,2,3] and Margus Niitsoo[4,⋆]

[1] Cybernetica AS, Mäealuse 2/1, 12618 Tallinn, Estonia
[2] Tallinn University of Technology, Raja 15, 12618 Tallinn, Estonia
[3] Guardtime AS, Tammsaare tee 60, 11316 Tallinn, Estonia
[4] University of Tartu, Liivi 2, 50409 Tartu, Estonia

**Abstract.** Oracle separation methods are used in cryptography to rule out black-box reductions between cryptographic primitives. It is sufficient to find an oracle relative to which the base primitive exists but there are no secure instances of the constructed primitive. It is often beyond our current reach to construct a fixed oracle with such properties because it is difficult to prove the existence of secure base primitives. To overcome this gap, randomized oracles are used to create random base primitives that are secure on average. After that, a fixed oracle is extracted from the probability distribution by using non-constructive probabilistic arguments and the countability of the set of adversaries. Such extraction only applies to uniform reductions because the set of non-uniform adversaries is not countable. We study how to adapt oracle separation results to the non-uniform model. The known separation techniques are capable of ruling out the so-called fully black-box reductions and a certain strong form of semi black-box reductions also in the non-uniform model. We study how to go beyond the barrier of strong semi black-box reductions and show that this is possible by using random oracles with auxiliary advice. For that end, we prove a conjecture of Unruh (2007) about pre-sampling being a sufficient substitute for advice for any oracle distribution.

## 1 Introduction

Complex cryptographic protocols are often built from simpler building blocks called primitives. Usually, the security of such protocols is proved based solely on the security guarantees of the original primitives independent of their actual implementation details. Such constructions are called *black-box reductions*. To date, almost all security proofs for efficient cryptographic constructions utilize black-box reductions.

Although back-box reductions are extremely useful cryptographic tools, there exist limits on where they can be applied. There are many known cases for which it is proved that such a reduction cannot exist. This usually means that a very clever proof construction is necessary if the reduction can be achieved at all. As very few of these clever constructions are known, the power and limits of black-box reductions are of a rather special interest to many people working in the field.

⋆ This research was supported by the European Regional Development Fund through the Estonian Center of Excellence in Computer Science, EXCS, and by Estonian Research Council's personal research grant PUT-2.

The first separation result involving black-box reductions was given in 1989 by Impagliazzo and Rudich [7]. They showed that there are no such reductions from key agreement protocols to one-way functions. Their seminal paper was followed by a long line of similar types of results [5,6,10]. The approach was even generalized by Kim, Simon and Tetali [8] to give bounds on reduction efficiency. Though all these results leave open the existence of more specific security proofs they are still valuable hardness indicators as they rule out the most obvious approaches.

Non-existence of black-box reductions is commonly shown by oracle separation techniques. In complexity theory, oracle separation has been widely applied to prove limits of the proof techniques capable of establishing set-theoretical inclusions between complexity classes. For example, with oracle separation one can easily show that diagonal arguments are insufficient to solve the famous $\mathbf{P}$ vs $\mathbf{NP}$ question. This is done by showing two oracles—one relative to which the conjecture holds and another for which it does not. In cryptography, oracle separation is used to argue that it is impossible to securely implement a primitive or a protocol $\mathcal{P}$ given only black-box access to a secure low-level primitive $f$ as an instance of a class of primitives $\mathcal{Q}$. This is done by defining an oracle so that $f$ remains secure even if the adversary can access the oracle but any instance $\mathcal{P}^f$ of the protocol $\mathcal{P}$ being considered is insecure relative to the oracle.

In classical separation results of complexity theory, oracles are defined as fixed functions that always behave the same way. In cryptographic separations, it is often hard to explicitly define a fixed separation oracle. For example, if one wishes that one way functions exist relative to the oracle, an explicit functional description of such function should then be given (as a part of the oracle). This is however still unreachable for the state of the art computer science—the existence of one way functions is conjectured but not yet proved. So, in cryptographic separations we often assume that oracles are chosen randomly from certain probability distributions. We then prove that the separation statements hold *on average* and then argue that there exists a particular choice of the oracle for which the statements hold. For example, one-way functions indeed exist in the *random oracle model* because random functions are *one-way on average* [7].

It would then seem natural that the oracle separation results would also be stated with respect to the random oracles. However, as the classical separation theorems are adopted from the classical model, the authors still try to make their oracle choice deterministic. Such an *oracle extraction* approach, though, has a big limitation—it usually requires that the number of adversaries is countable, and hence the whole approach is usable only in the *uniform* model, where the adversaries are ordinary Turing machines.

To avoid the countability argument, Buldas, Laur and Niitsoo [2] proposed an alternative oracle extraction approach where the oracle extraction step is unnecessary. Rather than trying to extract a suitable deterministic oracle from a probability distribution, they assumed that there exists a black-box reduction (that works for every $f$) and derived a contradiction by assuming the probabilistic separation condition and the average (over $f$) version of the reduction condition. We call the method they introduced the *averaging approach*. They proved that the averaging approach is capable of showing that there are no semi black-box reductions between two primitives. However, they were able to do this only for a strong version of semi black-box reduction where the simulator $A$ does not depend on the instance $f$ of the source primitive.

In this paper, we give an overview on both the traditional oracle extraction based separation and the averaging-based separation techniques. For each type of the reduction, we outline the main steps of the separation and point out the steps where the countability assumption is used. Thereby, we determine the exact reason why the separation fails in the non-uniform model. We achieved the following results:

– The traditional oracle extraction approach still works for the strong semi black-box reductions, because the separation oracle can be chosen for a fixed adversary $A$ and a fixed simulator $S$. Similar to the averaging approach, this is not so for the weak semi black-box reductions, and from this viewpoint, the averaging approach has no advantage over the traditional methods.
– We derive sufficient average-based separation criterions for the weak semi black-box and the variable semi black-box reductions. It turns out that proving the security condition of the oracle separation for the weak semi black-box reduction is equivalent of proving the security of a cryptographic construction in a model where the adversary is given a function $\varphi(\mathcal{O})$ of the oracle.
– We generalize the results of Unruh [11] about oracles with auxiliary strings so that they would apply to arbitrary oracle distributions.

## 2 Notation

By $x \leftarrow \mathcal{D}$ we mean that $x$ is chosen randomly according to a distribution $\mathcal{D}$. We use the Big Oh notation for describing asymptotic properties of functions. In particular, $f(k) = O(1)$ means that $f$ is bounded and $f(k) = k^{-\omega(1)}$ means that $f(k)$ decreases faster than any polynomial, i.e., $f$ is *negligible*. A Turing machine $\mathsf{M}$ is *poly-time* if it runs in time $k^{O(1)}$, where $k$ denotes the input size that is mostly referred to as the *security parameter*.

By an *oracle Turing machine* we mean an incompletely specified Turing machine $S$ that comprises calls to *oracles*. The description can be completed by defining the oracle as a function $\mathcal{O} : \{0,1\}^* \to \{0,1\}^*$. In this case, the machine is denoted by $S^{\mathcal{O}}$. The function $y \leftarrow \mathcal{O}(x)$ does not have to be computable but has a conditional running time $t(x)$, which does not necessarily reflect the actual amount of computations needed to produce $y$ from $x$. The running time of $S^{\mathcal{O}}$ comprises the conditional running time of oracle calls—each call $\mathcal{O}(x)$ takes $t(x)$ steps. We assume that all the oracles $\mathcal{O}$ are *poly-time*, that is $t(x) = \{x\}^{O(1)}$, where $\{x\}$ denotes the bit-length of $x$. Note that though the classical complexity-theoretic oracles only require a single step, this more general notion is appropriate in cryptography where oracles often model abstract adversaries with running time $t$. We say that $S$ is a *poly-time oracle machine* if $S^{\mathcal{O}}$ runs in poly-time, whenever $\mathcal{O}$ is poly-time. By a *non-uniform* poly-time oracle machine we mean an ordinary poly-time oracle machine $S$ together with a family $\mathcal{A} = \{a_k\}_{k\in\mathbb{N}}$ of (advice) bit-strings $a_k$ with length $k^{O(1)}$. For any oracle $\mathcal{O}$ and any input $x$, it is assumed that $S^{\mathcal{O}}(x)$ has access to the advice string $a_{\{x\}}$. Usually, the advice strings are omitted for simplicity, but their presence must always be assumed when $S$ is non-uniform. One of the most important facts about non-uniform machines is that there are uncountably many of them, whereas the set of ordinary Turing machines is countable.

## 3 Basic Lemmas

**Lemma 1 (Probabilistic Argument).** *Let $\mathcal{F}$ be a probability space and $\mathcal{P}$ be a predicate function. Then* $\Pr_{f \leftarrow \mathcal{F}}[\mathcal{P}(f)] > 0 \Rightarrow \exists f\colon \mathcal{P}(f)$.

**Lemma 2 (Countability Argument).** *Let $\mathcal{F}$ be a probability space and $\mathcal{P}(f, A)$ be a predicate function where $A$ varies over all poly-time Turing machines, then*

$$\underset{pol}{\forall} A\colon \Pr_{f \leftarrow \mathcal{F}}[\mathcal{P}(f, A)] = 1 \quad \Rightarrow \quad \Pr_{f \leftarrow \mathcal{F}}\left[\underset{pol}{\forall} A\colon \mathcal{P}(f, A)\right] = 1 \ .$$

*Proof.* Countable intersection of measure one sets is a measure one set. □

**Lemma 3 (Borel-Cantelli).** *Let $\{\mathsf{E}_i\}_{i \in \mathbb{N}}$ be a countable set of events and $\mathsf{E}_\infty$ be the event that infinitely many of these events happen. If* $\sum_n \Pr[\mathsf{E}_n] < \infty$ *then* $\Pr[\mathsf{E}_\infty] = 0$.

*Proof.* Indeed, let $B_k = \bigcup_{n=k}^{\infty} \mathsf{E}_k$. If $x \in \mathsf{E}_\infty$ then $x \in \cap_k B_k$, because otherwise $x$ only belongs to a finite sequence $\mathsf{E}_1, \ldots, \mathsf{E}_{k-1}$ of events. Hence, $\mathsf{E}_\infty \subseteq \cap_k \mathsf{B}_k$ and $\Pr[\mathsf{E}_\infty] \le \Pr[\cap_k \mathsf{B}_k] \le \Pr[\mathsf{B}_k]$. From $\sum_n \Pr[\mathsf{E}_n] < \infty$ it follows that for every $\epsilon > 0$ there is $k$ such that $\sum_{n=k}^{\infty} \Pr[\mathsf{E}_k] < \epsilon$. Thus, $\Pr[\mathsf{E}_\infty] \le \Pr[\mathsf{B}_k] = \Pr[\cup_{n=k}^{\infty} \mathsf{E}_k] \le \sum_{n=k}^{\infty} \Pr[\mathsf{E}_k] < \epsilon$, which implies $\Pr[\mathsf{E}_\infty] = 0$. □

**Lemma 4 (Negligible Average Argument).** *Let $\mathcal{F}$ be a distribution so that for every $f \leftarrow \mathcal{F}$ there is a real-valued function $\delta_f\colon \mathbb{N} \to [0,1]$. If $\mathbf{E}_{f \leftarrow \mathcal{F}}[\delta_f(k)] = \varepsilon(k) = k^{-\omega(1)}$, then $\delta_f(k) = k^{-\omega(1)}$ for measure one of $f$'s.*

*Proof.* As $\Pr_{f \leftarrow \mathcal{F}}\left[\delta_f(k) > k^2 \cdot \varepsilon(k)\right] \le k^{-2}$ and $\Pr_{f \leftarrow \mathcal{F}}\left[\delta_f(k) \le k^2 \cdot \varepsilon(k)\right] \ge 1 - k^{-2}$ by Markov inequality, we define $\mathsf{E}_k$ as the event that $\delta_f(k) > k^2 \cdot \varepsilon(k)$. Now we use the Borel-Cantelli lemma and $\sum_k \Pr[\mathsf{E}_k] \le \sum_k k^{-2} < \infty$ to imply

$$\Pr_{f \leftarrow \mathcal{F}}\left[\text{"}\delta_f(k) > k^2 \cdot \varepsilon(k) \text{ for infinitely many } k\text{-s"}\right] = \Pr[\mathsf{E}_\infty] = 0 \ .$$

Thus, for measure one of $f$'s: $\exists k_0 \forall k > k_0 : \delta_f(k) \le k^2 \cdot \varepsilon(k) = k^{-\omega(1)}$. □

**Lemma 5 (Overwhelming Average Argument).** *Let $\mathcal{F}$ be a distribution so that for every $f \leftarrow \mathcal{F}$ there is a function $\delta_f\colon \mathbb{N} \to [0,1]$. If $\mathbf{E}_{f \leftarrow \mathcal{F}}[\delta_f(k)] = 1 - k^{-\omega(1)}$, then $\delta_f(k) = 1 - k^{-\omega(1)}$ for measure one of $f$'s.*

*Proof.* $\mathbf{E}_{f \leftarrow \mathcal{F}}[1 - \delta_f(k)] = 1 - \mathbf{E}_{f \leftarrow \mathcal{F}}[\delta_f(k)] = 1 - (1 - k^{-\omega(1)}) = k^{-\omega(1)}$, which by Lemma 4 implies that $1 - \delta_f(k) = k^{-\omega(1)}$ for measure one of $f$'s. □

**Lemma 6.** *There exist quantities $\delta_i(k) = k^{-\omega(1)}$ for which $\mathbf{E}_i[\delta_i(k)] \ne k^{-\omega(1)}$.*

*Proof.* Let $I = \{1, 2, \ldots\}$ and $p_i = \frac{6}{\pi^2 i^2}$ for all $i \in I$. Then $\sum_{i \in I} p_i = 1$. For all $i \in I$ we define the function $\delta_i$ by $\delta_i(k) = \delta_{ik}$, where $\delta_{ik}$ is the Kronecker delta. Now we define a probability space on $\{\delta_i\}_{i \in I}$ such that $\Pr[\delta_i] = p_i$ for all $i \in I$. Note that $\delta_i(k) = k^{-\omega(1)}$ for all $i \in I$ but the average of all $\delta_i$-s is non-negligible, because $\mathbf{E}_i[\delta_i(k)] = \frac{6}{\pi^2} \cdot k^{-2} = k^{-O(1)} \ne k^{-\omega(1)}$. □

## 4 Primitives

Complex cryptographic constructions can often be decomposed into a few fundamental building blocks that are called *primitives*. One is usually interested in proving the security of constructions based solely on the properties of the underlying primitives. Reingold et al. [9] give a formal definition by considering primitives as families of functions of type $f\colon \{0,1\}^* \to \{0,1\}^*$ along with a matching between these functions and Turing machines implementing them. Indeed, for many common primitives such as one-way permutations or collision-resistant hash functions this formalization coincides with our intuition—an instance of a primitive is indeed just one function.

In some more complicated cases that actually have more than one type of functionality, it may make more sense to define a primitive as a tuple of functions. However, we can usually concatenate these functions into one single function – we just use the first few input bits to tell the function which sub-function we want to use. This means that we can still formalize the primitive as one single function, although it may be a little counter-intuitive.

A primitive is usually defined in terms of functional requirements that the instances of the primitive must satisfy before it makes sense to talk about their security. These requirements, though, are just syntactic and have nothing to do with security. For example, every permutation is an instance of the one-way permutation primitive, however, it does not have to be a secure instance.

In cryptography we also have to define the *security* of primitives. Reingold et al. [9] define security as a relation between primitives and Turing machines that possibly break them. That is, a machine either breaks the primitive or not. In this work, we use a more specific (but still sufficiently general) definition of security given in [2], where the breakage advantage is a real-valued function that also depends on the security parameter $k$ which is usually tied to the actual input (or output) lengths of the primitive:

**Definition 1 (Primitives, Adversaries and Advantage).** *A* primitive $\mathcal{P}$ *is a set of functions of type* $f\colon \{0,1\}^* \to \{0,1\}^*$. *Primitives have an* advantage function $\textsc{Adv}_k^{\mathcal{P}}(\cdot,\cdot)$, *which given as input the security parameter* $k \in \mathbb{N}$, *an instance* $f$ *of* $\mathcal{P}$, *and an oracle Turing machine* $\mathsf{A}^{\mathcal{O}}$ *(an* adversary*) returns a real number* $\textsc{Adv}_k^{\mathcal{P}}(\mathsf{A}^{\mathcal{O}}, f) \in [0,1]$ *(the* advantage *of* $\mathsf{A}^{\mathcal{O}}$*). The function* $\textsc{Adv}_k^{\mathcal{P}}(\cdot, f)$ *is extended to probabilistic Turing machines by taking the average over their randomness strings*[1]*. We say that* $\mathsf{A}^{\mathcal{O}}$ breaks *an instance* $f$ *of* $\mathcal{P}$ *if* $\textsc{Adv}_k^{\mathcal{P}}(\mathsf{A}^{\mathcal{O}}, f) \neq k^{-\omega(1)}$. *If for a fixed oracle* $\mathcal{O}$ *no probabilistic poly-time oracle Turing machine* $\mathsf{A}^{\mathcal{O}}$ *breaks* $f$ *then* $f$ *is said to be* secure relative to $\mathcal{O}$.

We emphasize that our definition says nothing about the efficiency of $f$. The function may even be non-computable, as long as the advantage that can be gained by any adversary is negligible. In practice, one needs an instantiation of a primitive that is both efficient and secure. Commonly, it is required that we can compute the function $f$ with a (uniform) poly-time Turing machine for $f$ to be called *efficient*.

[1] Each fixed randomness string gives a deterministic poly-time Turing machine for which $\textsc{Adv}_k^{\mathcal{P}}()$ is already defined.

## 5 Black-Box Reductions

Reductions capture the relations between different security notions. A primitive $\mathcal{P}$ can be reduced to a primitive $\mathcal{Q}$ if there exists a construction that given a secure instance of the primitive $\mathcal{Q}$ yields a secure instance of $\mathcal{P}$. Most common cryptographic reductions are *black-box reductions*, where an instance $f$ of a primitive $\mathcal{Q}$ is treated as an atomic object in the construction and in the security proof. In this work, we consider four sub-notions of black-box reductions: fully-black box reductions, strong and weak semi black box reductions, and variable semi black box reductions.

The first step towards classification of black-box reductions was made by Reingold, Trevisan and Vadhan [9]. They showed a complex hierarchy of 7 types of different reductions. Our classification is based on their work but leaves out some of the more general reductions and introduces one that actually arises quite often in practice. Also, we assume that the reduction construction is deterministic whereas the original hierarchy uses probabilistic Turing machines everywhere. This is necessary for reductions between deterministic primitives as the reduction cannot be allowed any randomization in that case. If we consider randomized primitives, $G$ can usually be made deterministic even when it is randomized in essence – the key idea is to use the oracle as a source of randomness. This approach was already used by Impagliazzo and Rudich [7] in the first paper about oracle separations in cryptology.

In the first three definitions, the construction of a derived primitive is independent of the actual implementation of $\mathcal{P}$, whereas the construction itself may depend on the implementation in the last definition. The reduction in question is *uniform* if all oracle machines in the corresponding definition are uniform, otherwise the reduction is *non-uniform*. We assume that the construction $G$ is always deterministic but the adversaries are allowed to be randomized.

**Definition 2 (Fully black-box reduction).** *A fully black-box reduction $\mathcal{P} \xRightarrow{\mathrm{f}} \mathcal{Q}$ is determined by two poly-time oracle machines $G$ and $S$, satisfying the next two conditions:*
(C) *If $f$ implements $\mathcal{Q}$ then $G^f$ implements $\mathcal{P}$.*
(S) *For every instance $f \in \mathcal{Q}$, if $\mathsf{A}$ breaks $G^f$ (as $\mathcal{P}$) then $S^{\mathsf{A},f}$ breaks $f$ (as $\mathcal{Q}$).*

In brief, we must to provide a universal oracle algorithm $S$ that can handle all successful adversaries to establish a fully black-box reduction. So-called *semi-black-box* reductions weaken this restriction by allowing for some specialization in the security proofs. The degree to which the specialization can go is different for different authors. We give two possible definitions, one just slightly stronger than the other.

**Definition 3 (Strong semi black-box reduction).** *A strong semi-black-box reduction $\mathcal{P} \xRightarrow{ss} \mathcal{Q}$ is a poly-time oracle machine $G$, satisfying the next two conditions:*

(C) *If $f$ correctly implements $\mathcal{Q}$ then $G^f$ correctly implements $\mathcal{P}$.*
(S) *For all poly-time oracle machines $\mathsf{A}$ there exists a poly-time oracle machine $\mathsf{B}$ such that for every instance $f \in \mathcal{Q}$ if $\mathsf{A}^f$ breaks $G^f$ then $\mathsf{B}^f$ breaks $f$.*

**Definition 4 (Weak semi black-box reduction).** *By weak semi-black-box reduction $\mathcal{P} \xRightarrow{ws} \mathcal{Q}$ we mean a poly-time oracle machine $G$, satisfying the next two conditions:*[2]

[2] This was the reduction given by Reingold et al. [9] as the semi-black-box reduction.

(C) *If $f$ correctly implements $\mathcal{Q}$ then $G^f$ correctly implements $\mathcal{P}$.*

(S) *For every instance $f \in \mathcal{Q}$ and a poly-time oracle machine* $\mathsf{A}$*, there exists a poly-time oracle machine* $\mathsf{B}$ *such that if* $\mathsf{A}^f$ *breaks* $G^f$ *then* $\mathsf{B}^f$ *breaks* $f$.

The difference between the two reduction types is very subtle. In the strong case, the construction of $\mathsf{B}$ may non-constructively depend on $\mathsf{A}$ and $G$ but it has to be universal for all $f \in \mathcal{Q}$. In the weak case, such a universal $\mathsf{B}$ might not exist – the construction of $\mathsf{B}$ may depend on $f$ as well as on $\mathsf{A}$ and $G$. This subtle difference is however extremely important, as it seems to create a theoretical boundary for at least one general separation method we show later.

In both semi-black-box reductions $G$ must be universal for all valid instances $f \in \mathcal{Q}$ and as such, specific properties of an instance $f$ cannot be used in the construction. Variable semi-black-box reductions [3] weaken even this restriction so that the constructions of both $G$ and $\mathsf{B}$ may depend on the instance $f$. However, such constructions must exist for all instances of $\mathcal{Q}$ regardless of the actual efficiency of $f$. If we restrict $f$ in the following definition to efficiently implementable instances of $\mathcal{Q}$, then we get the definition of *white-box reductions*, which is the most general type of reductions.

**Definition 5 (Variable semi black-box reduction).** *We say that there is a* variable semi-black-box reduction $\mathcal{P} \overset{v}{\Longrightarrow} \mathcal{Q}$ *iff for any correct implementation $f$ of $\mathcal{Q}$:*

(C) *there exists a poly-time oracle machine $G^f$ that correctly implements $\mathcal{P}$;*

(S) *for every instance $f \in \mathcal{Q}$ and for any poly-time oracle machine* $\mathsf{A}$*, there exists a poly-time oracle machine* $\mathsf{B}$ *such that if* $\mathsf{A}^f$ *breaks* $G^f$*, then* $\mathsf{B}^f$ *breaks* $f$.

These reduction types form a linear hierarchy with fully black-box reductions being the strongest and white-box reductions being the weakest. Existence of a reduction of one type also trivially implies the existence of reductions of all weaker types. This is important as it means that non-existence of a weaker reduction also implies non-existence of all stronger reductions.

## 6 Oracle-Extraction Based Separation

Showing the non-existence of black-box reductions of a primitive $\mathcal{P}$ to a primitive $\mathcal{Q}$ by oracle separation involves two major steps:

- (Breakage argument) Define an oracle $\mathcal{O}$ relative to which there is no secure $\mathcal{P}$;
- (Security argument) Show that there is secure $\mathcal{Q}$ relative to $\mathcal{O}$.

In cryptography, oracle separation is almost never done with an explicitly defined oracle. Instead of that, the existence of a suitable oracle is proved by using probabilistic arguments, i.e. it is shown that an oracle with the desired properties can be extracted from a probability space of oracles. So, the first step of an oracle separation is to define a probability distribution $\mathcal{O} \leftarrow \mathcal{F}$ of oracles and show that there is secure instance $f^{\mathcal{O}}$ of $\mathcal{Q}$ but no instance $G^{\mathcal{O}}$ of $\mathcal{P}$ is secure relative to $\mathcal{O}$. By using the so-called oracle embedding techniques first introduced by Simon [10], the secure instance $f$ of $\mathcal{Q}$ can be identified with the oracle $\mathcal{O}$, i.e. the oracle distribution is $f \leftarrow \mathcal{F}$. To show the security argument in the oracle-extraction based separation techniques we show that:

[3] They were called $\forall\exists$-semi-black-box by Reingold et al.

s$_1$: Every fixed poly-time adversary $S$ that uses $f$ as an oracle can break $f$ only with negligible success, on average, i.e. $\underset{\text{pol}}{\forall} S\colon \underset{f \leftarrow \mathcal{F}}{\mathbf{E}}\left[\text{ADV}_k(S^f, f)\right] = k^{-\omega(1)}$.

s$_2$: $\underset{f \leftarrow \mathcal{F}}{\Pr}\left[\text{ADV}_k(S^f, f) = k^{-\omega(1)}\right] = 1$, i.e. for measure one of $f$'s, no poly time $S$ can break $f$ (by s$_1$ and Lemma 4).

s$_3$: For measure one of oracles $f$, no poly-time $S^f$ can break $f$ better than with negligible success, i.e. $\underset{f \leftarrow \mathcal{F}}{\Pr}\left[\underset{\text{pol}}{\forall} S\colon \text{ADV}_k(S^f, f) = k^{-\omega(1)}\right] = 1$ (by s$_2$, Lemma 2).

To show the breakage argument we have to show that:

b$_1$: Every instance $G^f$ of $\mathcal{P}$ can be broken by a poly-time $A$ with overwhelming probability, i.e. $\underset{\text{pol}}{\forall} G \underset{\text{pol}}{\exists} A\colon \underset{f \leftarrow \mathcal{F}}{\mathbf{E}}\left[\text{ADV}_k(A^f, G^f)\right] = 1 - k^{-\omega(1)}$.

b$_2$: For every instance $G^f$ of $\mathcal{P}$ there is a poly-time $A$, so that $A^f$ breaks $G^f$ for measure one of $f$'s, i.e. $\underset{\text{pol}}{\forall} G \underset{\text{pol}}{\exists} A\colon \underset{f \leftarrow \mathcal{F}}{\Pr}\left[\text{ADV}_k(A^f, G^f) = 1 - k^{-\omega(1)}\right] = 1$ (b$_1$, Lemma 5).

b$_3$: For measure one of oracles $f$, every $G^f$ can be broken by a poly-time machine $A$, i.e. $\underset{f \leftarrow \mathcal{F}}{\Pr}\left[\underset{\text{pol}}{\forall} G \underset{\text{pol}}{\exists} A\colon \text{ADV}_k(A^f, G^f) = 1 - k^{-\omega(1)}\right] = 1$ (b$_2$, Lemma 2).

Finally, by combining these two sets of measure one, we have that measure one of oracles satisfy both the breakage and the security conditions, which means that by the probabilistic argument (Lemma 1) there exists a fixed separation oracle. Note that (b$_1$) cannot be replaced with the weaker statement $\underset{f \leftarrow \mathcal{F}}{\mathbf{E}}\left[\text{ADV}_k(A^f, G^f)\right] = k^{-O(1)}$, because it does not imply that there is an $f$ for which $\text{ADV}_k(A^f, G^f) = k^{-O(1)}$. There exists a counterexample (Lemma 6) for which $\text{ADV}_k(A^f, G^f) = k^{-\omega(1)}$ for all $f$.

**Table 1.** Reduction types and separation conditions for oracle extraction based separations. The quantifier $\underset{\text{of}}{\forall}$ means that the quantified variable varies over all oracle functions.

| Type | Reduction Condition | Separation Condition |
|---|---|---|
| Fully bb | $\underset{\text{of}}{\exists} p \underset{\text{pol}}{\exists} S \forall f \forall A$: $A \,\mathbf{br}\, p(f) \Rightarrow S^{A,f} \,\mathbf{br}\, f$ | $\underset{\text{of}}{\forall} p \underset{\text{pol}}{\forall} S \exists \mathcal{F}$: $\underset{f,A \leftarrow \mathcal{F}}{\mathbf{E}}[\text{ADV}_k(A, p(f))] = 1 - k^{-\omega(1)}$ $\underset{f,A \leftarrow \mathcal{F}}{\mathbf{E}}\left[\text{ADV}_k(S^{f,A}, f)\right] = k^{-\omega(1)}$ |
| Strong Semi bb | $\underset{\text{of}}{\exists} p \underset{\text{pol}}{\forall} A \underset{\text{pol}}{\exists} S \forall f$: $A^f \,\mathbf{br}\, p(f) \Rightarrow S^f \,\mathbf{br}\, f$ | $\underset{\text{of}}{\forall} p \underset{\text{pol}}{\exists} A \underset{\text{pol}}{\forall} S \exists \mathcal{F}$: $\underset{f \leftarrow \mathcal{F}}{\mathbf{E}}\left[\text{ADV}_k(A^f, p(f))\right] = 1 - k^{-\omega(1)}$ $\underset{f \leftarrow \mathcal{F}}{\mathbf{E}}\left[\text{ADV}_k(S^f, f)\right] = k^{-\omega(1)}$ |
| Weak Semi bb | $\underset{\text{of}}{\exists} p \underset{\text{pol}}{\forall} A \forall f \underset{\text{pol}}{\exists} S$: $A^f \,\mathbf{br}\, p(f) \Rightarrow S^f \,\mathbf{br}\, f$ | $\underset{\text{of}}{\forall} p \underset{\text{pol}}{\exists} A \exists \mathcal{F}$: $\underset{f \leftarrow \mathcal{F}}{\mathbf{E}}\left[\text{ADV}_k(A^f, p(f))\right] = 1 - k^{-\omega(1)}$ $\underset{\text{pol}}{\forall} S \underset{f \leftarrow \mathcal{F}}{\mathbf{E}}\left[\text{ADV}_k(S^f, f)\right] = k^{-\omega(1)}$ Countability argument for $S$ |
| Variable Semi bb | $\forall f \underset{\text{pol}}{\exists} G \underset{\text{pol}}{\forall} A \underset{\text{pol}}{\exists} S$: $A^f \,\mathbf{br}\, G^f \Rightarrow S^f \,\mathbf{br}\, f$ | $\exists \mathcal{F}$: $\underset{\text{pol}}{\forall} G \underset{\text{pol}}{\exists} A \underset{f \leftarrow \mathcal{F}}{\mathbf{E}}\left[\text{ADV}_k(A^f, G^f)\right] = 1 - k^{-\omega(1)}$ $\underset{\text{pol}}{\forall} S \underset{f \leftarrow \mathcal{F}}{\mathbf{E}}\left[\text{ADV}_k(S^f, f)\right] = k^{-\omega(1)}$ Countability arguments for $G$ and $S$ |

The steps $s_1$, $s_2$, $b_1$ and $b_2$ also apply to non-uniform reductions, whereas the steps $s_3$ and $b_3$ do not, because there are uncountably many non-uniform machines $S$ and $G$.

In order to still apply the separation technique in the non-uniform model, there are several ways to go on. The first way is to use the fact that stronger types of black-box reductions may need weaker separation arguments and perhaps there is no need to use the countability argument, i.e. the existence of the separation oracle may be deducible without extraction. Indeed, if we examine the negations of the reduction statements for all four types of black-box reductions (Table 1), we observe that the non-existence of fully black-box and the strong semi black-box reductions can be proven without the countability argument. The main reason is that the oracle distribution $\mathcal{F}$ must be defined for a particular choice of $A$ and $S$. Formally, this means that in the separation condition, the quantifier $\exists\mathcal{F}$ stands after the quantifiers $\underset{\text{pol}}{\exists} A$ and $\underset{\text{pol}}{\forall} S$ (Table 1).

It also turns out in a natural way that in order to show non-existence of fully black-box reductions, we may use two separate oracles $A$ and $f$, where the secure instance $(f)$ of $\mathcal{Q}$ only has access to $f$, while the adversary $S$ has access to both oracles. This fact was first pointed out by Hsiao and Reyzin [6]. In Table 1, we list the separation conditions for all four types of reductions. All proofs are given in Appendix A.

To conclude, the oracle extraction based separation techniques are applicable to the *strong semi black-box reductions* (and hence also for the *fully black-box reductions*) but not for the *weak semi black-box* and *variable semi black-box reductions*, because the latter would require the countability argument.

## 7 Averaging-Based Separation

In practical separations, both the security- and the breakage assumption are probabilistic, i.e. involve an average success over the oracle. The reduction condition (for fully black-box reduction) is deterministic and has the form:

$$\textsc{Adv}_k^{\mathcal{P}}(A, G^f) \neq k^{-\omega(1)} \quad \Rightarrow \quad \textsc{Adv}_k^{\mathcal{Q}}(S^{f,A}, f) \neq k^{-\omega(1)} \ . \tag{1}$$

For showing that there are no fully black-box reductions of $\mathcal{P}$ to $\mathcal{Q}$, we have to derive a contradiction based on the reduction condition (1) and the separation conditions:

$$\text{(S)} \quad \mathop{\mathbf{E}}_{f,A\leftarrow\mathcal{F}}\left[\textsc{Adv}_k^{\mathcal{P}}(A, G^f)\right] = 1 - k^{-\omega(1)}$$

$$\text{(B)} \quad \underset{\text{pol}}{\forall} S\colon \quad \mathop{\mathbf{E}}_{f,A\leftarrow\mathcal{F}}\left[\textsc{Adv}_k^{\mathcal{Q}}(S^{f,A}, f)\right] = k^{-\omega(1)} \ .$$

To derive a contradiction from (S), (B), and (1), the traditional approach focuses on conditions (S) and (B) and tries to derive a negation of (1) from these two conditions. This is done by using oracle extraction, i.e. by extracting a fixed oracles $f$ and $A$ from $\mathcal{F}$ so that (1) is not satisfied. The average-based separation technique [2] does the opposite: it first focuses on (1) and tries to derive the following averaged version:

$$\mathop{\mathbf{E}}_{f,A\leftarrow\mathcal{F}}\left[\textsc{Adv}_k^{\mathcal{P}}(A, G^f)\right] \neq k^{-\omega(1)} \quad \Rightarrow \quad \mathop{\mathbf{E}}_{f,A\leftarrow\mathcal{F}}\left[\textsc{Adv}_k^{\mathcal{Q}}(S^{f,A}, f)\right] \neq k^{-\omega(1)} \ , \tag{2}$$

and then derive a contradiction based on (S), (B) and (2). Indeed, from (S) it follows that $\mathop{\mathbf{E}}_{f,A\leftarrow\mathcal{F}}\left[\textsc{Adv}_k^{\mathcal{P}}(A,G^f)\right] = 1 - k^{-\omega(1)} \neq k^{-\omega(1)}$. By (2) we imply that $\mathop{\mathbf{E}}_{f,A\leftarrow\mathcal{F}}\left[\textsc{Adv}_k^{\mathcal{Q}}(S^{f,A},f)\right] \neq k^{-\omega(1)}$ which contradicts (B).

### 7.1 Poly-Preserving Reductions

The main problem with the the averaging approach described above is that the averaged condition (2) cannot be derived from the general reduction condition (1). Indeed, let $a_f(k) = \textsc{Adv}_k^{\mathcal{P}}(A,G^f)$ and $b_f(k) = \textsc{Adv}_k^{\mathcal{Q}}(S^{f,A},f)$. We would like to prove that if (for all $f$) $b_f(k) = k^{-\omega(1)}$ implies $a_f(k) = k^{-\omega(1)}$, then

$$\mathop{\mathbf{E}}_f[b_f(k)] = k^{-\omega(1)} \Rightarrow \mathop{\mathbf{E}}_f[a_f(k)] = k^{-\omega(1)} \ .$$

The negligible average argument (Lemma 4) implies that $a_f(k) = k^{-\omega(1)}$ for measure one of $f$'s, but this does not mean that $\mathop{\mathbf{E}}_f[a_f(k)]$ is negligible (Lemma 6).

So, for average-based separation, the guarantee condition (S) is too weak—much weaker than what is usually expected when constructing practical reductions. For these reasons, the guarantee condition was strengthened [2] and the class of reductions restricted in the following reasonable way:

**Definition 6 (Poly-preserving reductions).** *A reduction of $\mathcal{P}$ to $\mathcal{Q}$ is* poly-preserving *if the security guarantee (S) decreases the advantage by at most a polynomial amount, i.e. there exists $c \geq 1$ (independent of $f$, $A$ and $k$) such that*

$$\textsc{Adv}_k^{\mathcal{Q}}(S^f,f) \geq \left[\textsc{Adv}_k^{\mathcal{P}}(A^f,p(f))\right]^c \ . \tag{3}$$

### 7.2 Averaging-Based Separation for Poly-Preserving Reductions

For fully-black box reductions, (3) is in the form $\textsc{Adv}_k^{\mathcal{Q}}(S^{f,A},f) \geq \left[\textsc{Adv}_k^{\mathcal{P}}(A,G^f)\right]^c$. For poly-preserving reductions, the averaged reduction condition (2) easily follows:

$$\mathop{\mathbf{E}}_{f,A\leftarrow\mathcal{F}}\left[\textsc{Adv}_k^{\mathcal{Q}}(S^{f,A},f)\right] \geq \mathop{\mathbf{E}}_{f,A\leftarrow\mathcal{F}}\left[\left(\textsc{Adv}_k^{\mathcal{P}}(A,G^f)\right)^c\right] \geq \left(\mathop{\mathbf{E}}_{f,A\leftarrow\mathcal{F}}\left[\textsc{Adv}_k^{\mathcal{P}}(A,G^f)\right]\right)^c \ ,$$

where the second inequality is an application of the Jensen inequality. This implies that if $\mathop{\mathbf{E}}_{f,A\leftarrow\mathcal{F}}\left[\textsc{Adv}_k^{\mathcal{P}}(A,G^f)\right]$ is non-negligible, then so is $\mathop{\mathbf{E}}_{f,A\leftarrow\mathcal{F}}\left[\textsc{Adv}_k^{\mathcal{Q}}(S^{f,A},f)\right]$.

Table 2 lists the separation conditions for all four types of reductions. Note that the breakage condition for averaging-based separation can be somewhat weaker than in the traditional extraction-based approach. We only have to assume that the success of $A$ is non-negligible. All proofs are given in Appendix B.

**Table 2.** Reduction types and separation conditions for averaging-based separation in the case of poly-preserving black-box reductions

| Type | Reduction Condition | Separation Condition |
|---|---|---|
| Full | $\underset{\text{of}}{\exists} p \underset{\text{pol}}{\exists} S \forall f \forall A:$ | $\underset{\text{of}}{\forall} p \underset{\text{pol}}{\forall} S \exists \mathcal{F}: \underset{f,A\leftarrow\mathcal{F}}{\mathbf{E}} [\textsc{Adv}_k(A, p(f))] \neq k^{-\omega(1)}$ |
| bb | $\textsc{Adv}_k(S^{f,A}, f) \geq \left[\textsc{Adv}_k(A^f, p(f))\right]^c$ | $\underset{f,A\leftarrow\mathcal{F}}{\mathbf{E}} [\textsc{Adv}_k(S^{f,A}, f)] = k^{-\omega(1)}$ |
| Str | $\underset{\text{of}}{\exists} p \underset{\text{pol}}{\forall} A \underset{\text{pol}}{\exists} S \forall f:$ | $\underset{\text{of}}{\forall} p \underset{\text{pol}}{\exists} A \underset{\text{pol}}{\forall} S \exists \mathcal{F}: \underset{f\leftarrow\mathcal{F}}{\mathbf{E}} [\textsc{Adv}_k(A^f, p(f))] \neq k^{-\omega(1)}$ |
| s-bb | $\textsc{Adv}_k(S^f, f) \geq \left[\textsc{Adv}_k(A^f, p(f))\right]^c$ | $\underset{f\leftarrow\mathcal{F}}{\mathbf{E}} [\textsc{Adv}_k(S^f, f)] = k^{-\omega(1)}$ |
| Weak | $\underset{\text{of}}{\exists} p \underset{\text{pol}}{\forall} A \forall f \underset{\text{pol}}{\exists} S_f:$ | $\underset{\text{of}}{\forall} p \underset{\text{pol}}{\exists} A \exists \mathcal{F}: \underset{f\leftarrow\mathcal{F}}{\mathbf{E}} [\textsc{Adv}_k(A^f, p(f))] \neq k^{-\omega(1)}$ |
| s-bb | $\textsc{Adv}_k(S_f^f, f) \geq \left[\textsc{Adv}_k(A^f, p(f))\right]^c$ | $\underset{\text{pol}}{\forall} S \underset{\text{of}}{\forall} \varphi \underset{f\leftarrow\mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(S^f_{\varphi(f)}, f)\right] = k^{-\omega(1)}$ |
| Vari | $\forall f \underset{\text{pol}}{\exists} G_f \underset{\text{pol}}{\forall} A \underset{\text{pol}}{\exists} S_f:$ | $\underset{\text{of}}{\forall} \psi \exists \mathcal{F}: \underset{\text{pol}}{\forall} G \underset{\text{pol}}{\exists} A \underset{f\leftarrow\mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(A^f, G^f_{\psi(f)})\right] \neq k^{-\omega(1)}$ |
| s-bb | $\textsc{Adv}_k(S_f^f, f) \geq \left[\textsc{Adv}_k(A^f, G_f^f)\right]^c$ | $\underset{\text{pol}}{\forall} S \underset{\text{of}}{\forall} \varphi \underset{f\leftarrow\mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(S^f_{\varphi(f)}, f)\right] = k^{-\omega(1)}$ |

## 8 Going beyond the Strong Semi Black-Box Boundary

The strong Semi Black-box separations are a clear boundary for oracle extraction based approaches, as anything below that requires the use of countability arguments, which fail in the non-uniform model. It therefore seems that the best hope of proving stronger reductions would rest with the averaging-based approach. For this to succeed, however, one would have to be able to account for oracle-dependent advice strings $\varphi(\mathcal{F})$ being available to the adversary construction. This is by not a trivial obstacle to overcome.

The most promising approach for doing so stems from the work of Unruh [11], where they showed that under reasonable assumptions, the oracle could be switched out with one fairly independent from the original without the adversary having a significant chance of noticing the switch. A problem with his approach was that it only applies to standard random oracles, which separation oracles rarely are. If this idea was to be used, the result first needed to be generalized to work for other, less standard oracles as well. This turned out to be possible as we managed to prove one of the conjectures presented in the original paper [11] pertaining to the fully general choice of oracle distributions.

**Theorem 1.** *Let $\mathcal{F}$ be any distribution of Oracles and let $f \leftarrow \mathcal{F}$. We say that $f$ is consistent with a matching $M = \{x_1 \rightarrow y_1, \ldots, x_m \rightarrow y_m\}$ if $f(x_i) = y_i$ for all $i \in \{1, 2, \ldots, m\}$. Let $\varphi(f)$ be an oracle function with an output of length $p$. Then there is an oracle function $S$ such that $S^f$ is a matching of length $m$ and the following holds: For any probabilistic oracle Turing machine $B$ that makes at most $q$ queries to its oracle, $\Delta(B^{\mathcal{F}}_{\varphi(\mathcal{F})}; B^{\mathcal{F}/S}_{\varphi(\mathcal{F})}) \leq \sqrt{\frac{pq}{2m}}$, where $\mathcal{F}/S$ is an oracle sampled according to $\mathcal{F}$ conditioned only on being consistent with $S^{\mathcal{F}}$ (which is also a random variable).*

We will generalize the proof for Theorem 2 of Unruh [11] to work for arbitrary oracle distributions. The proof is quite similar to the original one, with only the notion of information $J(\mathsf{M})$ gathered by an adaptive list $\mathsf{M}$ about the advice string $z$ given a more general definition. Since most of the proofs are completely analogous, we will only give a detailed description of the parts that have to be changed. We will use the notion of adapative lists from the original paper. To reiterate, a adaptive list is a Turing Machine that takes as input a finite advice string $z$ and then proceeds to make a number of oracle queries, terminating in finite time, outputting all the oracle queries that it made along with their results. It is assumed that the query responses are cached so that oracle is never queried twice with the same input. It is further assumed that $\mathsf{F}$ is deterministic, although this does not seem to be an essential assumption in our treatment. We will define a TM $\mathsf{G}$ so that when given such an input, it will methodically query the oracle on all the yet unqueried inputs.

For an adaptive list $\mathsf{F}$, we will define the variable $F_k^{\mathbf{o},z}$ as the response to the $k$-th oracle query made by $\mathsf{G} \circ \mathsf{F}^{\mathbf{o}}(z)$ when run with the oracle $\mathbf{o}$ and input $z$. According to the preceeding assumptions, $F_k^{\mathbf{o},z}$ is well defined for all $k \leq |\text{Range}|$, $\mathbf{o} \in \mathcal{O}$ and $z \in \mathcal{Z}$. Let $\mathbf{F}_k^{\mathcal{O},\mathbf{Z}}$ be the variable induced by $F_k$ by choosing $\mathbf{o} \leftarrow \mathcal{O}$ and $z \leftarrow \mathbf{Z}$. For convenience, we note $\mathbf{F}_{k\to l} = \mathbf{F}_{k+1}, \ldots, \mathbf{F}_l$ and $\mathbf{F}_{*l} = \mathbf{F}_{0\to l}$.

Let the distribution of advice strings $\mathbf{Z}$ be dependent on the distribution of oracles $\mathcal{O}$. Let $\mathcal{O}'$ stand for the distribution of oracles that is distributed identically to $\mathcal{O}$ but that is independent from $\mathbf{Z}$. Let $\mathcal{O}'/S_k$ denote the distribution $\mathcal{O}'$ conditioned on agreeing with $\mathcal{O}$ on all the queries $F_{*k}^{\mathcal{O},\mathbf{Z}}$. The goal is then to show a bound on the statistical distance $\Delta(\mathbf{F}_{*k}^{\mathcal{O},\mathbf{Z}}, \mathbf{F}_{k\to k+q}^{\mathcal{O}'/S_k,\mathbf{Z}}; \mathbf{F}_{*(k+q)}^{\mathcal{O},\mathbf{Z}})$. We will use the Kullback-Leibler distance:

$$D(\mathbf{X}||\mathbf{Y}|\mathbf{Z}) = \sum_{z\in\mathcal{Z}} \Pr[\mathbf{Z}=z] \sum_{x\in\mathcal{X}} \Pr[\mathbf{X}=x|\mathbf{Z}=z] \lg\left(\frac{\Pr[\mathbf{X}=x|\mathbf{Z}=z]}{\Pr[\mathbf{Y}=x|\mathbf{Z}=z]}\right) .$$

The following useful properties (Gibbs inequality, chain rule, and funcion applications only decreasing the distance) are well known and easy to verify.

$$\begin{aligned} &D(\mathbf{X}||\mathbf{Y}) \geq 0 \ , \\ &D(\mathbf{X}_1,\ldots,\mathbf{X}_k||\mathbf{Y}_1,\ldots,\mathbf{Y}_k) = \\ &\qquad\qquad D(\mathbf{X}_1||\mathbf{X}_2) + D(\mathbf{X}_2||\mathbf{Y}_2|\mathbf{X}_1) + \ldots D(\mathbf{X}_k||\mathbf{Y}_k|\mathbf{X}_1,\ldots,\mathbf{X}_{k-1}) \ , \\ &D(f(\mathbf{X})||f(\mathbf{Y})) \leq D(\mathbf{X}||\mathbf{Y}) \ . \end{aligned}$$

It is worth noting that for the chain rule,

$$\begin{aligned} &D(\mathbf{X}_2||\mathbf{Y}_2|\mathbf{X}_1) = \\ &\quad \sum_{x_1\in\mathcal{X}_1} \Pr[\mathbf{X}_1=x_1] \sum_{x_2\in\mathcal{X}_2} \Pr[\mathbf{X}=x_2|\mathbf{X}=x_1] \lg\left(\frac{\Pr[\mathbf{X}_2=x_2|\mathbf{X}_1=x_1]}{\Pr[\mathbf{Y}_2=x_2|\mathbf{Y}_1=x_1]}\right) \ , \end{aligned}$$

the conditioning is actually over both the values of $\mathbf{X}_1$ and $\mathbf{Y}_1$. We define[4] $J_k(\mathsf{F}) = D(\mathbf{F}_{*k}^{\mathcal{O},\mathbf{Z}}||\mathbf{F}_{*k}^{\mathcal{O}',\mathbf{Z}}|\mathbf{Z})$ and $J_{k\to l}(\mathsf{F}) = J_l(\mathsf{F}) - J_k(\mathsf{F}) = D(\mathbf{F}_{k\to l}^{\mathcal{O},\mathbf{Z}}||\mathbf{F}_{k\to l}^{\mathcal{O}'/S_k,\mathbf{Z}}|\mathbf{F}_{*k}^{\mathcal{O},\mathbf{Z}},\mathbf{Z})$. We note that although we use a slightly different notation for $J$ that makes the length

[4] Introduction of $S_k$ is due to the chain rule conditioning over both distibutions.

of the list $k$ explicit, this is purely for syntactic convenience. As in the original, let $J_k = \max_{\mathsf{F}} J_k(\mathsf{F})$.

The proof in the original paper requires three properties from $J(\mathsf{F})$. Two of them ($J_k(\mathsf{F}) \geq 0$ and $J_l(\mathsf{F}) \leq J_{k\to l}(\mathsf{F}) + J_k(\mathsf{F})$) follow directly from the properties of Kullback-Leibler distance. The third property $J_k(\mathsf{F}) \leq H(\mathbf{Z})$ is just slightly trickier, but follows trivially from the following lemma.

**Lemma 7.** *Let* $\mathbf{X}$ *and* $\mathbf{Y}$ *be identically distributed. Additionally, let* $\mathbf{Z}$ *be variable independent from* $\mathbf{Y}$ *(but possibly related to* $\mathbf{X}$*). In such a case,* $D(\mathbf{X}||\mathbf{Y}|\mathbf{Z}) \leq H(\mathbf{Z})$.

*Proof.*

$$\begin{aligned}
D(\mathbf{X}||\mathbf{Y}|\mathbf{Z}) &= \sum_{z\in\mathcal{Z}} \Pr[\mathbf{Z}=z] \sum_{x\in\mathcal{X}} \Pr[\mathbf{X}=x|\mathbf{Z}=z] \lg\left(\frac{\Pr[\mathbf{X}=x|\mathbf{Z}=z]}{\Pr[\mathbf{Y}=x|\mathbf{Z}=z]}\right) \\
&= \sum_{z\in\mathcal{Z}} \Pr[\mathbf{Z}=z] \sum_{x\in\mathcal{X}} \Pr[\mathbf{X}=x|\mathbf{Z}=z] \lg\left(\frac{\Pr[\mathbf{X}=x,\mathbf{Z}=z]}{\Pr[\mathbf{Y}=x]\Pr[\mathbf{Z}=z]}\right) \\
&\leq \sum_{z\in\mathcal{Z}} \Pr[\mathbf{Z}=z] \sum_{x\in\mathcal{X}} \Pr[\mathbf{X}=x|\mathbf{Z}=z] \lg\left(\frac{1}{\Pr[\mathbf{Z}=z]}\right) \\
&= \sum_{z\in\mathcal{Z}} \Pr[\mathbf{Z}=z] \lg\left(\frac{1}{\Pr[\mathbf{Z}=z]}\right) = H(\mathbf{Z}) \ ,
\end{aligned}$$

where the inequality is due to $\frac{\Pr[\mathbf{X}=x,\mathbf{Z}=z]}{\Pr[\mathbf{X}=x]} \leq 1$ and $\mathbf{X}, \mathbf{Y}$ are identically distributed. □

**Corollary 1.** $J_k(\mathsf{F}) \leq H(\mathbf{Z})$.

*Proof.*

$$\begin{aligned}
J_k(\mathsf{F}) &= D(\mathbf{F}_{*k}^{\mathcal{O},\mathbf{Z}}||\mathbf{F}_{*k}^{\mathcal{O}',\mathbf{Z}}|\mathbf{Z}) = D(f(\mathcal{O},\mathbf{Z})||f(\mathcal{O}',\mathbf{Z})|\mathbf{Z}) \leq D(\mathcal{O},\mathbf{Z}||\mathcal{O}',\mathbf{Z}|\mathbf{Z}) \\
&= D(\mathcal{O}||\mathcal{O}'|\mathbf{Z}) \leq H(\mathbf{Z}) \ .
\end{aligned}$$

□

The only piece missing is to use $D(\mathbf{X}||\mathbf{Y}|\mathbf{Z})$ for bounding $\Delta(X,Z;Y,Z)$. This is also completely analogous to the proof in Unruh, as

$$\begin{aligned}
\Delta(\mathbf{X},\mathbf{Z};\mathbf{Y},\mathbf{Z}) &= \sum_{z\in\mathcal{Z}} \Pr[\mathbf{Z}=z]\Delta(\mathbf{X};\mathbf{Y}|\mathbf{Z}=z) \leq \sum_{z\in\mathcal{Z}} \Pr[\mathbf{Z}=z]\sqrt{\frac{D(\mathbf{X}||\mathbf{Y}|\mathbf{Z}=z)}{2}} \\
&\leq \sqrt{\frac{1}{2}\sum_{z\in\mathcal{Z}} \Pr[\mathbf{Z}=z]D(\mathbf{X}||\mathbf{Y}|\mathbf{Z}=z)} = \sqrt{\frac{1}{2}D(\mathbf{X}||\mathbf{Y}|\mathbf{Z})} \ ,
\end{aligned}$$

where the first inequality is due to Kullback-Leibler and the second is an application of Jensen's inequality. All the other parts of the proof remain fairly unaltered, with a few pieces (such as replacing $G$ with $\nabla G$) becoming obsolete due to independence requirements being relaxed.

This theorem basically allows one to formally replace a polynomial-length oracle-dependent advice string with just fixing a super-polynomial number of responses to oracle queries, with only negligible chance of the adversary behaving differently. This fits in well with many of the already known proofs for separation results, which will still work even when the number of queries is slightly super-polynomial just as long as it is still negligible when compared with the full domain and range of the oracle function. In such cases, one can then replace the original usually oracle extraction based argumentation with the averaging-based argumentation to yield a stronger result that holds also in the non-uniform model. For instance, this seems to be the case with the work of Simon [10] where it was shown that collision-resistant hash functions cannot be constructed based purely on one-way functions. As his argumentation still remains valid when the adversary makes a super-polynomial number of queries, the result can be generalized to the non-uniform model.

## References

1. Buldas, A., Jürgenson, A., Niitsoo, M.: Efficiency bounds for adversary constructions in black-box reductions. In: Boyd, C., González Nieto, J. (eds.) ACISP 2009. LNCS, vol. 5594, pp. 264–275. Springer, Heidelberg (2009)
2. Buldas, A., Laur, S., Niitsoo, M.: Oracle separation in the non-uniform model. In: Pieprzyk, J., Zhang, F. (eds.) ProvSec 2009. LNCS, vol. 5848, pp. 230–244. Springer, Heidelberg (2009)
3. Gennaro, R., Gertner, Y., Katz, J.: Lower bounds on the efficiency of encryption and digital signature schemes. In: STOC 2003, pp. 417–425 (2003)
4. Gennaro, R., Gertner, Y., Katz, J., Trevisan, L.: Bounds on the efficiency of generic cryptographic constructions. SIAM Journal on Computing 35, 217–246 (2006)
5. Gertner, Y., Kannan, S., Malkin, T., Reingold, O., Viswanathan, M.: The relationship between public key encryption and oblivious transfer. In: FOCS 2000, pp. 325–335 (2000)
6. Hsiao, C.-Y., Reyzin, L.: Finding collisions on a public road, or do secure hash functions need secret coins? In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 92–105. Springer, Heidelberg (2004)
7. Impagliazzo, R., Rudich, S.: Limits on the provable consequences of one-way permutations. In: STOC 1989, pp. 44–61 (1989)
8. Kim, J.H., Simon, D.R., Tetali, P.: Limits on the efficiency of one-way permutation-based hash functions. In: FOCS 1999, pp. 535–542 (1999)
9. Reingold, O., Trevisan, L., Vadhan, S.: Notions of reducibility between cryptographic primitives. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 1–20. Springer, Heidelberg (2004)
10. 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)
11. Unruh, D.: Random oracles and auxiliary input. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 205–223. Springer, Heidelberg (2007)

## A Oracle Extraction Examples

**Theorem 2.** *If* $\underset{pol}{\forall} G \underset{pol}{\exists} A \underset{pol}{\forall} S \exists \mathcal{F}:$ *so that (I)* $\underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\text{Adv}_k(A^f, G^f)\right] = 1 - k^{-\omega(1)}$ *and (II)* $\underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\text{Adv}_k(S^f, f)\right] = k^{-\omega(1)}$*, there exist no strong semi black-box reductions.*

*Proof.* Using the overwhelming average argument for (I) and the negligible average argument for (II), we imply that $\Pr_{f\leftarrow\mathfrak{F}}\left[A^f\,\mathbf{br}\,G^f\right]=1$ and $\Pr_{f\leftarrow\mathfrak{F}}\left[S^f\,\not\mathbf{br} f\right]=1$, which implies $\underset{\text{pol}}{\forall} G\,\underset{\text{pol}}{\exists} A\,\underset{\text{pol}}{\forall} S \exists\mathfrak{F}\colon\ \Pr_{f\leftarrow\mathfrak{F}}\left[A^f\,\mathbf{br}\,G^f\wedge S^f\,\not\mathbf{br} f\right]=1$ and hence from the probabilistic argument: $\underset{\text{pol}}{\forall} G\,\underset{\text{pol}}{\exists} A\,\underset{\text{pol}}{\forall} S\exists f\colon\ \left[A^f\,\mathbf{br}\,G^f\wedge S^f\,\not\mathbf{br} f\right]$, which is the negation of the strong semi black-box reduction condition.[5] □

**Theorem 3.** *If* $\underset{pol}{\forall} G\,\underset{pol}{\exists} A\exists\mathfrak{F}\colon$ *so that (I)* $\mathop{\mathbf{E}}_{f\leftarrow\mathfrak{F}}\left[\mathrm{ADV}_k(A^f,G^f)\right]=1-k^{-\omega(1)}$ *and (II)* $\underset{pol}{\forall} S\colon\ \mathop{\mathbf{E}}_{f\leftarrow\mathfrak{F}}\left[\mathrm{ADV}_k(S^f,f)\right]=k^{-\omega(1)}$*, there exist no uniform weak semi b-b reductions.*

*Proof.* The overwhelming average argument for (I) and the negligible average argument for (II) imply $\Pr_{f\leftarrow\mathfrak{F}}\left[A^f\,\mathbf{br}\,G^f\right]=1$ and $\underset{\text{pol}}{\forall} S\colon\ \Pr_{f\leftarrow\mathfrak{F}}\left[S^f\,\not\mathbf{br} f\right]=1$. By the countability argument for $S$, we obtain $\Pr_{f\leftarrow\mathfrak{F}}\left[\underset{\text{pol}}{\forall} S\colon\ S^f\,\not\mathbf{br} f\right]=1$, which implies $\underset{\text{pol}}{\forall} G\,\underset{\text{pol}}{\exists} A\exists\mathfrak{F}\colon\quad \Pr_{f\leftarrow\mathfrak{F}}\left[A^f\,\mathbf{br}\,G^f\wedge\underset{\text{pol}}{\forall} S\colon\ S^f\,\not\mathbf{br} f\right]=1$, and we have the negation of the weak semi black-box reduction: $\underset{\text{pol}}{\forall} G\,\underset{\text{pol}}{\exists} A\exists f\,\underset{\text{pol}}{\forall} S\colon\ \left[A^f\,\mathbf{br}\,G^f\wedge S^f\,\not\mathbf{br} f\right]$.[6] □

**Theorem 4.** *If* $\exists\mathfrak{F}\colon$ *so that (I)* $\underset{pol}{\forall} G\,\underset{pol}{\exists} A\colon\ \mathop{\mathbf{E}}_{f\leftarrow\mathfrak{F}}\left[\mathrm{ADV}_k(A^f,G^f)\right]=1-k^{-\omega(1)}$ *and (II)* $\underset{pol}{\forall} S\colon\ \mathop{\mathbf{E}}_{f\leftarrow\mathfrak{F}}\left[\mathrm{ADV}_k(S^f,f)\right]=k^{-\omega(1)}$*, there are no uniform variable semi b-b reductions.*

A proof was already presented by the steps $s_1$-$b_3$ in Section 6.

## B Averaging Examples

**Lemma 8.** *The existence of weak semi black-box reductions is equivalent to:*

$$\underset{of}{\exists} p\,\underset{pol}{\forall} A\,\underset{pol}{\exists}\mathcal{S}\,\underset{of}{\exists}\varphi\forall f\colon\quad \mathrm{ADV}_k(\mathcal{S}^f_{\varphi(f)},f)\geq\left[\mathrm{ADV}_k(A^f,p(f))\right]^c\quad \textit{where } c\geq 1\ . \tag{4}$$

*Proof.* Assume first that $\underset{\text{of}}{\exists} p\,\underset{\text{pol}}{\forall} A\forall f\,\underset{\text{pol}}{\exists} S_f\colon\ \mathrm{ADV}_k(S^f_f,f)\geq\left[\mathrm{ADV}_k(A^f,p(f))\right]^c$, i.e. there exists a weak semi black-box reduction and prove (4). Let $\varphi$ be an oracle function so that $\varphi(f)$ is a bit-representation of $S_f$. Let $\mathcal{S}$ be the universal $f$-oracle machine, which when given as input a bit-representation $\varphi(f)$ behaves exactly like $S_f$. This means that $\mathrm{ADV}_k(\mathcal{S}^f_{\varphi(f)},f)=\mathrm{ADV}_k(S^f_f,f)$. Moreover, as such simulation is possible with logarithmic overhead, it follows that $\mathcal{S}_{\varphi(f)}$ is poly-time. As $\mathcal{S}$ and $\varphi$ are the same for all instances of $f$, the statement (4) follows.

From (4) by defining $S_f:=\mathcal{S}_{\varphi(f)}$, there exists $p$ such that for all poly-time $A$ and for all $f$ there is $S_f$, so that $\mathrm{ADV}_k(S^f_f,f)=\mathrm{ADV}_k(\mathcal{S}^f_{\varphi(f)},f)\geq\left[\mathrm{ADV}_k(A^f,p(f))\right]^c$, which proves the existence of weak semi black-box reduction. □

[5] No countability arguments were used.

[6] As we used the countability argument for $S$, the result does not apply to the models where the adversaries are allowed to be non-uniform.

**Theorem 5.** *If* $\underset{of}{\forall} p \underset{pol}{\exists} A \exists \mathcal{F}$: *so that (I)* $\underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(A^f, p(f))\right] \neq k^{-\omega(1)}$ *and (II)* $\underset{pol}{\forall} S \underset{of}{\forall} \varphi \underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(S^f_{\varphi(f)}, f)\right] = k^{-\omega(1)}$*, there exist no weak semi b-b reductions.*

*Proof.* By using (4), (I) and (II), we will derive a contradiction. Let $p$ be as in (4). By applying the assumption of the theorem to this $p$, we conclude that there exist a poly-time oracle machine $A$ and a distribution $\mathcal{F}$ with the properties (I) and (II). Now, from (4) it follows that for this $A$, there exist a poly-time oracle machine $\mathcal{S}$ and an oracle function $\varphi$ such that (*) $\textsc{Adv}_k(\mathcal{S}^f_{\varphi(f)}, f) \geq \left[\textsc{Adv}_k(A^f, p(f))\right]^c$ holds for all $f$. By (II), we have (**) $\underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(\mathcal{S}^f_{\varphi(f)}, f)\right] = k^{-\omega(1)}$. Finally, by averaging (*) and using the Jensen's inequality we have:

$$\underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(\mathcal{S}^f_{\varphi(f)}, f)\right] \geq \underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(A^f, p(f))^c\right] \geq \left[\underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(A^f, p(f))\right]\right]^c ,$$

which is a contradiction between (I) and (**). □

**Lemma 9.** *The existence of variable semi black-box reductions is equivalent to:*

$$\underset{of}{\exists} \psi \underset{pol}{\exists} \mathcal{P} \underset{pol}{\forall} A \underset{pol}{\exists} \mathcal{S} \underset{of}{\exists} \varphi \forall f: \quad \textsc{Adv}_k(\mathcal{S}^f_{\varphi(f)}, f) \geq \left[\textsc{Adv}_k(A^f, \mathcal{P}^f_{\psi(f)})\right]^{O(1)} . \tag{5}$$

*Proof.* Assume first that $\forall f \underset{\text{pol}}{\exists} G_f \underset{\text{pol}}{\forall} A \underset{\text{pol}}{\exists} S_f$: $\textsc{Adv}_k(S^f_f, f) \geq \left[\textsc{Adv}_k(A^f, G^f_f)\right]^c$, i.e. there exists a variable semi black-box reduction, and prove (5). Let $\psi$ be a mapping so that $\psi(f)$ is the bit-string representation of $G_f$. Let $\mathcal{P}$ be the universal $f$-oracle machine so that $\mathcal{P}_{\psi(f)}$ behaves identical to $G_f$. Hence, $\textsc{Adv}_k(A^f, \mathcal{P}^f_{\psi(f)}) = \textsc{Adv}_k(A^f, G^f_f)$, and due to the efficiency of simulation, $\mathcal{P}_{\psi(f)}$ is poly-time. For every $A$ we define $\mathcal{S}$ and $\varphi$ like in Lemma 8. The statement (5) follows. □

**Theorem 6.** *If* $\forall \psi \exists \mathcal{F}$: *so that (I)* $\underset{pol}{\forall} G \underset{pol}{\exists} A \underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(A^f, G_{\psi(f)})\right] \neq k^{-\omega(1)}$ *and (II)* $\underset{pol}{\forall} S \underset{of}{\forall} \varphi \underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(S^f_{\varphi(f)}, f)\right] = k^{-\omega(1)}$*, there exist no weak semi b-b reductions.*

*Proof.* By using (5), (I) and (II), we derive a contradiction. Let $\psi$ and $\mathcal{P}$ be as in (4). By applying the assumption of the theorem to $\psi$, we conclude that a distribution $\mathcal{F}$ with the properties (I) and (II). By applying (I) to $\mathcal{P}$, we conclude that there exists $A$ such that (*) $\underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(A^f, \mathcal{P}_{\psi(f)})\right] \neq k^{-\omega(1)}$. From (5) it follows that there exist a poly-time $\mathcal{S}$ and an oracle function $\varphi$ so that (**) $\textsc{Adv}_k(\mathcal{S}^f_{\varphi(f)}, f) \geq \left[\textsc{Adv}_k(A^f, \mathcal{P}^f_{\psi(f)})\right]^c$ for all $f$. By (II), we have (***) $\underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(\mathcal{S}^f_{\varphi(f)}, f)\right] = k^{-\omega(1)}$. Finally, by averaging (**) and using the Jensen's inequality, we have

$$\underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(\mathcal{S}^f_{\varphi(f)}, f)\right] \geq \underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(A^f, \mathcal{P}^f_{\psi(f)})^c\right] \geq \left[\underset{f \leftarrow \mathcal{F}}{\mathbf{E}} \left[\textsc{Adv}_k(A^f, \mathcal{P}^f_{\psi(f)})\right]\right]^c .$$

A contradiction between (*) and (***). □

# Attribute-Based Identification: Definitions and Efficient Constructions[⋆]

Hiroaki Anada, Seiko Arita, Sari Handa, and Yosuke Iwabuchi

Graduate School of Information Security, Institute of Information Security, Japan
2-14-1 Tsuruya-Cho, Kanagawa-Ku, Yokohama-Shi, 221-0835, Japan
hiroaki.anada@gmail.com, {arita,mgs125502,mgs123101}@iisec.ac.jp

**Abstract.** We propose a notion of attribute-based identification (ABID) in two flavors: prover-policy ABID (PP-ABID) and verifier-policy ABID (VP-ABID). In a PP-ABID scheme, a prover has an authorized access policy written as a boolean formula over attributes, while each verifier maintains a set of attributes. The prover is accepted when his access policy fits the verifier's set of attributes. In a VP-ABID scheme, a verifier maintains an access policy written as a boolean formula over attributes, while each prover has a set of authorized attributes. The prover is accepted when his set of attributes satisfies the verifier's access policy. Our design principle is first to construct key-policy and ciphertext-policy attribute-based key encapsulation mechanisms (KP-ABKEM and CP-ABKEM). Second, we convert KP-ABKEM and CP-ABKEM into challenge-and-response PP-ABID and VP-ABID, respectively, by encapsulation-and-decapsulation. There, we show that KP-ABKEM and CP-ABKEM only have to be secure against chosen-ciphertext attacks on one-wayness (OW-CCA secure) for the obtained PP-ABID and VP-ABID to be secure against concurrent man-in-the-middle attacks (cMiM secure). According to the design principle, we construct concrete KP-ABKEM and CP-ABKEM with the OW-CCA security by enhancing the KP-ABKEM of Ostrovsky, Sahai and Waters and CP-ABKEM of Waters, respectively. Finally, we obtain concrete PP-ABID and VP-ABID schemes that are proved to be selectively secure in the standard model against cMiM attacks.

**Keywords:** access policy, attribute, identification, key encapsulation mechanism.

## 1 Introduction

An identification (ID) scheme enables a prover to convince a verifier that the prover certainly knows a secret key that corresponds to the matching public key. For example, $\Sigma$-protocols [7] such as the Schnorr protocol [14,5] are widely recognized. In these ID schemes, the public key to which the verifier refers limits the corresponding secret key uniquely, and also, the corresponding prover.

[⋆] Supported by MEXT-Supported Program for the Strategic Research Foundation at Private Universities, 2011-2013.

In this paper, we will describe an *attribute-based identification (ABID)*. In an ABID scheme, each entity has credentials called *attributes*. On the other hand, an *access policy* is written as a boolean formula over those attributes. Then, a verifier can identify that a prover certainly belongs to a set of entities that have authorized access policies that fit the verifier's attributes, or, in the dual flavor, a verifier can identify that a prover certainly belongs to a set of entities that possess authorized attributes that satisfy the verifier's access policy. Hence, ABID schemes can be considered as an expansion of the usual ID schemes.

However, ABID schemes are not a mere expansion, but have useful applications beyond those of the usual ID schemes. For example, the following scenarios of smart card systems motivate us to apply ABID.

**Functional Tickets.** Suppose that we are going to stay at a resort complex, a ski resort, for instance. We search Web sites or brochures for information about services: available dates, accommodation, ski lifts, restaurants in ski areas and hot springs around the areas. For each service, we usually buy a ticket, paying with money or using a credit card. However, acquiring many tickets and carrying a wallet is inconvenient, and therefore, it would be more convenient if we could gain access to these services by using only one smart card. In the smart card, a service authority writes an access policy in terms of the service names that we choose, for instance, [January 1 to 4, 2014] AND [[Hotel A] OR [Ski Lift AND [Day OR Night]] OR [Lunch OR Beer] OR [Hot Spring X]]. A *functional ticket* is a ticket embedded in a smart card that realizes an access policy as a boolean formula over services, as in this scenario. Here, the access policy is chosen according to our requirements.

**Functional Gates.** Suppose that we have to design a security gate system for an office building in which different kinds of people work: employees of several companies holding many different positions, security guards, food service staffs, cleaning staffs and janitors. There are also many types of security gates to be designed: building entrances, intelligent elevators to limit available floors, company gates, common refreshment areas and room doors for the above staffs. In this case, one solution is to use smart cards and gates with sensors. That is, an authority issues each person a smart card in which a set of attribute data is written. Each gate decides whether to "pass" each person carrying a smart card according to the gate's access policy, for instance, [Year 2014] AND [[[Company A] AND [Manager]] OR [Security Guard]]. A *functional gate* is a gate that maintains an access policy as a boolean formula over attributes of people, as in this scenario. Here, the access policy is chosen according to the kind of people that the gate should allow to pass.

## 1.1 Our Contributions

Bearing the above scenarios in mind, we propose a notion of attribute-based identification (ABID) that has two flavors corresponding to the scenarios: *prover-policy* ABID and *verifier-policy* ABID.

**Prover-Policy ABID.** In a prover-policy ABID scheme (PP-ABID, for short), a prover has his own authorized access policy, while each verifier maintains its attributes. Here, the access policy is described over the verifier's attributes. Sending his access policy, each prover queries an authority for his secret key. Then, using this secret key, each prover can convince the verifier that his access policy fits the verifier's attributes. Our PP-ABID defined in this way realizes a functional ticket system.

**Verifier-Policy ABID.** In a verifier-policy ABID scheme (VP-ABID, for short), a verifier maintains its access policy, while each prover has his own authorized attributes. Here, the access policy is described over the prover's attributes. Sending his attributes, each prover queries an authority for his secret key. Then, using this secret key, each prover convinces the verifier that his attributes satisfy the verifier's access policy. Our VP-ABID defined in this way realizes a functional gate system.

**Attack and Security Analysis.** An adversary's objective is *impersonation*: giving a *target* set of attributes (or, a *target* access policy) to a verifier, the adversary tries to make the verifier accept him.

First, to reflect a *collusion attack* (that is, an attack launched by collecting secret keys that satisfy a condition), we consider an attack model in which an adversary issues key-extraction queries, as is the case for attribute-based encryptions [13,15]. The condition is that the adversary cannot collect any secret key whose intrinsic access policy fits the target set of attributes (or, whose intrinsic set of attributes satisfies the target access policy).

Our main objective is to define a model of *concurrent man-in-the-middle attack (cMiM attack)* in the setting of ABID. "Concurrent" means that an adversary can invoke provers that have *different* secret keys corresponding to different access policies (or, different sets of attributes). The adversary interacts with these provers in an arbitrarily interleaved order of messages. Then, interacting with a verifier on a target set of attributes (or, on a target access policy, respectively) the adversary tries to impersonate a prover. The concurrent attack modeled in this way is a real threat, especially to smart card systems. On the other hand, "man-in-the-middle (MiM)" means that an adversary stands between a prover and a verifier simultaneously. Typically, the adversary first receives a message from the verifier, and then, the adversary begins to interact with the prover *adaptively* to the verifier's message. The MiM attack and the cMiM attack modeled in this way are real threats, especially to network applications.

As is the case for usual ID schemes, *reset attacks* should be considered. In a reset attack, an adversary aborts an interaction at any point, and then rewinds the interaction back to any other point to start the interaction again. At that re-starting point, the adversary is allowed to change messages as long as the interaction remains valid (as captured by the word "reset"). Such a reset attack is a strong threat, not only to smart card systems [4] (including the functional tickets and functional gates described above) but also to virtual machine services in cloud computing [17]. As our contribution, an ABID constructed using our generic conversion becomes secure against the reset attacks in both senses of prover-resettable and verifier-resettable [4].

It is desirable that a verifier learns nothing about a prover more than that he belongs to the set of entities that have access policies fitting the verifier's attributes (or, belongs to the set of entities that possess attributes satisfying the verifier's access policy). In fact, by this property (*anonymity*), the prover's privacy is protected when using a functional ticket, as opposed to using a credit card the track of which is recorded. As our contribution, our concrete ABID in Section 5 possesses this anonymity.

**Design Principle.** First, we construct key-policy and ciphertext-policy attribute-based key encapsulation mechanisms (KP-ABKEM and CP-ABKEM [13,15]). Second, we convert the KP-ABKEM and CP-ABKEM into challenge-and-response PP-ABID and VP-ABID, respectively, by encapsulation-and-decapsulation. There, we show that KP-ABKEM and CP-ABKEM only have to be secure against chosen-ciphertext attacks on one-wayness (OW-CCA secure) for the obtained PP-ABID and VP-ABID to be secure against cMiM attacks (cMiM secure). We stress that the security of indistinguishability against chosen-ciphertext attacks (the IND-CCA security) is excessive, and the OW-CCA security is enough for constructing a cMiM secure ABID.

**Concrete Constructions.** We construct KP-ABKEM and CP-ABKEM with the OW-CCA security from the KP-ABKEM of Ostrovsky, Sahai and Waters [13] and CP-ABKEM of Waters [15]. Their KEMs are secure in the indistinguishability game of chosen-plaintext attack (IND-CPA secure). Our strategy is to apply the algebraic trick of Boneh and Boyen [3] and Kiltz [11] to attain CCA security. Then our generic conversions yield concrete PP-ABID and VP-ABID.

**New Number Theoretic Assumptions.** We will introduce the Computational Bilinear Diffie-Hellman Assumption with Gap on Target Group and the Computational $q$-Parallel Bilinear Diffie-Hellman Exponent Assumption with Gap on Target Group. These assumptions are reasonable, for example, for the bilinear map of a pairing on a elliptic curve. We need these assumptions for security proofs of concrete constructions.

## 1.2 Related Works

**Anonymous Deniable Predicate Authentication.** First, we should refer to the work of Yamada et al. [16]. Our generic construction of ABID can be considered as a special case of their predicate authentication; however, it differs in at least two points. The first point is that our objective is to provide a simple, fast ABID. In contrast, Yamada et al.'s objective is to apply their verifiable predicate encryption to yield an anonymous deniable message authentication. In fact, we simply consider a 2-round challenge-and-response ABID, whereas they proposed a 6-round protocol for deniability. The second point is that we provide more efficient concrete ABID by using the algebraic trick ([3,11]). In contrast, they used their generic transformation which causes a longer secret key, a longer ciphertext and more computational costs for encryption and decryption than ours, because a verification key of a one-time signature is involved in their generic transformation.

**Identification Scheme from KEM.** Anada and Arita [1] proposed a design principle of obtaining a cMiM secure ID scheme by constructing KEM. Their concrete ID scheme is more efficient than known $\Sigma$-protocol-based cMiM secure ID schemes ([9]). Our scheme can be seen as an attribute-based version of theirs.

### 1.3 Organization of the Paper

In Section 2, we survey the required terms. In Section 3, we define the notions of PP-ABID and VP-ABID, cMiM attacks and security against it. In Section 4, we provide generic conversions from KP-ABKEM to PP-ABID and from CP-ABKEM to VP-ABID. In Section 5, we construct concrete KP-ABKEM and CP-ABKEM. Finally, we obtain concrete PP-ABID and VP-ABID. In Section 6, we present the conclusions of our study. Because of space limitation, the case of PP-ABID is described in the main text and the case of VP-ABID is only shortly described in the Appendix.

## 2 Preliminaries

The security parameter is denoted by $\lambda$. A prime of bit length $\lambda$ is denoted by $p$. A multiplicative cyclic group of order $p$ is denoted by $\mathbb{G}$. The ring of the exponent domain of $\mathbb{G}$, which consists of integers from 0 to $p-1$ with modulo $p$ operation, is denoted by $\mathbb{Z}_p$. When an algorithm $A$ with input $a$ outputs $z$, we denote it as $z \leftarrow A(a)$. When $A$ with input $a$ and $B$ with input $b$ interact with each other and $B$ outputs $z$, we denote it as $z \leftarrow \langle A(a), B(b) \rangle$. When $A$ has oracle-access to $\mathcal{O}$, we denote it as $A^{\mathcal{O}}$. When $A$ has concurrent oracle-access to $n$ oracles $\mathcal{O}_1, \ldots, \mathcal{O}_n$, we denote it as $A^{\mathcal{O}_i|_{i=1}^n}$. Here "concurrent" means that $A$ accesses to oracles in arbitrarily interleaved order of messages. A probability of an event E is denoted by Pr[E]. A probability of an event E on condition that events $\mathrm{E}_1, \ldots, \mathrm{E}_m$ occur in this order is denoted as $\Pr[\mathrm{E}_1; \cdots; \mathrm{E}_m : \mathrm{E}]$.

### 2.1 Access Structure

Let $\mathcal{U} = \{\chi_1, \ldots, \chi_u\}$ be an attribute universe, or simply set $\mathcal{U} = \{1, \ldots, u\}$. We must distinguish two cases: the case that $\mathcal{U}$ is small (i.e. $|\mathcal{U}| = u$ is bounded by some polynomial in $\lambda$) and the case that $\mathcal{U}$ is large (i.e. $u$ is not necessarily bounded). We assume the *small case* unless we state the large case explicitly. An *access structure*, which reflects a given access policy, is defined as a collection $\mathbb{A}$ of non-empty subsets of $\mathcal{U}$. That is, $\mathbb{A} \subset 2^{\mathcal{U}} \backslash \{\phi\}$. An access structure $\mathbb{A}$ is called *monotone* if for any $B \in \mathbb{A}$ and $B \subset C$, $C \in \mathbb{A}$ holds. We will consider in this paper only monotone access structures.

## 2.2 Linear Secret-Sharing Scheme

A secret-sharing scheme $\Pi$ over a set of parties $\mathcal{P}$ is called a linear secret-sharing scheme (LSSS) over $\mathbb{Z}_p$ ([2]), if $\Pi$ satisfies the following conditions.

1. The shares for each party form a vector over $\mathbb{Z}_p$
2. There exists a matrix $M$ called the *share-generating matrix* for $\Pi$, of size $l \times n$, and a function $\rho$ which maps each row index $i$ of $M$ to a party $\mathcal{P}$, $\rho : \{1, ..., l\} \to \mathcal{P}$.

To make shares for a secret $s \in \mathbb{Z}_p$, we first choose $n-1$ random values $v_2, \ldots, v_n \in \mathbb{Z}_p$ and form a vector $\boldsymbol{v} = (s, v_2, \ldots, v_n)$. For $i = 1$ to $l$, we calculate each share $\lambda_i = \boldsymbol{v} \cdot M_i$, where $M_i$ denotes the $i$-th row vector of $M$ and $\cdot$ denotes the formal inner product. The share $\lambda_i$ belongs to the party $\rho(i)$.

Looking at $\mathcal{P}$ as an attribute universe $\mathcal{U}$, $\Pi$ determines an access structure $\mathbb{A}$ as $(M, \rho)$ ([13,15]). Suppose that an attribute set $S \subset \mathcal{U}$ satisfies $\mathbb{A}$ ($S \in \mathbb{A}$) and put $I_S = \rho^{-1}(S) \subset \{1, \ldots, l\}$. Then, there exists a set of constants $\{\omega_i \in \mathbb{Z}_p; i \in I_S\}$ called *linear reconstruction constants* ([2]) that satisfies $\sum_{i \in I_S} \omega_i \lambda_i = s$. These constants $\{\omega_i\}_{i \in I_S}$ can be computed in time polynomial in the size of $M$. We denote the algorithm by $\mathrm{Recon}(I_S, M)$. If $S$ does not satisfy $\mathbb{A}$ ($S \notin \mathbb{A}$), then no such constants $\{\omega_i\}_{i \in I_S}$ exist, but instead, there is a vector $\boldsymbol{w} = (w_1, \ldots, w_n) \in \mathbb{Z}_p^n$ such that $w_1 = 1$ and $\boldsymbol{w} \cdot M_i = 0$ for all $i \in I_S$. $\boldsymbol{w}$ also can be computed in time polynomial in the size of $M$ ([15]).

## 2.3 Key-Policy Attribute-Based KEM

**Scheme.** A key-policy ABKEM, `KP-ABKEM`, consists of four probabilistic polynomial time (PPT, for short) algorithms (Setup, KeyGen, Encap, Decap).

**Setup**$(\lambda, \mathcal{U}) \to$ **(PK, MSK).** Setup takes as input the security parameter $\lambda$ and the attribute universe $\mathcal{U}$. It returns a public key PK and a master secret key MSK.

**KeyGen(PK, MSK,** $\mathbb{A}) \to \mathbf{SK}_{\mathbb{A}}$**.** A key generation algorithm KeyGen takes as input the public key PK, the master secret key MSK and an access structure $\mathbb{A}$. It returns a secret key $\mathrm{SK}_{\mathbb{A}}$ that corresponds to $\mathbb{A}$.

**Encap(PK,** $S) \to (\kappa, \psi)$**.** Encap takes as input the public key PK and an attribute set $S$. It returns a random KEM key $\kappa$ and its encapsulation $\psi$ (we also call it a ciphertext). We denote the set of all possible output $(\kappa, \psi)$ of $\mathrm{Encap}(\mathrm{PK}, S)$ by $[\mathrm{Encap}(\mathrm{PK}, S)]$. If $(\tilde{\kappa}, \tilde{\psi}) \in [\mathrm{Encap}(\mathrm{PK}, S)]$, then $(\tilde{\kappa}, \tilde{\psi})$ is called *consistent* and otherwise, *inconsistent*.

**Decap(PK,** $\mathbf{SK}_{\mathbb{A}}, \psi) \to \hat{\kappa}$**.** Decap takes as input the public key PK, an encapsulation $\psi$ and a secret key $\mathrm{SK}_{\mathbb{A}}$. It returns a decapsulation result $\hat{\kappa}$ of $\psi$ under $\mathrm{SK}_{\mathbb{A}}$. We demand correctness of `KP-ABKEM` that for any $\lambda$ and $\mathcal{U}$, and if $S \in \mathbb{A}$, then $\Pr[(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathrm{Setup}(\lambda, \mathcal{U}); \mathrm{SK}_{\mathbb{A}} \leftarrow \mathrm{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A}); (\kappa, \psi) \leftarrow \mathrm{Encap}(\mathrm{PK}, S); \hat{\kappa} \leftarrow \mathrm{Decap}(\mathrm{PK},\ \mathrm{SK}_{\mathbb{A}}, \psi) : \kappa = \hat{\kappa}] = 1$.

```
$\mathbf{Exprmt}^{\text{ow-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda,\mathcal{U})$: //Adaptive $S^*$
    $(\mathrm{PK},\mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda,\mathcal{U})$
    $S^* \leftarrow \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathcal{DEC}(\mathrm{PK},\mathrm{SK}.,\cdot)}(\mathrm{PK},\mathcal{U})$
    $(\kappa^*,\psi^*) \leftarrow \mathbf{Encap}(\mathrm{PK},S^*)$
    $\hat{\kappa^*} \leftarrow \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathcal{DEC}(\mathrm{PK},\mathrm{SK}.,\cdot)}(\psi^*)$
    If $\hat{\kappa^*} = \kappa^*$ Return WIN else Return LOSE
```

```
$\mathbf{Exprmt}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda,\mathcal{U})$: //Selective $S^*$
    $(\mathrm{PK},\mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda,\mathcal{U})$
    $S^* \leftarrow \mathcal{A}(\lambda,\mathcal{U})$
    $(\kappa^*,\psi^*) \leftarrow \mathbf{Encap}(\mathrm{PK},S^*)$
    $\hat{\kappa^*} \leftarrow \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathcal{DEC}(\mathrm{PK},\mathrm{SK}.,\cdot)}(\mathrm{PK},\psi^*)$
    If $\hat{\kappa^*} = \kappa^*$ Return WIN else Return LOSE
```

**Fig. 1.** The experiment of an adversary $\mathcal{A}$ that executes a chosen-ciphertext attack on one-wayness of `KP-ABKEM`. The left side: the case of adaptive $S^*$; the right side: the case of selective $S^*$.

**Chosen-Ciphertext Attack on One-Wayness of KP-ABKEM and Security.** The following experiment $\mathbf{Exprmt}^{\text{ow-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda,\mathcal{U})$ of an adversary $\mathcal{A}$ defines the game of chosen-ciphertext attack on one-wayness of `KP-ABKEM` (the OW-CCA game).

In the experiment, $\mathcal{A}$ issues two types of queries. One is key-extraction queries to the key-generation oracle $\mathcal{KG}$. Giving an attribute set $\mathbb{A}_i$, $\mathcal{A}$ queries $\mathcal{KG}(\mathrm{PK}, \mathrm{MSK}, \cdot)$ for the secret key $\mathrm{SK}_{\mathbb{A}_i}$. Another is decapsulation queries to the decapsulation oracle $\mathcal{DEC}$. Giving a pair $(\mathbb{A}_j,\psi_j)$ of an attribute set and an encapsulation, $\mathcal{A}$ queries $\mathcal{DEC}(\mathrm{PK},\mathrm{SK}.,\cdot)$ for the decapsulation result $\hat{\kappa}_j$. Here an attribute set $S_j$, which is used to generate a ciphertext, is included in $\psi_j$. When $S_j \notin \mathbb{A}_j$, $\hat{\kappa}_j = \perp$ is replied to $\mathcal{A}$.

The attribute set $S^*$ declared by $\mathcal{A}$ is called a *target attribute set*. The encapsulation $\psi^*$ is called a *challenge ciphertext*. Two restrictions are imposed on $\mathcal{A}$ concerning $S^*$ and $\psi^*$. In key-extraction queries, each attribute set $\mathbb{A}_i$ must satisfy $S^* \notin \mathbb{A}_i$. In decapsulation queries, each pair $(\mathbb{A}_j,\psi_j)$ must satisfy $S^* \notin \mathbb{A}_j \vee \psi_j \neq \psi^*$. Both types of queries are at most $q_{\mathrm{k}}$ and $q_{\mathrm{d}}$ times in total, respectively, which are bounded by a polynomial in $\lambda$.

The *advantage* of $\mathcal{A}$ over `KP-ABKEM` in the OW-CCA game is defined as [1]

$$\mathbf{Adv}^{\text{ow-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda) \overset{\text{def}}{=} \Pr[\mathbf{Exprmt}^{\text{ow-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda,\mathcal{U}) \text{ returns WIN}].$$

`KP-ABKEM` is called *secure against chosen-ciphertext attacks on one-wayness* if, for any PPT $\mathcal{A}$ and for any $\mathcal{U}$, $\mathbf{Adv}^{\text{ow-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda)$ is negligible in $\lambda$.

**Selective Security.** In the *selective game on a target attribute set* (OW-sel-CCA game), $\mathcal{A}$ declares $S^*$ *before* $\mathcal{A}$ receives PK. $\mathbf{Exprmt}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda,\mathcal{U})$ defines the selective game. The *advantage* in the OW-sel-CCA game is defined as

$$\mathbf{Adv}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda) \overset{\text{def}}{=} \Pr[\mathbf{Exprmt}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda,\mathcal{U}) \text{ returns WIN}].$$

`KP-ABKEM` is called *selectively secure against chosen-ciphertext attacks on one-wayness* if, for any PPT $\mathcal{A}$ and for any $\mathcal{U}$, $\mathbf{Adv}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda)$ is negligible in $\lambda$.

[1] Although we follow the convention, we should write the advantage as a function of $\lambda$ and $u$: $\mathbf{Adv}^{\text{ow-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda,u)$, where $u$ is the size of the attribute universe $\mathcal{U}$.

### 2.4 Bilinear Map

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order $p$. We call $\mathbb{G}$ a source group and $\mathbb{G}_T$ a target group. Let $g$ be a generator of $\mathbb{G}$ and $e$ be a bilinear map, $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. The map $e$ satisfies

1. Bilinearity: for all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p,$ we have $e(u^a, v^b) = e(u, v)^{ab}$.
2. Non-degeneracy: $e(g, g) \neq \mathrm{id}_{\mathbb{G}_T}$(: the identity element of the group $\mathbb{G}_T$).

Groups and a bilinear map are generated by a PPT algorithm **Grp** on input $\lambda$: $(p, \mathbb{G}, \mathbb{G}_T,\ g, e) \leftarrow \mathbf{Grp}(\lambda)$. We assume that the group operation in $\mathbb{G}$ and $\mathbb{G}_T$ and the bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ are computable in time polynomial in $\lambda$.

### 2.5 Computational Bilinear Diffie-Hellman Assumption with Gap on Target Group

We introduce in this paper a new number theoretic assumption, which we call the *Computational Bilinear Diffie-Hellman Assumption with Gap on Target Group*. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map. Let $a, b, c \in \mathbb{Z}_p, a, b, c \neq 0$, be chosen at random. Put $A := g^a, B := g^b, C := g^c$. Then our new assumption says it is at most with a negligible probability in $\lambda$ that, for any PPT algorithm $\mathcal{B}$ given input $(g, A, B, C)$, to output $Z = e(g, g)^{abc} \in \mathbb{G}_T$, even with the aid of the decisional Diffie-Hellman oracle for $\mathbb{G}_T$: $\mathcal{DDH}_{\mathbb{G}_T}(\cdot, \cdot, \cdot, \cdot)$. Here a tuple $(\overline{g}, \overline{g}^{z_1}, \overline{g}^{z_2}, \overline{g}^{z_3}) \in \mathbb{G}_T^4$ $(\overline{g} := e(g, g))$ is called a Diffie-Hellman tuple (in $\mathbb{G}_T$) if $z_1 z_2 = z_3$. The oracle $\mathcal{DDH}_{\mathbb{G}_T}$ returns TRUE or FALSE according to whether an input tuple is a Diffie-Hellman tuple or not, respectively. The probability for $\mathcal{B}$ to output $e(g, g)^{abc}$ is denoted as $\mathbf{Adv}^{\text{c-bdh-gap}}_{\mathcal{B},(e,\mathbb{G},\mathbb{G}_T)}(\lambda)$ (the advantage of $\mathcal{B}$ in the computational BDH game with gap on $\mathbb{G}_T$). Note that the above assumption is *different* from the *Gap Bilinear Diffie-Hellman Assumption* [6].

### 2.6 Target Collision Resistant Hash Functions

Target collision resistant (TCR) hash functions [12] are treated as a family $\mathit{Hfam}_\lambda = \{H_\mu\}_{\mu \in \mathit{HKey}_\lambda}$. The advantage $\mathbf{Adv}^{\text{tcr}}_{\mathcal{CF},\mathit{Hfam}_\lambda}(\lambda)$ of a PPT algorithm $\mathcal{CF}$ over $\mathit{Hfam}_\lambda$ is defined as the success probability to find a target collision.

## 3 Attribute-Based Identification

In this section, we define a notion of *prover-policy* attribute-based identification (PP-ABID), a concurrent man-in-the-middle attack on PP-ABID and security against it. The case of *verifier-policy* ABID is described in Appendix A in a dual manner to PP-ABID on an access structure $\mathbb{A}$ and an attribute set $S$.

### 3.1 Prover-Policy ABID

**Scheme.** `PP-ABID` consists of four PPT algorithms (Setup, KeyGen, P, V).

**Setup**$(\lambda,\mathcal{U}) \rightarrow$ **(PK, MSK).** Setup takes as input the security parameter $\lambda$ and the attribute universe $\mathcal{U}$. It outputs a public key PK and a master secret key MSK.

**KeyGen(PK, MSK,** $\mathbb{A}$**)** $\rightarrow$ **SK**$_{\mathbb{A}}$**.** A key-generation algorithm KeyGen takes as input the public key PK, the master secret key MSK and an access structure $\mathbb{A}$. It outputs a secret key $\mathrm{SK}_{\mathbb{A}}$ corresponding to $\mathbb{A}$.

**P(PK, SK**$_{\mathbb{A}}$**) and V(PK,** $S$**).** P and V are interactive algorithms called a *prover* and a *verifier*, respectively. P takes as input the public key PK and the secret key $\mathrm{SK}_{\mathbb{A}}$. Here the secret key $\mathrm{SK}_{\mathbb{A}}$ is given to P by an authority that runs KeyGen(PK,MSK,$\mathbb{A}$). V takes as input the public key PK and an attribute set $S$. P is provided V's attribute set $S$ by the first round. P and V interact with each other for some, at most constant rounds. Then, V finally returns its decision bit $b$. $b = 1$ means that V *accepts* P in the sense P has a secret key $\mathrm{SK}_{\mathbb{A}}$ such that S satisfies $\mathbb{A}$. $b = 0$ means that V *rejects* P. We demand correctness of PP-ABID that for any $\lambda$ and $\mathcal{U}$, and if $S \in \mathbb{A}$, then $\Pr[(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathrm{Setup}(\lambda,\mathcal{U}); \mathrm{SK}_{\mathbb{A}} \leftarrow \mathrm{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A}); b \leftarrow \langle \mathrm{P}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}}), \mathrm{V}(\mathrm{PK}, S)\rangle : b = 1] = 1$.

## 3.2 Concurrent Man-in-the-Middle Attack on PP-ABID and Security

An adversary $\mathcal{A}$'s objective is impersonation. $\mathcal{A}$ tries to make a verifier V accept with a target attribute set $S^*$. The following experiment $\mathbf{Exprmt}^{\mathrm{cmim}}_{\mathcal{A},\texttt{PP-ABID}}(\lambda,\mathcal{U})$ of an adversary $\mathcal{A}$ defines the game of concurrent man-in-the-middle attack (cMiM attack, for short) on PP-ABID.

```
$\mathbf{Exprmt}^{\mathrm{cmim}}_{\mathcal{A},\texttt{PP-ABID}}(\lambda,\mathcal{U})$: //Adaptive $S^*$
  $(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda,\mathcal{U})$
  $S^* \leftarrow \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathbf{P}_i(\mathrm{PK},\mathrm{SK}_{\mathbb{A}_i})|_{i=1}^{q'_\mathrm{P}}}(\mathrm{PK},\mathcal{U})$
  $b \leftarrow \langle \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathbf{P}_i(\mathrm{PK},\mathrm{SK}_{\mathbb{A}_i})|_{i=1}^{q_\mathrm{P}}},$
          $\mathbf{V}(\mathrm{PK}, S^*)\rangle$
  If $b = 1$ Return WIN else Return LOSE
```

```
$\mathbf{Exprmt}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\texttt{PP-ABID}}(\lambda,\mathcal{U})$: //Seletive $S^*$
  $(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda,\mathcal{U})$
  $S^* \leftarrow \mathcal{A}(\lambda,\mathcal{U})$
  $b \leftarrow \langle \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathbf{P}_i(\mathrm{PK},\mathrm{SK}_{\mathbb{A}_i})|_{i=1}^{q_\mathrm{P}}}(\mathrm{PK}),$
          $\mathbf{V}(\mathrm{PK}, S^*)\rangle$
  If $b = 1$ Return WIN else Return LOSE
```

**Fig. 2.** The experiment of an adversary $\mathcal{A}$ that executes a cMiM attack on PP-ABID. The left side: the case of adaptive $S^*$; the right side: the case of selective $S^*$.

In the experiment, $\mathcal{A}$ issues key-extraction queries to the key-generation oracle $\mathcal{KG}$. Giving an access structure $\mathbb{A}_i$, $\mathcal{A}$ queries $\mathcal{KG}(\mathrm{PK}, \mathrm{MSK}, \cdot)$ for the secret key $\mathrm{SK}_{\mathbb{A}_i}$. In addition, the adversary $\mathcal{A}$ invokes provers $\mathrm{P}_j(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}_j})$ ($j = 1, \dots, q'_\mathrm{p}, \dots, q_\mathrm{p}$) by giving an access structure $\mathbb{A}_j$ of $\mathcal{A}$'s choice. Acting as a verifier with an attribute set $S_j$, $\mathcal{A}$ interacts with each $\mathrm{P}_j$.

The attribute set $S^*$ declared by $\mathcal{A}$ is called a *target attribute set.* Two restrictions are imposed on $\mathcal{A}$ concerning $S^*$. In key-extraction queries, each access structure $\mathbb{A}_i$ must satisfy $S^* \notin \mathbb{A}_i$. In interactions with each prover, every transcript of messages of a whole interaction with a prover $\mathrm{P}_j(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}_j})$ must not be equal to a transcript of messages of a whole interaction with a verifier $\mathrm{V}(\mathrm{PK}, S^*)$ (that is, a mere *relay of messages* is prohibited in the game of man-in-the-middle attack), or, $S^* \notin \mathbb{A}_j$. The number of key-extraction queries and the number of invoked provers are at most $q_{\mathrm{k}}$ and $q_{\mathrm{p}}$ in total, respectively, which are bounded by a polynomial in $\lambda$.

The *advantage* of $\mathcal{A}$ over `PP-ABID` in the game of cMiM attack is defined as

$$\mathbf{Adv}^{\mathrm{cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda) \overset{\mathrm{def}}{=} \Pr[\mathbf{Exprmt}^{\mathrm{cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda, \mathcal{U}) \text{ returns } \textsc{Win}].$$

`PP-ABID` is called *secure against cMiM attacks* if, for any PPT $\mathcal{A}$ and for any attribute universe $\mathcal{U}$, $\mathbf{Adv}^{\mathrm{cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda)$ is negligible in $\lambda$.

**Selective Security.** In the *selective game on a target attribute set* (the game of sel-cMiM attack), $\mathcal{A}$ declares $S^*$ *before* $\mathcal{A}$ receives PK. $\mathbf{Exprmt}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda, \mathcal{U})$ defines the selective game. The *advantage* in the game of sel-cMiM attack is defined as

$$\mathbf{Adv}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda) \overset{\mathrm{def}}{=} \Pr[\mathbf{Exprmt}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda, \mathcal{U}) \text{ returns } \textsc{Win}].$$

`PP-ABID` is called *selectively secure against cMiM attacks* if, for any PPT $\mathcal{A}$ and for any $\mathcal{U}$, $\mathbf{Adv}^{\mathrm{sel\text{-}cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda)$ is negligible in $\lambda$.

## 4 Generic Conversions from ABKEM to ABID

In this section, we provide a generic conversion from a key-policy ABKEM to a prover-policy ABID. The conversion yields a challenge-and-response protocol of encapsulation-and-decapsulation. We show that KP-ABKEM only has to be OW-CCA secure for the obtained PP-ABID to be cMiM secure. A generic conversion from a ciphertext-policy ABKEM to a verifier-policy ABID is provided in a similar way (in the full version).

### 4.1 Generic Conversion from KP-ABKEM to PP-ABID

Let `KP-ABKEM`= (KEM.Setup, KEM.KeyGen, KEM.Encap, KEM.Decap) be a KP-ABKEM. Then `PP-ABID`= (Setup, KeyGen, Encap, Decap) is obtained as a challenge-and-response protocol of encapsulation-and-decapsulation. Figure 3 shows this conversion. Setup of `PP-ABID` uses KEM.Setup. KeyGen of `PP-ABID` uses KEM.KeyGen. The verifier V, given a public key PK and an attribute set $S$ as input, invokes the encapsulation algorithm KEM.Encap on $(\mathrm{PK}, S)$. V gets a return $(\kappa, \psi)$. V sends the encapsulation $\psi$ to the prover $P$ as a challenge message. $P$, given a public key PK and the secret key $\mathrm{SK}_{\mathbb{A}}$ as input, and receiving $\psi$ as a message, invokes the decapsulation algorithm KEM.Decap on $(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}}, \psi)$. P gets a return $\hat{\kappa}$. P sends the decapsulation $\hat{\kappa}$ to V as a response message. Finally, V, receiving $\hat{\kappa}$ as a message, verifies whether $\hat{\kappa}$ is equal to $\kappa$. If so, then V returns 1 and otherwise, 0.

```
$\mathbf{Setup}(\lambda, \mathcal{U})$:                              $\mathbf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A})$:
  $(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{KEM.Setup}(\lambda, \mathcal{U})$      $\mathrm{SK}_{\mathbb{A}} \leftarrow \mathbf{KEM.KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A})$
  Return $(\mathrm{PK}, \mathrm{MSK})$                              Return $(\mathrm{SK}_{\mathbb{A}})$
--------------------------------------------------------------------------------
$\mathbf{P}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}})$:                              $\mathbf{V}(\mathrm{PK}, S)$:
                                                  $(\kappa, \psi) \leftarrow \mathbf{KEM.Encap}(\mathrm{PK}, S)$
  Receiving $\psi$ as input:              $\xleftarrow{\psi}$      Send $\psi$ to $\mathbf{P}$
  $\hat{\kappa} \leftarrow \mathbf{KEM.Decap}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}}, \psi)$
  Send $\hat{\kappa}$ to $\mathbf{V}$                    $\xrightarrow{\hat{\kappa}}$      Receiving $\hat{\kappa}$ as input:
                                                  If $\hat{\kappa} = \kappa$ then $b := 1$ else $b := 0$, Return $b$
```

**Fig. 3.** A generic conversion from KP-ABKEM to PP-ABID

**Theorem 1.** *If KP-ABKEM is OW-CCA secure, then the derived PP-ABID is cMiM secure. More precisely, for any given PPT adversary $\mathcal{A}$ on PP-ABID in the game of cMiM attack, and for any given attribute universe $\mathcal{U}$, there exists a PPT adversary $\mathcal{B}$ on KP-ABKEM in the OW-CCA game that satisfies the following tight reduction.*

$$\mathbf{Adv}^{cmim}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda) \leqslant \mathbf{Adv}^{ow\text{-}cca}_{\mathcal{B},\mathtt{KP\text{-}ABKEM}}(\lambda).$$

*Proof.* Employing any given PPT cMiM adversary $\mathcal{A}$ on PP-ABID in Theorem 1, we construct a PPT OW-CCA adversary $\mathcal{B}$ on KP-ABKEM. The left side of Figure 4 shows the construction.

On input $(\mathrm{PK}, \mathcal{U})$, $\mathcal{B}$ initializes its inner state and invokes $\mathcal{A}$ on $(\mathrm{PK}, \mathcal{U})$. When $\mathcal{A}$ issues a key-extraction query for $\mathbb{A}$, $\mathcal{B}$ queries its key-generation oracle $\mathcal{KG}(\mathrm{PK}, \mathrm{MSK}, \cdot)$ for the answer for $\mathbb{A}$ and gets a reply $\mathrm{SK}_{\mathbb{A}}$. $\mathcal{B}$ reply $\mathrm{SK}_{\mathbb{A}}$ to $\mathcal{A}$. When $\mathcal{A}$ sends a challenge message $(\mathbb{A}, \psi)$ to a prover $\mathbf{P}$, $\mathcal{B}$ queries its decapsulation oracle $\mathcal{DEC}(\mathrm{PK}, \mathrm{SK}_{\cdot}, \cdot)$ for the answer for $(\mathbb{A}, \psi)$ and gets a reply $\hat{\kappa}$. $\mathcal{B}$ reply $\hat{\kappa}$ to $\mathcal{A}$. When $\mathcal{A}$ outputs a target attribute set $S^*$, $\mathcal{B}$ output $S^*$ as its target attribute set. Then $\mathcal{B}$ receives a challenge ciphertext $\psi^*$ from its challenger. When $\mathcal{A}$ queries $\mathbf{V}$ for a challenge message, $\mathcal{B}$ sends $\psi^*$ to $\mathcal{A}$ as a challenge message. When $\mathcal{A}$ sends the response message $\hat{\kappa^*}$ to $\mathbf{V}$, $\mathcal{B}$ returns $\hat{\kappa^*}$ as its guess.

The view of $\mathcal{A}$ in $\mathcal{B}$ is the same as the real view of $\mathcal{A}$. If $\mathcal{A}$ wins, then $\mathcal{B}$ wins. Hence the inequality in Theorem 1 holds. □

## 4.2 Discussion

**Selective Security.** The right side of Figure 4 shows the construction of $\mathcal{B}$ in the game of selective $S^*$. The inequality of advantages becomes

$$\mathbf{Adv}^{\text{sel-cmim}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda) \leqslant \mathbf{Adv}^{\text{ow-sel-cca}}_{\mathcal{B},\mathtt{KP\text{-}ABKEM}}(\lambda).$$

**Resettable Security.** We note that the derived PP-ABID is prover-resettable in the sense in [4] because underlying KP-ABKEM has the OW-CCA security. PP-ABID is also verifier-resettable because PP-ABID consists of two rounds interaction.

```
$\mathcal{B}$(PK,$\mathcal{U}$): //Adaptive $S^*$
//**Set up**
  Initialize inner state, Invoke $\mathcal{A}$ on (PK,$\mathcal{U}$)
//**Answering $\mathcal{A}$'s Queries**
  When $\mathcal{A}$ issues a key-ext. query for $\mathbb{A}$
      $\mathrm{SK}_{\mathbb{A}} \leftarrow \mathcal{KG}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A})$
      Reply $\mathrm{SK}_{\mathbb{A}}$ to $\mathcal{A}$
  When $\mathcal{A}$ sends a chal. msg. $(\mathbb{A}, \psi)$ to **P**
      $\hat{\kappa} \leftarrow \mathcal{DEC}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}}, \psi)$
      Send $\hat{\kappa}$ to $\mathcal{A}$ as the res. msg.
  When $\mathcal{A}$ outputs a target attribute set $S^*$
      Output $S^*$ as its target attribute set
      Receive $\psi^*$ as a chal. ciphertext
  When $\mathcal{A}$ queries **V** for a chal. msg.
      Send $\psi^*$ to $\mathcal{A}$ as a chal. msg.
  When $\mathcal{A}$ sends the res. msg. $\hat{\kappa^*}$ to **V**
      Return $\hat{\kappa^*}$
```

```
$\mathcal{B}$(PK,$\mathcal{U}$): //Selective $S^*$
//**Set up**
  Initialize inner state, Invoke $\mathcal{A}$ on ($\lambda$,$\mathcal{U}$)
//**Answering $\mathcal{A}$'s Queries**
  $S^* \leftarrow \mathcal{A}(\lambda, \mathcal{U})$
  Output $S^*$ as its target attribute set
  Receive $\psi^*$ as a chal. ciphertext
  Give PK to $\mathcal{A}$
  When $\mathcal{A}$ issues a key-ext query for $\mathbb{A}$
      $\mathrm{SK}_{\mathbb{A}} \leftarrow \mathcal{KG}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A})$
  When $\mathcal{A}$ sends a chal. msg. $(\mathbb{A}, \psi)$ to **P**
      $\hat{\kappa} \leftarrow \mathcal{DEC}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}}, \psi)$
      Send $\hat{\kappa}$ to $\mathcal{A}$ as the res. msg.
  When $\mathcal{A}$ queries **V** for a chal. msg.
      Send $\psi^*$ to $\mathcal{A}$ as a chal. msg.
  When $\mathcal{A}$ sends the res. msg. $\hat{\kappa^*}$ to **V**
      Return $\hat{\kappa^*}$
```

**Fig. 4.** A one-way CCA adversary $\mathcal{B}$ on KP-ABKEM that employs a given cMiM adversary $\mathcal{A}$ on the PP-ABID. The left side: the case of adaptive $S^*$; the right side: the case of selective $S^*$.

## 5 Concrete Constructions of ABKEM

In this section, we construct a concrete KP-ABKEM that is OW-sel-CCA secure. Using the algebraic trick of Boneh and Boyen [3] and Kiltz [11], we build an enhanced version, KP-ABKEM, of the KP-ABKEM of Ostrovsky, Sahai and Waters [13] (OSW, for short). Then we obtain our concrete PP-ABID by applying the generic conversion. (Our concrete CP-ABKEM and VP-ABID is described in Appendix C).

### 5.1 Our Enhanced OSW KP-ABKEM and PP-ABID

The construction of our concrete KP-ABKEM is described in Figure 5. We only explain the enhanced part from the original [13]. We indicate the part of the original scheme by the index: $_{\mathrm{cpa}}$. In Setup, a second component $\alpha_2 \in \mathbb{Z}_p$ is added to the master secret key $\mathrm{MSK}_{\mathrm{cpa}}$. Also, the corresponding $Y_2 := e(g,g)^{\alpha_2 b}$ and a hash key $\eta$ is added to the public key $\mathrm{PK}_{\mathrm{cpa}}$. In KeyGen, components in $\mathrm{SK}_{\mathrm{cpa},\mathbb{A}}$ are doubled reflecting the index 2 (but randomness is chosen independently of index 1). So computational cost for KeyGen is doubled. In Encap, a temporal KEM key $\kappa_2$ is generated in the same way as $\kappa_1$. Next, a hash value $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$ and a *check sum* $d := \kappa_1^\tau \kappa_2$ are computed. Then $(\kappa, \psi) := (\kappa_1, (\psi_{\mathrm{cpa}}, d))$ is a new KEM key and encapsulation. In Decap, first, $\mathrm{Decap}_{\mathrm{cpa}}$ is executed twice for index 1 and 2 to yield $\hat{\kappa_1}$ and $\hat{\kappa_2}$. Then, whether $\psi_{\mathrm{cpa}}$ is a consistent ciphertext and $(e(g,g), Y_1^\tau Y_2, e(C',g), d)$ is a Diffie-Hellman tuple is verified. These two conditions are verified by one equation $\hat{\kappa_1}^\tau \hat{\kappa_2} = d$, though the verification equation overlooks inconsistent $\psi_{\mathrm{cpa}}$ only with a negligible probability. Finally, $\hat{\kappa} := \hat{\kappa_1}$ is returned only when the verification equation holds.

```
$\mathbf{Setup}(\lambda, \mathcal{U})$:
    $(p, \mathbb{G}, \mathbb{G}_T, g, e) \leftarrow \mathbf{Grp}(\lambda)$
    For $x = 1$ to $u$: $T_x \leftarrow \mathbb{G}$
    $b \leftarrow \mathbb{Z}_p, B := g^b, \alpha_1, \alpha_2 \leftarrow \mathbb{Z}_p$
    $Y_1 := e(g,g)^{\alpha_1 b}, Y_2 := e(g,g)^{\alpha_2 b}$
    $\eta \leftarrow HKey_\lambda$
    $\mathrm{PK} := (g, T_1, \ldots, T_u, B, Y_1, Y_2, \eta)$
    $\mathrm{MSK} := (\alpha_1, \alpha_2)$
    Return $(\mathrm{PK}, \mathrm{MSK})$
```

```
$\mathbf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A} = (M, \rho))$:
    For $k = 1, 2$: For $j = 2$ to $n$: $v_{k,j} \leftarrow \mathbb{Z}_p$
    For $k = 1, 2$: $\boldsymbol{v}_k := (\alpha_k, v_{k,2}, \ldots, v_{k,n})$
    For $i = 1$ to $l$: $\lambda_{k,i} := \boldsymbol{v}_k \cdot M_i$
    For $k = 1, 2$: For $i = 1$ to $l$:
        $r_{k,i} \leftarrow \mathbb{Z}_p,\ K_{k,i} := B^{\lambda_{k,i}} T_{\rho(i)}^{r_{k,i}}$,
                $L_{k,i} := g^{r_{k,i}}$
    $\mathrm{SK}_{\mathbb{A}} := (((K_{k,i}, L_{k,i}); i = 1, \ldots, l); k = 1, 2)$
    Return $\mathrm{SK}_{\mathbb{A}}$
```

```
$\mathbf{Encap}(\mathrm{PK}, S)$:
    $s \leftarrow \mathbb{Z}_p, C' = g^s$, For $x \in S : C_x = T_x^s$
    $\psi_{\mathrm{cpa}} := (S, C', (C_x; x \in S))$
    $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
    For $k = 1, 2$: $\kappa_k := Y_k^s;\ d := \kappa_1^\tau \kappa_2$
    $(\kappa, \psi) := (\kappa_1, (\psi_{\mathrm{cpa}}, d))$
    Return $(\kappa, \psi)$
```

```
$\mathbf{Decap}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}}, \psi)$:
    If $S \notin \mathbb{A}$ Return $\hat{\kappa} := \perp$
    else $I_S := \rho^{-1}(S)$
     $\{\omega_i; i \in I_S\} \leftarrow \mathbf{Recon}(I_S, M)$
     For $k = 1, 2$:
      $\hat{\kappa_k} := \prod_{i \in I_S} (e(K_{k,i}, C') / e(L_{k,i}, C_{\rho(i)}))^{\omega_i}$
     $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
     If $\hat{\kappa_1}^\tau \hat{\kappa_2} \neq d$ then $\hat{\kappa} := \perp$ else $\hat{\kappa} := \hat{\kappa_1}$
     Return $\hat{\kappa}$
```

**Fig. 5.** Our concrete `KP-ABKEM` (an enhanced OSW KP-ABKEM)

**Theorem 2.** *If the computational bilinear Diffie-Hellman assumption with gap on target group holds, and an employed hash function family has target collision resistance, then our* `KP-ABKEM` *is OW-sel-CCA secure. More precisely, for any given PPT adversary $\mathcal{A}$ on* `KP-ABKEM` *in the OW-sel-CCA game and for any given attribute universe $\mathcal{U}$, there exist a PPT adversary $\mathcal{B}$ on $(e, \mathbb{G}, \mathbb{G}_T)$ in the computational BDH game with gap on $\mathbb{G}_T$ and a PPT target collision finder $\mathcal{CF}$ on $Hfam_\lambda$ that satisfy the following tight reduction.*

$$\mathbf{Adv}^{ow\text{-}sel\text{-}cca}_{\mathcal{A},\mathtt{KP\text{-}ABKEM}}(\lambda) \leqslant \mathbf{Adv}^{c\text{-}bdh\text{-}gap}_{\mathcal{B},(e,\mathbb{G},\mathbb{G}_T)}(\lambda) + \mathbf{Adv}^{tcr}_{\mathcal{CF},Hfam_\lambda}(\lambda).$$

## 5.2 Proof for Theorem 2

Using any given OW-sel-CCA adversary $\mathcal{A}$ as a subroutine, we construct a PPT solver $\mathcal{B}$ of the problem of the computational bilinear Diffie-Hellman assumption with gap on target group, as follows.

**Set Up.** $\mathcal{B}$ is given a random instance of the problem, $g, A = g^a, B = g^b, C = g^c$, as input. $\mathcal{B}$ initializes its inner state. $\mathcal{B}$ chooses an attribute universe $\mathcal{U} = \{1, \ldots, u\}$ at random. $\mathcal{B}$ invokes $\mathcal{A}$ on input $(\lambda, \mathcal{U})$.

In return, $\mathcal{B}$ receives a target attribute set $S^* \in \mathcal{U}$ from $\mathcal{A}$, For each $x = 1, \ldots, u$, $\mathcal{B}$ puts each component $T_x$ of PK as

If $x \in S^*$ then $t_x \leftarrow \mathbb{Z}_p, T_x := g^{t_x}$ else $\theta_x, \eta_x \leftarrow \mathbb{Z}_p$ s.t. $\theta_x \neq 0, T_x := B^{\theta_x} g^{\eta_x}$.

Here, in else case, we have implicitly set $t_x := b\theta_x + \eta_x$. $\mathcal{B}$ sets $Y_1 := e(A, B) = e(g,g)^{ab}$ and $\mathrm{PK}_{\mathrm{cpa}} := (g, T_1, \ldots, T_u, B, Y_1)$. Here we have implicitly set $\alpha_1 := a$.

Challenge ciphertext are computed as follows (we implicitly set $s^* = c$):

$$\psi^*_{\text{cpa}} := (S^*, C'^* = g^{s^*} := C, (C^*_x := C^{t_x}, x \in S^*)).$$

Then a public key PK and a whole challenge ciphertext $\psi^*$ is computed as

$$\eta \leftarrow HKey_\lambda,\ \tau^* \leftarrow H_\eta(\psi^*_{\text{cpa}}),\ \mu \leftarrow \mathbb{Z}_p,\ Y_2 := e(B,g)^\mu / Y_1^{\tau^*},$$
$$\text{PK} := (\text{PK}_{\text{cpa}}, Y_2, \eta),\ d^* := e(B, C'^*)^\mu,\ \psi^* := (\psi^*_{\text{cpa}}, d^*).$$

Here we have an implicit relation $\alpha_2 b = b\mu - \alpha_1 b \tau^*, b \neq 0$. That is,

$$\alpha_2 = \mu - \alpha_1 \tau^*. \tag{1}$$

$\mathcal{B}$ inputs $(\text{PK}, \psi^*)$ to $\mathcal{A}$.

**Answering $\mathcal{A}$'s Queries. (1) Key-Extraction Queries.** When $\mathcal{A}$ issues a key-extraction query for an attribute set $\mathbb{A} = (M, \rho)$, where $M$ is of size $l \times n$, $\mathcal{B}$ has to reply a corresponding secret key $\text{SK}_{\mathbb{A}}$.

$\mathcal{B}$ computes a vector $\boldsymbol{w} = (w_1, \ldots, w_n) \in \mathbb{Z}_p^n$ such that $w_1 = 1$ and for all $i \in \rho^{-1}(S^*), \boldsymbol{w} \cdot M_i = 0$. Note here that $S^* \notin \mathbb{A}$, so such $\boldsymbol{w}$ surely exists. $\mathcal{B}$ chooses random values $u_{1,1}, \ldots, u_{1,n} \in \mathbb{Z}_p$ and put $\boldsymbol{u}_1 := (u_{1,1}, \ldots, u_{1,n})$. Then we implicitly set $\boldsymbol{v}_1 := \boldsymbol{u}_1 + (a - u_{1,1})\boldsymbol{w}$.

Here for each $i = 1, \ldots, l$, $\mathcal{B}$ can compute $g^{\lambda_{1,i}}$ as $g^{M_i \cdot \boldsymbol{v}_1} = g^{M_i \cdot (\boldsymbol{u}_1 - u_{1,1}\boldsymbol{w})} A^{M_i \cdot \boldsymbol{w}}$. Then $\mathcal{B}$ computes the index 1 components of $\text{SK}_S$ as

```
For $i = 1$ to $l$ : If $i \in \rho^{-1}(S^*)$ then $r_{1,i} \leftarrow \mathbb{Z}_p, K_{1,i} := B^{M_i \cdot \boldsymbol{u}_1} T_{\rho(i)}^{r_{1,i}}, L_{1,i} := g^{r_{1,i}}$
  else $r'_{1,i} \leftarrow \mathbb{Z}_p, K_{1,i} := (g^{\lambda_{1,i}})^{-\eta_{\rho(i)}/\theta_{\rho(i)}} (B^{\theta_{\rho(i)}} g^{\eta_{\rho(i)}})^{r'_{1,i}}, L_{1,i} := (g^{\lambda_{1,i}})^{-1/\theta_{\rho(i)}} g^{r'_{1,i}}$.
```

Here, in else case, we implicitly set $r_{1,i} := r'_{1,i} - \lambda_{1,i}/\theta_{\rho(i)}$.

Now $\mathcal{B}$ has to compute the index 2 components $K_{2,i}, L_{2,i}$ for $i = 1, \ldots, l$. To do so, $\mathcal{B}$ chooses random values $u_{2',1}, \ldots, u_{2',n}, r_{2',i}$(or $r'_{2',i}$) $\in \mathbb{Z}_p$ and computes $K_{2',i}, L_{2',i}, i = 1, \ldots, l$ *just in the same way* as to the index 1. Then $\mathcal{B}$ converts them as follows:

$$K_{2,i} := B^{M_{i,1}\mu} (K_{2',i})^{-\tau^*}, L_{2,i} := (L_{2',i})^{-\tau^*}, i = 1, \ldots, l.$$

Then $\mathcal{B}$ replies $\text{SK}_{\mathbb{A}} = (((K_{k,i}, L_{k,i}); i = 1, \ldots, l); k = 1, 2)$ to $\mathcal{A}$.

**(2) Decapsulation Queries.** When $\mathcal{A}$ issues a decapsulation query for $(\mathbb{A}, \psi = (\psi_{\text{cpa}}, d))$ (where $\psi_{\text{cpa}}$ is about $S$), $\mathcal{B}$ has to reply the decapsulation $\hat{\kappa}$ to $\mathcal{A}$. To do so, $\mathcal{B}$ computes as follows. (Note that the oracle $\mathcal{DDH}_{\mathbb{G}_T}$ is accessed.)

```
If $S \notin \mathbb{A}$ then $\hat{\kappa} :=\perp$
else If $\textbf{Verify}(\text{PK}_{\text{cpa}}, \psi_{\text{cpa}}) = \text{FALSE}$ then $\hat{\kappa} :=\perp$
    else $\tau \leftarrow H_\eta(\psi_{\text{cpa}})$ If $\mathcal{DDH}_{\mathbb{G}_T}(e(g,g), Y_1^\tau Y_2, e(C',g), d) = \text{FALSE}$ then $\hat{\kappa} :=\perp$
                                 else If $\tau = \tau^*$ then Abort //Call this case ABORT
                                     else $\hat{\kappa} := (d / e(B, C')^\mu)^{1/(\tau - \tau^*)}$
```

where **Verify** is the following PPT algorithm to check consistency of $\psi_{\mathrm{cpa}}$:

$$\textbf{Verify}(\mathrm{PK}_{\mathrm{cpa}}, \psi_{\mathrm{cpa}}) : \text{For } x \in S : \text{ If } e(T_x, C') \neq e(C_x, g) \text{ then Return } \textsc{False}$$
$$\text{Return } \textsc{True}.$$

**Guess.** When $\mathcal{A}$ returns $\mathcal{A}$'s guess $\hat{\kappa^*}$, $\mathcal{B}$ returns $Z := \hat{\kappa^*}$ as $\mathcal{B}$'s guess.

$\mathcal{B}$ can perfectly simulate the real view of $\mathcal{A}$ until the case Abort happens. To see why, we prove the following claims. (The proofs will be in the full version.)

**Claim 1.** *The reply $SK_{\mathbb{A}}$ to a key-extraction query is a perfect simulation.* □

**Claim 2.** *The reply $\hat{\kappa}$ to a decapsulation query is a simulation that is computationally indistinguishable from a real, until the case* Abort *happens.* □

**Claim 3.** *The challenge ciphertext $\psi^* = (\psi^*_{cpa}, d^*)$ is correctly distributed.* □

Now we are ready to evaluate the advantage of $\mathcal{B}$ in the OW-sel-CCA game. First, the following claim holds. (The proof will be described in the full version.)

**Claim 4.** *The probability that* Abort *occurs is negligible in $\lambda$. More precisely, the following equality holds: $Pr[\textsc{Abort}] = \mathbf{Adv}^{tcr}_{\mathcal{CF},Hfam_\lambda}(\lambda)$.* □

By definition, $\mathcal{A}$ wins in the OW-sel-CCA game if and only if $\hat{\kappa}^*$ is correctly guessed. That is, $\hat{\kappa^*} = Y_1^{s^*} = e(g,g)^{abs^*} = e(g,g)^{abc}$. This is the definition that $\mathcal{B}$ succeeds in computing the answer for the given instance $(g, A, B, C)$.

Therefore, the probability that $\mathcal{B}$ wins is equal to the probability that $\mathcal{A}$ wins and Abort never occurs. So we have:

$$\Pr[\mathcal{B} \text{ wins}] = \Pr[(\mathcal{A} \text{ wins}) \wedge (\neg\textsc{Abort})] \geqslant \Pr[\mathcal{A} \text{ wins}] - \Pr[\textsc{Abort}].$$

Substituting advantages and using the equality in Claim 4, we have:

$$\mathbf{Adv}^{\text{c-bdh-gap}}_{\mathcal{B},(e,\mathbb{G},\mathbb{G}_T)}(\lambda) \geqslant \mathbf{Adv}^{\text{ow-sel-cca}}_{\mathcal{A},\texttt{KP-ABKEM}}(\lambda) - \mathbf{Adv}^{\text{tcr}}_{\mathcal{CF},Hfam_\lambda}(\lambda).$$

This is what we should prove. □

**Theorem 3 (Corollary to Theorem 1 and 2).** *Our* `PP-ABID` *is selectively secure against cMiM attacks. More precisely, the following inequality holds:*

$$\mathbf{Adv}^{sel\text{-}cmim}_{\mathcal{A},\texttt{PP-ABID}}(\lambda) \leqslant \mathbf{Adv}^{c\text{-}bdh\text{-}gap}_{\mathcal{B},(e,\mathbb{G},\mathbb{G}_T)}(\lambda) + \mathbf{Adv}^{tcr}_{\mathcal{CF},Hfam_\lambda}(\lambda).$$

Figure 6 shows an interaction of our `PP-ABID`.

```
$\mathbf{P}(\mathrm{PK} = (g, T_1, \ldots, T_u, B, Y_1, Y_2, \eta),$                    $\mathbf{V}(\mathrm{PK}, S)$:
    $\mathrm{SK}_{\mathbb{A}} = (((K_{k,i} = B^{\lambda_{k,i}} T_{\rho(i)}^{r_{k,i}},$              $s \leftarrow \mathbb{Z}_p, C' := g^s$, For $x \in S : C_x := T_x^s$
        $L_{k,i} = g^{r_{k,i}});$                                $\psi_{\mathrm{cpa}} := (S, C', (C_x; x \in S))$
        $i = 1, \ldots, l); k = 1, 2))$:                          $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
                                                            For $k = 1, 2$: $\kappa_k := Y_k^s$; $d := \kappa_1^\tau \kappa_2$
                                                            $(\kappa, \psi) := (\kappa_1, (\psi_{\mathrm{cpa}}, d))$
  Receiving $\psi$ as input:                          $\xleftarrow{\psi}$  Send $\psi$ to $\mathbf{P}$
  If $S \notin \mathbb{A}$ then $\hat{\kappa} := \perp$
  else $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}}), I_S := \rho^{-1}(S)$
    $\{\omega_i; i \in I_S\} \leftarrow \mathbf{Recon}(I_S, M)$
    For $k = 1, 2$:
      $\hat{\kappa_k} := \prod_{i \in I_S} (e(K_{K,i}, C')$
        $/e(L_{K,i}, C_{\rho(i)}))^{\omega_i}$
    If $\hat{\kappa_1}^\tau \hat{\kappa_2} \neq d$ then $\hat{\kappa} := \perp$ else $\hat{\kappa} := \hat{\kappa_1}$
  Send $\hat{\kappa}$ to $\mathbf{V}$                            $\xrightarrow{\hat{\kappa}}$  Receiving $\hat{\kappa}$ as input:
                                                            If $\hat{\kappa} = \kappa$ then $b := 1$ else $b := 0$, Return $b$
```

**Fig. 6.** An interaction of our concrete PP-ABID

## 5.3 Discussion

**Anonymity.** Consider the following experiment in Figure 7. (In the experiment, an adversary $\mathcal{A}$ interacts with $\mathbf{P}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}_b})$ as a verifier with $S^*$.)

We say that PP-ABID have *anonymity* if, for any PPT $\mathcal{A}$ and for any $\mathcal{U}$, $\mathbf{Adv}^{\mathrm{anonym}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda) \overset{\mathrm{def}}{=} |\Pr[\mathbf{Exprmt}^{\mathrm{anonym}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda, \mathcal{U})$ returns WIN$] - 1/2|$ is negligible in $\lambda$. Our concrete PP-ABID possesses the anonymity because, in the case that $\psi$ is inconsistent, the randomness in $\mathrm{SK}_{\mathbb{A}_b}$ is *not canceled out correctly* in the computation of a response message $\hat{\kappa_b}$.

```
$\mathbf{Exprmt}^{\mathrm{anonym}}_{\mathcal{A},\mathtt{PP\text{-}ABID}}(\lambda, \mathcal{U})$:
  $(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda, \mathcal{U}), (\mathbb{A}_0, \mathbb{A}_1, S^*) \leftarrow \mathcal{A}(\mathrm{PK})$
            s.t. $(S^* \in \mathbb{A}_0 \wedge S^* \in \mathbb{A}_1) \vee (S^* \notin \mathbb{A}_0 \wedge S^* \notin \mathbb{A}_1)$
  $\mathrm{SK}_{\mathbb{A}_0} \leftarrow \mathbf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A}_0), \mathrm{SK}_{\mathbb{A}_1} \leftarrow \mathbf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A}_1)$
  $b \leftarrow \{0, 1\}, \hat{b} \leftarrow \mathcal{A}^{\mathbf{P}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}_b})}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}_0}, \mathrm{SK}_{\mathbb{A}_1})$
  If $b = \hat{b}$ Return WIN else Return LOSE
```

**Fig. 7.** The anonymity experiment of an adversary $\mathcal{A}$ on PP-ABID

**Large Universe Case.** If the attribute universe $\mathcal{U}$ is large, we have to modify our concrete schemes to make security reductions in time polynomial in $\lambda$. As is proposed by Waters [15], we use for $x \in \mathcal{U}$ a hashed value $H(x)$ instead of $T_x$ (and hence $T_x$ is removed from PK). Although the resulting schemes are proved to be secure only in the random oracle model, we get relief from rewriting the public key PK each time when a new attribute $x$ is added.

**Exiting the Gap Assumption.** Instead of the oracle $\mathcal{DDH}_{\mathbb{G}_T}$, we can apply the twin Diffie-Hellman trapdoor test of Cash, Kiltz and Shoup [8] in the security proofs. In compensation, the resulting schemes become to have a twice as long secret key and twice as much computational cost in decapsulation.

**Security against Adaptive Target.** To attain the adaptive security in the OW-CCA game, we can apply our enhancing technique to the dual system encryption of Lewko, Okamoto, Sahai, Takashima and Waters [10] in the random oracle model.

## 6 Conclusions

We proposed PP-ABID and VP-ABID. We established a design principle. We constructed concrete KP-ABKEM and CP-ABKEM with the OW-CCA security. Finally, we obtained concrete PP-ABID and VP-ABID. Functional tickets and functional gates are realized by those PP-ABID and VP-ABID, respectively.

## References

1. Anada, H., Arita, S.: Identification Schemes from Key Encapsulation Mechanisms. In: Nitaj, A., Pointcheval, D. (eds.) AFRICACRYPT 2011. LNCS, vol. 6737, pp. 59–76. Springer, Heidelberg (2011)
2. Beimel, A.: Secure schemes for secret sharing and key distribution. Ph.D. thesis, Israel Institute of Technology, Technion, Haifa, Israel (1996)
3. Boneh, D., Boyen, X.: Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)
4. Bellare, M., Fischlin, M., Goldwasser, S., Micali, S.: Identification Protocols Secure against Reset Attacks. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 495–511. Springer, Heidelberg (2001)
5. Bellare, M., Palacio, A.: GQ and Schnorr Identification Schemes: Proofs of Security against Impersonation under Active and Concurrent Attacks. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 162–177. Springer, Heidelberg (2002)
6. 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)
7. Cramer, R., Damgård, I., Nielsen, J.B.: Multiparty Computation from Threshold Homomorphic Encryption. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 280–300. Springer, Heidelberg (2001)
8. Cash, D., Kiltz, E., Shoup, V.: The Twin Diffie-Hellman Problem and Applications. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 127–145. Springer, Heidelberg (2008); Full version available at Cryptology ePrint Archive, 2008/067, `http://eprint.iacr.org/`
9. Gennaro, R.: Multi-trapdoor Commitments and their Applications to Proofs of Knowledge Secure Under Concurrent Man-in-the-Middle Attacks. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 220–236. Springer, Heidelberg (2004)
10. 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); Full version available at IACR Cryptology ePrint Archive, 2010/110, `http://eprint.iacr.org/`

11. Kiltz, E.: Chosen-Ciphertext Security from Tag-Based Encryption. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 581–600. Springer, Heidelberg (2006)
12. Naor, M., Yung, M.: Universal One-Way Hash Functions and their Cryptographic Applications. In: The 21st Symposium on Theory of Computing 1989, pp. 33–43. Association for Computing Machinery, New York (1989)
13. Ostrovsky, R., Sahai, A., Waters, B.: Attribute-based encryption with non-monotonic access structures. In: ACM Conference on Computer and Communications Security, pp. 195–203 (2007)
14. Schnorr, C.P.: Efficient Identification and Signatures for Smart Cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer, Heidelberg (1990)
15. 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), Full version available at IACR Cryptology ePrint Archive,2008/290, `http://eprint.iacr.org/`
16. Yamada, S., Attrapadung, N., Santoso, B., Schuldt, J.C.N., Hanaoka, G., Kunihiro, N.: Verifiable Predicate Encryption and Applications to CCA Security and Anonymous Predicate Authentication. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 243–261. Springer, Heidelberg (2012)
17. Yilek, S.: Resettable Public-Key Encryption: How to Encrypt on a Virtual Machine. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 41–56. Springer, Heidelberg (2010)

## A Verifier-Policy ABID

We define a notion of *verifier-policy* attribute-based identification (VP-ABID).

**Scheme.** `VP-ABID` consists of four PPT algorithms (Setup, KeyGen, P, V).

**Setup**$(\lambda, \mathcal{U}) \to$ **(PK, MSK).** Setup takes as input the security parameter $\lambda$ and the attribute universe $\mathcal{U}$. It outputs public a public key PK and a master secret key MSK.

**KeyGen**$(\mathbf{PK}, \mathbf{MSK}, S) \to \mathbf{SK}_S$**.** A key-generation algorithm KeyGen takes as input the public key PK, the master secret key MSK and an attribute set $S$. It outputs a secret key $\mathrm{SK}_S$ corresponding to $S$.

$\mathbf{P}(\mathbf{PK}, \mathbf{SK}_S)$ **and** $\mathbf{V}(\mathbf{PK}, \mathbb{A})$**.** P and V are interactive algorithms called a *prover* and a *verifier*, respectively, which are defined in a similar way as in Section 3.1. A cMiM attack on `VP-ABID` is defined in a similar way as in Section 3.2. `CP-ABKEM`=(**Setup**,**KeyGen**,**Encap**,**Decap**) is defined in a dual manner to `KP-ABKEM` on an access structure $\mathbb{A}$ and an attribute set $S$. Then a generic conversion from `CP-ABKEM` to `VP-ABID` is defined in a similar way as in Section 4.1.

**Theorem 4.** *If `CP-ABKEM` is OW-CCA secure, then the derived `VP-ABID` is cMiM secure.*

## B Computational Parallel Bilinear Diffie-Hellman Exponent Assumption with Gap on Target Group

Let $a, s, b_1, \ldots, b_q \in \mathbb{Z}_p$, all of which is not zero, be chosen at random. Let $\boldsymbol{y} := (g, g^s, g^a, \ldots, g^{(a^q)}, g^{(a^{q+2})}, \ldots, g^{(a^{2q})}, \forall_{1 \leqslant j \leqslant q} g^{sb_j}, g^{a/b_j}, \ldots, g^{(a^q/b_j)}, g^{(a^{q+2}/b_j)}, \ldots, g^{(a^{2q}/b_j)}, \forall_{1 \leqslant j,k \leqslant q, k \neq j} g^{asb_k/b_j}, \ldots, g^{a^q sb_k/b_j})$. Then our new assumption says it is at most with a negligible probability in $\lambda$ that, for any PPT algorithm $\mathcal{B}$ given input $\boldsymbol{y}$ (parametrized by $q$), to output $Z = e(g,g)^{a^{q+1}s} \in \mathbb{G}_T$, even with the aid of the oracle $\mathcal{DDH}_{\mathbb{G}_T}(\cdot,\cdot,\cdot,\cdot)$.

## C Our Enhanced Waters CP-ABKEM and VP-ABID

We can build an enhanced version, CP-ABKEM, of the CP-ABKEM of Waters [15]. Then we can obtain our concrete VP-ABID as in Figure 8.

```
$\mathbf{P}(\mathrm{PK} = (g, T_1, \ldots, T_u, A, Y_1, Y_2, \eta),$
  $\mathrm{SK}_S = ((K_k = g^{\alpha_k} A^{l_k}, L_k = g^{l_k},$
   $(K_{k,x} = T_x^{l_k}; x \in \mathcal{S}); k = 1, 2))):$

$\mathbf{V}(\mathrm{PK}, \mathbb{A}):$
  $s \leftarrow \mathbb{Z}_p$, For $j = 2$ to $n$: $v_j \leftarrow \mathbb{Z}_p$,
                 $\boldsymbol{v} := (s, v_2, \ldots, v_n)$
  For $i = 1$ to $l$: $\lambda_i := \boldsymbol{v} \cdot M_i,\ r_i \leftarrow \mathbb{Z}_p$
  $C' := g^s$
  For $i = 1$ to $l$: $C_i := A^{\lambda_i} T_{\rho(i)}^{-r_i}, D_i := g^{r_i}$
  $\psi_{\mathrm{cpa}} := (\mathbb{A}, C', ((C_i, D_i); i = 1, \ldots, l))$
  $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}})$
  For $k = 1, 2$: $\kappa_k := Y_k^s; d := \kappa_1^\tau \kappa_2$
  $(\kappa, \psi) := (\kappa_1, (\psi_{\mathrm{cpa}}, d))$
  Send $\psi$ to $\mathbf{P}$                                $\xleftarrow{\psi}$

P:
  Receiving $\psi$ as input:
  If $S \notin \mathbb{A}$ then $\hat{\kappa} := \perp$
  else $\tau \leftarrow H_\eta(\psi_{\mathrm{cpa}}), I_S := \rho^{-1}(S)$
   $\{\omega_i; i \in I_S\} \leftarrow \mathbf{Recon}(I_S, M)$
   For $k = 1, 2$ : $\hat{\kappa_k} := e(K_k, C') /$
     $\prod_{i \in I_S} (e(L_k, C_i) e(K_{k,\rho(i)}, D_i))^{\omega_i}$
   If $\hat{\kappa_1}^\tau \hat{\kappa_2} \neq d$ then $\hat{\kappa} := \perp$ else $\hat{\kappa} := \hat{\kappa_1}$
  Send $\hat{\kappa}$ to $\mathbf{V}$                        $\xrightarrow{\hat{\kappa}}$

V:
  Receiving $\hat{\kappa}$ as input:
  If $\hat{\kappa} = \kappa$ then $b := 1$ else $b := 0$, Return $b$
```

**Fig. 8.** An interaction of our concrete VP-ABID

**Theorem 5.** *If the computational q-parallel bilinear Diffie-Hellman exponent assumption with gap on target group holds, and an employed hash function family has target collision resistance, then our CP-ABKEM is OW-sel-CCA secure with a challenge matrix of size $l^* \times n^*$, $l^*, n^* \leqslant q$.*

**Theorem 6 (Corollary to Theorem 4 and 5).** *Our VP-ABID is selectively secure against cMiM attacks.*

The proof of Theorem 5 will be described in the full version.

# Relations among Privacy Notions for Signcryption and Key Invisible “Sign-then-Encrypt”

Yang Wang[1], Mark Manulis[2], Man Ho Au[1], and Willy Susilo[1,⋆]

[1] Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Australia
`yw990@uowmail.uow.edu.au`,
`{wsusilo,aau}@uow.edu.au`

[2] Department of Computing, University of Surrey, United Kingdom
`mark@manulis.eu`

**Abstract.** Signcryption simultaneously offers authentication through unforgeability and confidentiality through indistinguishability against chosen ciphertext attacks by combining the functionality of digital signatures and public-key encryption into a single operation. Libert and Quisquater (PKC 2004) extended this set of basic requirements with the notions of ciphertext anonymity (or key privacy) and key invisibility to protect the identities of signcryption users and were able to prove that key invisibility implies ciphertext anonymity by imposing certain conditions on the underlying signcryption scheme.

This paper revisits the relationship amongst privacy notions for signcryption. We prove that key invisibility implies ciphertext anonymity without any additional restrictions. More surprisingly, we prove that key invisibility also implies indistinguishability against chosen ciphertext attacks. This places key invisibility on the top of privacy hierarchy for public-key signcryption schemes.

On the constructive side, we show that general “sign-then-encrypt” approach offers key invisibility if the underlying encryption scheme satisfies two existing security notions, indistinguishable against adaptive chosen ciphertext attacks and indistinguishability of keys against adaptive chosen ciphertext attacks. By this method we obtain the first key invisible signcryption construction in the standard model.

## 1 Introduction

*Signcryption methods.* The concept of signcryption was introduced by Zheng in 1997 [26], with the initial goal to achieve performance increase for simultaneous signing and public-key encryption. His idea was to derive the combined functionality by optimizing computations at the algorithmic level rather than considering joint execution of two different signing and encryption procedures.

⋆ W. Susilo is supported by ARC Future Fellowship FT0991397.

This idea was reflected in various signcryption constructions, including those based on discrete logarithms [4,21,25], factoring assumptions [18,22], and hard problems in groups with bilinear maps [15,16]. Some of these designs were less successful, e.g. [4,25] were cryptanalyzed in [21], a problem in [15] was identified in [23] and repaired in [9].

A more general approach to signcryption was initiated by An, Dodis, and Rabin [1]. They considered different methods for obtaining the signcryption functionality through a black-box composition of arbitrary signature and public-key encryption schemes, in particular showing that "encrypt-then-sign" (EtS) and "sign-then-encrypt" (StE) lead to secure singcryption schemes (as opposed to the symmetric-key setting [6]). They also introduced another approach, termed "commit-then-sign-and-encrypt" (CtS&E) that admits parallelization of the signing and encryption operations, motivated by the insecurity of the plain "sign-and-encrypt" (S&E) method. Dent et al. [10] recently proved security of S&E in the setting of high-entropy messages, assuming the confidentiality property of signatures. Alternative generic methods for (parallel) signcryption were introduced by Pieprzyk and Pointcheval [19] based on secret sharing techniques, by Dodis et al. [11] using trapdoor permutations and probabilistic padding schemes, and by Malone-Lee [17] from the hybrid KEM/DEM framework.

*Privacy notions for signcryption.* The first formal security model for signcryption in the public-key setting was introduced by Baek et al. [3], encompassing the requirements of message confidentiality (indistinguishability against adaptive chosen ciphertext attacks) and unforgeability against chosen-message attacks in the multi-user setting. This model has been strengthened by An, Dodis, and Rabin [1] towards the insider security setting that admits corruptions of senders and receivers, as opposed to the outsider security guarantees from [3] in which all involved parties must remain uncorrupted. The insider security setting became the de facto standard security setting for modern public-key signcryption schemes.

Libert and Quisquater [15], inspired by Boyen's work [7] on identity-based signcryption and the earlier definition of key privacy for public-key encryption schemes by Bellare et al. [5], formalized the notions of ciphertext anonymity (or key privacy) for public-key signcryption. This requirement, modeled within the insider security framework, prevents the adversary that is not in possession of the recipient's decryption key from obtaining information about the sender and the recipient of the signcrypted message. Libert and Quisquater also introduced the notion of key invisibility, for which they could prove that it implies ciphertext anonymity as long as signcryption ciphertexts have uniform distribution for random recipients' public keys.

## 1.1 Our Contribution

In this paper we focus on privacy notions for signcryption schemes and aim at closing gaps from previous work.

*Relations among privacy notions.* Using public-key signcryption notions from [15], namely key invisibility (SC-INVK-CCA), ciphertext anonymity (SC-INDK-CCA), and indistinguishability against chosen ciphertext attacks (SC-IND-CCA), we investigate their relationships and come to the following surprising results (cf. Figure 1): first, we show that key invisibility implies ciphertext anonymity without requiring uniformity of ciphertexts for random public keys (as opposed to the proof from [15]). Our proof of this implication involves a two-step approach: we first give a new definition of ciphertext anonymity, which we term SC-ANON-CCA and for which we prove the equivalence to SC-INDK-CCA from [15], before proving that SC-ANON-CCA is implied by SC-INVK-CCA. Even more surprising, we prove that SC-INVK-CCA implies SC-IND-CCA, that is key invisible signcryption schemes readily provide message confidentiality. Our analysis thus implies that key invisibility is strictly stronger than ciphertext anonymity and message confidentiality.

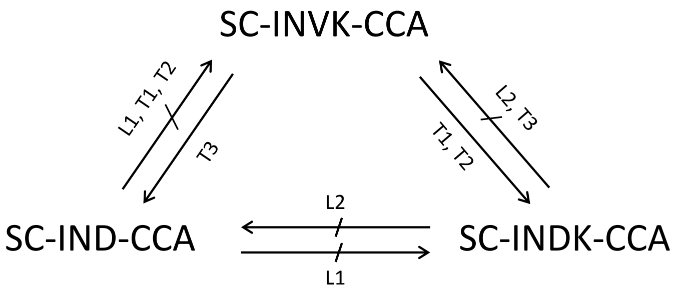


**Fig. 1. Relationships among privacy notions for signcryption.** An arrow denotes an implication while a barred arrow denotes a separation. T and L stand for Theorem and Lemma, respectively.

*Key invisibility of "Sign-then-Encrypt".* As observed in [15], parallel signcryption methods (incl. S&E and CtS&E) do not satisfy ciphertext anonymity — the recipient needs to know who is the sender in order to verify the signature. The key invisible signcryption scheme from [15], which has been revised in [9] following the analysis in [23], is a concrete construction based on bilinear maps and random oracles. As a second contribution we explore the key invisibility of the StE signcryption method, showing that it achieves SC-INVK-CCA (and by this SC-INDK-CCA and SC-IND-CCA) provided that the underlying public key encryption scheme satisfies two existing requirements, which are named *indistinguishability against adaptive chosen ciphertext attacks* (IND-CCA) and *indistinguishability of keys against adaptive chosen ciphertext attacks* (IK-CCA), respectively. It is well-known that Cramer-Shoup encryption scheme [8] offers both IND-CCA and IK-CCA security. In this way we readily obtain the first key invisible signcryption scheme in the standard model.

## 2 Preliminaries

### 2.1 Digital Signatures

SYNTAX. A signature scheme $\mathcal{S}$ comprises four efficient algorithms: $\mathcal{S} = (\mathsf{Setup}, \mathsf{KGen}, \mathsf{Sig}, \mathsf{Ver})$. The setup algorithm $\mathsf{Setup}$ takes as input a security parameter $1^k$ and outputs the public parameters $\lambda_{\mathcal{S}}$. The key generation algorithm $\mathsf{KGen}$ takes as input $\lambda_{\mathcal{S}}$ and outputs a signing key $sk$ and a verification key $vk$. The signing algorithm $\mathsf{Sig}$ takes as input a signing key $sk$ and a message $m$ from the associated message space $\mathcal{M}$, and outputs a signature $\sigma \leftarrow \mathsf{Sig}_{sk}(m)$. The verification algorithm $\mathsf{Ver}$ takes a message $m$, a signature $\sigma$ and a verification key $pk$ and outputs either a valid symbol $\top$ or an invalid symbol $\perp$. We require that $\mathsf{Ver}_{vk}(m, \mathsf{Sig}_{sk}(m)) = \top$, for any $m \in \mathcal{M}$.

SECURITY. We consider a standard security notion for signatures: *existential unforgeability under adaptive chosen message attacks* [13], denoted by UF-CMA. Intuitively, we require that an adversary is not able to generate a signature on a new message on behalf of a target signer. We define the adversary $\mathcal{A}$'s advantage $\mathsf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S},\mathcal{A}}(k)$ as

$$\Pr\left[\mathcal{S}.\mathsf{Ver}_{vk}(m,\sigma) = \top \,\middle|\, \begin{array}{l} \lambda_{\mathcal{S}} \leftarrow \mathsf{Setup}(1^k),\ (sk, vk) \leftarrow \mathcal{S}.\mathsf{KGen}(\lambda_{\mathcal{S}}), \\ (m,\sigma) \leftarrow \mathcal{A}^{O_{\mathsf{Sig}}(\cdot)}(vk),\ m \notin \mathrm{Query}(\mathcal{A}, O_{\mathsf{Sig}}(\cdot)) \end{array}\right],$$

where $\mathcal{A}$ is allowed to make a sequence of queries to the signing oracle $O_{\mathsf{Sig}}(\cdot)$, and $\mathrm{Query}(\mathcal{A}, O_{\mathsf{Sig}}(\cdot))$ is the set of queries made by $\mathcal{A}$ to oracle $O_{\mathsf{Sig}}(\cdot)$. $\mathcal{S}$ is said to be UF-CMA-secure, if the advantage function $\mathsf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S},\mathcal{A}}(k)$ is negligible in $k$ for any PPT adversary $\mathcal{A}$.

### 2.2 Public-Key Encryption

SYNTAX. A public key encryption scheme $\mathcal{E}$ comprises four efficient algorithms: $\mathcal{E} = (\mathsf{Setup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$. The setup algorithm $\mathsf{Setup}$ takes as input a security parameter $1^k$ and outputs the public parameters $\lambda_{\mathcal{E}}$. The key generation algorithm $\mathsf{KGen}$ takes as input $\lambda_{\mathcal{E}}$ and outputs a decryption key $dk$ and an encryption key $ek$. The encryption algorithm $\mathsf{Enc}$ takes as input an encryption key $ek$ and a message $m$ from the associated message space $\mathcal{M}$, and outputs a ciphertext $c \leftarrow \mathsf{Enc}_{ek}(m)$. The decryption algorithm $\mathsf{Dec}$ takes a decryption key $dk$ and a ciphertext $c$ to return the corresponding message $m$; we write $m \leftarrow \mathsf{Dec}_{dk}(c)$. We require that $\mathsf{Dec}_{dk}(\mathsf{Enc}_{ek}(m)) = m$, for any $m \in \mathcal{M}$.

SECURITY. We consider *indistinguishability against adaptive chosen ciphertext attacks* [20], denoted by IND-CCA, and *indistinguishability of keys against adaptive chosen ciphertext attacks* [5], denoted by IK-CCA. Intuitively, IND-CCA means that given a properly generated encryption key, no adversary $\mathcal{A}$ can distinguish encryptions of any two-equal length messages $m_0$, $m_1$ under this key.

$\mathsf{IND\text{-}CCA}$ security captures strong message (data)-privacy property and guarantees that, given a challenge ciphertext, no valid information about the underlying message (plaintext, or data) will be leaked. On the other hand, $\mathsf{IK\text{-}CCA}$ captures strong key-privacy property. It means that given two randomly selected encryption keys $ek_1$ and $ek_2$, no adversary $\mathcal{A}$ can distinguish encryptions of a same message $m$ under the two different keys. Given a challenge ciphertext, no valid information about the underlying key will be leaked in an $\mathsf{IK\text{-}CCA}$-secure encryption scheme. For $b = 0, 1$ and an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, which runs in two stages of $\mathsf{find}$ and $\mathsf{guess}$, consider the experiments

```
Experiment $\mathsf{Exp}^{\mathsf{IND\text{-}CCA},b}_{\mathcal{E},\mathcal{A}}(k)$ :
    $\lambda_{\mathcal{E}} \leftarrow \mathcal{E}.\mathsf{Setup}(1^k)$
    $(dk, ek) \leftarrow \mathcal{E}.\mathsf{KGen}(\lambda_{\mathcal{E}})$
    $(m_0, m_1, \omega) \leftarrow \mathcal{A}_1^{\mathcal{D}_{dk}(\cdot)}(\lambda_{\mathcal{E}}, ek, \mathsf{find})$
    $c_b \leftarrow \mathsf{Enc}_{ek}(m_b)$
    $d \leftarrow \mathcal{A}_2^{\mathcal{D}_{dk}(\cdot)}(c_b, \omega, \mathsf{guess})$
```

```
Experiment $\mathsf{Exp}^{\mathsf{IK\text{-}CCA},b}_{\mathcal{E},\mathcal{A}}(k)$ :
    $\lambda_{\mathcal{E}} \leftarrow \mathcal{E}.\mathsf{Setup}(1^k)$
    $(dk_0, ek_0) \leftarrow \mathcal{E}.\mathsf{KGen}(\lambda_{\mathcal{E}}$
    $(dk_1, ek_1) \leftarrow \mathcal{E}.\mathsf{KGen}(\lambda_{\mathcal{E}})$
    $(m, \omega) \leftarrow \mathcal{A}_1^{\mathcal{D}_{dk_0}(\cdot),\mathcal{D}_{dk_1}(\cdot)}(\lambda_{\mathcal{E}}, ek_0, ek_1, \mathsf{find})$
    $c_b \leftarrow \mathsf{Enc}_{ek_b}(m)$
    $d \leftarrow \mathcal{A}_2^{\mathcal{D}_{dk_0}(\cdot),\mathcal{D}_{dk_1}(\cdot)}(c_b, \omega, \mathsf{guess})$
```

where $|m_0| = |m_1|$, $\omega$ is some state information and $\mathcal{A}$ is allowed to invoke the decryption oracle $\mathcal{D}_{dk}(\cdot)$ (or $\mathcal{D}_{dk_1}(\cdot)$ and $\mathcal{D}_{dk_2}(\cdot)$) at any point with the only restriction that $c_b$ is not queried during the $\mathsf{guess}$ stage. We define the advantages $\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}(k)$ and $\mathsf{Adv}^{\mathsf{IK\text{-}CCA}}_{\mathcal{E},\mathcal{A}}(k)$, respectively, as follows:

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}(k) = \big|\Pr[\mathsf{Exp}^{\mathsf{IND\text{-}CCA},0}_{\mathcal{E},\mathcal{A}}(k) = 1] - \Pr[\mathsf{Exp}^{\mathsf{IND\text{-}CCA},1}_{\mathcal{E},\mathcal{A}}(k) = 1]\big|$$
$$\mathsf{Adv}^{\mathsf{IK\text{-}CCA}}_{\mathcal{E},\mathcal{A}}(k) = \big|\Pr[\mathsf{Exp}^{\mathsf{IK\text{-}CCA},0}_{\mathcal{E},\mathcal{A}}(k) = 1] - \Pr[\mathsf{Exp}^{\mathsf{IK\text{-}CCA},1}_{\mathcal{E},\mathcal{A}}(k) = 1]\big|.$$

$\mathcal{E}$ is said to be $\mathsf{IND\text{-}CCA}$ (resp. $\mathsf{IK\text{-}CCA}$) secure, if the advantage function $\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}(k)$ (resp. $\mathsf{Adv}^{\mathsf{IK\text{-}CCA}}_{\mathcal{E},\mathcal{A}}(k)$) is negligible in $k$ for any PPT adversary $\mathcal{A}$.

## 2.3 Signcryption Syntax

We will review the signcryption syntax used in [14, 15, 24]. A signcryption scheme is formalized by five PPT algorithms $\mathsf{SC}$ = ($\mathsf{Setup}$, $\mathsf{KeyGen}$, $\mathsf{SignCrypt}$, $\mathsf{UnSignCrypt}$, $\mathsf{Verify}$). The setup algorithm generates public parameters $\lambda_{sc} \leftarrow$

$\mathsf{Setup}(1^k)$. Taking as input the public parameters $\lambda_{sc}$, the key-generation algorithm outputs a key pair $(sk_U, pk_U) \leftarrow \mathsf{KG_s}(\lambda_{sc})$. On input a message $m$ from the associated message space $\mathcal{M}$, a private key $sk_U$, and a public key $pk_R$, the signcryption algorithm outputs a signcryption ciphertext $C \leftarrow \mathsf{SC.SignCrypt}(m, sk_U, pk_R)$. On input a private key $sk_R$ and a signcryption ciphertext $C$, the unsigncryption algorithm $\mathsf{UnSignCrypt}\ (sk_R, C)$ outputs either a tuple $(m, s, pk_U)$ where $m \in \mathcal{M}$, $s$ is auxiliary non-repudiation information (allowing to convince a third party of the origin of the message) and $pk_U$ is a public key, or a special symbol $\perp$ indicating failure. The verification algorithm $\mathsf{Verify}(m, s, pk_U)$ taking as input a message $m$, additional information $s$, and a public key $pk_U$, outputs either $\top$ if the additional information $s$ authenticates the message $m$ for the sender $pk_U$, or $\perp$ otherwise. The correctness requires that for any $m \in \mathcal{M}$, any correctly generated key pairs $(sk_U, pk_U)$ and $(sk_R, pk_R)$, we have $(m, s, pk_U) \leftarrow \mathsf{UnSignCrypt}(sk_R, \mathsf{SignCrypt}(m, sk_U, pk_R))$ and $\mathsf{Verify}(m, s, pk_U) = \perp$.

*Remark 1.* Note the slightly different syntax in comparison to [1]. The difference is that the unsigncryption algorithm takes as input sender's public key $pk_S$, receiver's secret key $sk_R$, and signcryption ciphertext $C$, and outputs either message $m$ or $\perp$. In this paper, we will adopt the signcryption syntax reviewed above since we intend to study various privacy notions in which the sender's identity may be unknown prior to the execution of the unsigncryption algorithm.

## 3 Security Notions for Signcryption Schemes

The existing security notions cover four aspects: existential unforgeability against chosen-message attacks, indistinguishability against chosen ciphertext attacks, ciphertext anonymity and key invisibility, which we recall in the following.

### 3.1 Unforgeability

A fundamental notion for signcryption schemes is existential unforgeability against chosen-message attacks [1]. This property prevents the adversary from forging a signcryption ciphertext on a new message or with respect to a new receiver on behalf of the target sender, and is formalized in the following experiment

$$\begin{aligned}
&\text{Experiment } \mathsf{Exp}^{\mathsf{UF\text{-}CMA}}_{\mathsf{SC},\mathcal{A}}(k) : \\
&\quad \lambda_{sc} \leftarrow \mathsf{SC.Setup}(1^k) \\
&\quad (sk_U, pk_U) \leftarrow \mathsf{SC.KeyGen}(\lambda_{sc}) \\
&\quad (C, sk_R, pk_R) \leftarrow \mathcal{A}^{\mathsf{SC.S}_{sk_U}(\cdot,\cdot), \mathsf{SC.D}_{sk_U}(\cdot)}(\lambda_{sc}, pk_U) \\
&\quad \text{success of } \mathcal{A} := [(m, s, pk_U) \leftarrow \mathsf{SC.UnSignCrypt}(sk_R, C) \\
&\qquad\qquad\qquad\quad \wedge\ \mathsf{Verify}(m, s, pk_U) = \top \\
&\qquad\qquad\qquad\quad \wedge\ (m, pk_R) \notin \mathit{Query}(\mathcal{A}, \mathsf{SC.S}_{sk_U}(\cdot,\cdot))]
\end{aligned}$$

where the signcryption oracle $\mathsf{SC.S}_{sk_U}(\cdot,\cdot)$ takes as input $(m', pk'_R)$ and outputs a signcryption ciphertext, the unsigncryption oracle $\mathsf{SC.D}_{sk_U}(\cdot)$ takes as input a signcryption ciphertext and outputs either $\perp$ or a tuple $(m', s', pk'_U)$ such that $\mathsf{Verify}(m', s', pk'_U) = \top$, and $Query(\mathcal{A}, \mathsf{SC.S}_{sk_U}(\cdot,\cdot))$ is the set of queries made by $\mathcal{A}$ to oracle $\mathsf{SC.S}_{sk_U}(\cdot,\cdot)$.

**Definition 1.** *A signcryption scheme is existentially unforgeable against chosen-message attacks* (SC-UF-CMA)*, if for all PPT adversaries $\mathcal{A}$ the following advantage function is negligible in $k$:*

$$\mathsf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathsf{SC},\mathcal{A}}(k) := \Pr[\mathcal{A} \text{ success}].$$

We remark existence of a stronger notion named strong existentially unforgeability against chosen-message attacks (SC-SUF-CMA), c.f. [14, 15, 24], which requires that the challenge signcryption ciphertext $C$ was not previously output by the signcryption oracle $\mathsf{SC.S}_{sk_U}(\cdot,\cdot)$ on input $(m, pk_R)$. However, as pointed out in [1] and similar to the signature setting in [13], the conventional (i.e. non-strong) unforgeability is sufficient for most scenarios in practice.

## 3.2 Confidentiality

The notion of indistinguishability against chosen ciphertext attacks [15] captures confidentiality of messages. That is, given a signcryption ciphertext, no valid information about the message that was signcrypted will be exposed to an adversary without the designated receiver's private key. Formally, for $b = 0, 1$ we consider the following experiments

$$\begin{aligned}
&\text{Experiment } \mathsf{Exp}^{\mathsf{IND\text{-}CCA},b}_{\mathsf{SC},\mathcal{A}}(k):\\
&\quad \lambda_{sc} \leftarrow \mathsf{SC.Setup}(1^k)\\
&\quad (sk_U, pk_U) \leftarrow \mathsf{SC.KeyGen}(\lambda_{sc})\\
&\quad (m_0, m_1, sk_S, \omega) \leftarrow \mathcal{A}_1^{\mathsf{SC.S}_{sk_U}(\cdot,\cdot),\mathsf{SC.D}_{sk_U}(\cdot)}(\lambda_{sc}, pk_U)\\
&\quad C_b \leftarrow \mathsf{SC.SignCrypt}(m_b, sk_S, pk_U)\\
&\quad d \leftarrow \mathcal{A}_2^{\mathsf{SC.S}_{sk_U}(\cdot,\cdot),\mathsf{SC.D}_{sk_U}(\cdot)}(C_b, \omega)
\end{aligned}$$

where $|m_0| = |m_1|$, $\omega$ is some state information, and oracles $\mathsf{SC.S}_{sk_U}(\cdot,\cdot)$ and $\mathsf{SC.D}_{sk_U}(\cdot)$ are the same as in the previous experiment $\mathsf{Exp}^{\mathsf{UF\text{-}CMA}}_{\mathsf{SC},\mathcal{A}}(k)$ with the only limitation of $\mathcal{A}_2$ not querying the challenge ciphertext $C_b$ to the unsigncryption oracle $\mathsf{SC.D}_{sk_U}(\cdot)$.

**Definition 2.** *A signcryption scheme is semantically secure against chosen ciphertext attacks* (SC-IND-CCA)*, if for all PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the following advantage function is negligible in $k$:*

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{SC},\mathcal{A}}(k) := |\Pr[\mathsf{Exp}^{\mathsf{IND\text{-}CCA},0}_{\mathsf{SC},\mathcal{A}}(k) = 1] - \Pr[\mathsf{Exp}^{\mathsf{IND\text{-}CCA},1}_{\mathsf{SC},\mathcal{A}}(k) = 1]|.$$

## 3.3 Ciphertext Anonymity

Intuitively, a signcryption scheme has ciphertext anonymity property [15] if signcryption ciphertexts reveal no information about the identities of the sender and receiver. Formally, consider the following experiment

```
Experiment $\mathsf{Exp}^{\mathsf{INDK\text{-}CCA}}_{\mathsf{SC},\mathcal{A}}(k)$ :
    $\lambda_{sc} \leftarrow \mathsf{SC.Setup}(1^k)$
    $(sk_{R,0}, pk_{R,0}) \leftarrow \mathsf{SC.KeyGen}(\lambda_{sc})$
    $(sk_{R,1}, pk_{R,1}) \leftarrow \mathsf{SC.KeyGen}(\lambda_{sc})$
    $\mathcal{O} := \{\mathsf{SC.S}_{sk_{R,0}}(\cdot,\cdot), \mathsf{SC.S}_{sk_{R,1}}(\cdot,\cdot), \mathsf{SC.D}_{sk_{R,0}}(\cdot), \mathsf{SC.D}_{sk_{R,1}}(\cdot)\}$
    $(m, sk_{S,0}, sk_{S,1}, \omega) \leftarrow \mathcal{A}_1^{\mathcal{O}}(\lambda_{sc}, pk_{R,0}, pk_{R,1})$
    $(b, b') \leftarrow \{0,1\}$
    $C \leftarrow \mathsf{SC.SignCrypt}(m, sk_{S,b}, pk_{R,b'})$
    $(d, d') \leftarrow \mathcal{A}_2^{\mathcal{O}}(C, \omega)$
```

where $\omega$ is some state information and $\mathcal{A}$ can have access to the signcryption and unsigncryption oracles at any point with the two limitations that $\mathcal{A}_2$ does not query $C$ to the unsigncryption oracles $\mathsf{SC.D}_{sk_{R,0}}(\cdot)$ and $\mathsf{SC.D}_{sk_{R,1}}(\cdot)$.

**Definition 3.** *A signcryption scheme is said to satisfy ciphertext anonymity* (SC-INDK-CCA)*, if for all PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the following advantage function is negligible in k:*

$$\mathsf{Adv}^{\mathsf{INDK\text{-}CCA}}_{\mathsf{SC},\mathcal{A}}(k) := |\Pr[(d, d') = (b, b')] - \frac{1}{4}|.$$

## 3.4 Key Invisibility

The notion of key invisibility for signcryption was formalized by Libert and Quisquater in [15]. It can be viewed as an extension of the invisibility concept proposed by Galbraith and Mao [12] for undeniable signatures. Intuitively, this notion captures that given a receiver, a specific signcryption ciphertext generated with respect to a chosen message, a chosen sender and a given receiver is indistinguishable to a random ciphertext uniformly chosen from the signcryption ciphertext space. Formally, for $b = 0, 1$ we consider the following experiments

```
Experiment $\mathsf{Exp}^{\mathsf{INVK\text{-}CCA},b}_{\mathsf{SC},\mathcal{A}}(k)$ :
    $\lambda_{sc} \leftarrow \mathsf{SC.Setup}(1^k)$
    $(sk_R, pk_R) \leftarrow \mathsf{SC.KeyGen}(\lambda_{sc})$
    $(m, sk_S, \omega) \leftarrow \mathcal{A}_1^{\mathsf{SC.S}_{sk_R}(\cdot,\cdot), \mathsf{SC.D}_{sk_R}(\cdot)}(\lambda_{sc}, pk_R)$
    $C_0 \leftarrow \mathsf{SC.SignCrypt}(sk_S, pk_R, m)$
    $C_1 \leftarrow \mathcal{C}$
    $d \leftarrow \mathcal{A}_2^{\mathsf{SC.S}_{sk_R}(\cdot,\cdot), \mathsf{SC.D}_{sk_R}(\cdot)}(C_b, \omega)$
```

where $\omega$ is some state information, $\mathcal{C}$ is the signcryption ciphertext space, $C_1$ is uniformly chosen at random from $\mathcal{C}$, and $\mathcal{A}$ can have access to the signcryption and unsigncryption oracles at any point with the two limitations that $\mathcal{A}_2$ does not query $C_b$ to the unsigncryption oracle $\mathsf{SC.D}_{sk_R}(\cdot)$.

**Definition 4.** *A signcryption scheme is said to satisfy key invisibility* (SC-INVK-CCA)*, if for all PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the following advantage function is negligible in k:*

$$\mathsf{Adv}_{\mathsf{SC},\mathcal{A}}^{\mathsf{INVK\text{-}CCA}}(k) := |\Pr[\mathsf{Exp}_{\mathsf{SC},\mathcal{A}}^{\mathsf{INVK\text{-}CCA},0}(k) = 1] - \Pr[\mathsf{Exp}_{\mathsf{SC},\mathcal{A}}^{\mathsf{INVK\text{-}CCA},1}(k) = 1]|.$$

## 4 Relations among Privacy Notions for Signcryption

We now define *anonymity*, an equivalent notion for ciphertext anonymity of signcryption schemes. This notion is conceptually simpler in comparison to *ciphertext anonimity* from [15] in that the adversary only needs to distinguish between two cases, depending on a single bit $b = 0, 1$, rather than between four cases in [15]. Formally, we consider the following experiments

```
Experiment $\mathsf{Exp}_{\mathsf{SC},\mathcal{A}}^{\mathsf{ANON\text{-}CCA},b}(k)$ :
    $\lambda_{sc} \leftarrow \mathsf{SC.Setup}(1^k)$
    $(sk_{R,0}, pk_{R,0}) \leftarrow \mathsf{SC.KeyGen}(\lambda_{sc})$
    $(sk_{R,1}, pk_{R,1}) \leftarrow \mathsf{SC.KeyGen}(\lambda_{sc})$
    $\mathcal{O} := \{\mathsf{SC.S}_{sk_{R,0}}(\cdot,\cdot), \mathsf{SC.S}_{sk_{R,1}}(\cdot,\cdot), \mathsf{SC.D}_{sk_{R,0}}(\cdot), \mathsf{SC.D}_{sk_{R,1}}(\cdot)\}$
    $(m, sk_{S,0}, sk_{S,1}, \omega) \leftarrow \mathcal{A}_1^{\mathcal{O}}(\lambda_{sc}, pk_{R,0}, pk_{R,1})$
    $C_b \leftarrow \mathsf{SC.SignCrypt}(m, sk_{S,b}, pk_{R,b})$
    $d \leftarrow \mathcal{A}_2^{\mathcal{O}}(C_b, \omega)$
```

where $\omega$ is some state information and $\mathcal{A}$ can have access to the signcryption and unsigncryption oracles at any point with the two limitations that $\mathcal{A}_2$ does not query $C_b$ to the unsigncryption oracles $\mathsf{SC.D}_{sk_{R,0}}(\cdot)$ and $\mathsf{SC.D}_{sk_{R,1}}(\cdot)$.

**Definition 5.** *A signcryption scheme is said to satisfy anonymity* (SC-ANON-CCA)*, if for all PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, the advantage function is negligible in k:*

$$\mathsf{Adv}_{\mathsf{SC},\mathcal{A}}^{\mathsf{ANON\text{-}CCA}}(k) := |\Pr[\mathsf{Exp}_{\mathsf{SC},\mathcal{A}}^{\mathsf{ANON\text{-}CCA},0}(k) = 1] - \Pr[\mathsf{Exp}_{\mathsf{SC},\mathcal{A}}^{\mathsf{ANON\text{-}CCA},1}(k) = 1]|.$$

We now show that ciphertext anonymity and anonymity are equivalent.

**Theorem 1 (SC-INDK-CCA $\Leftrightarrow$ SC-ANON-CCA).** *For signcryption schemes, anonymity is equivalent to ciphertext anonymity.*

Proof of Theorem 1 is presented in the full version of this paper. $\square$

## 4.1 Separation between Ciphertext Anonymity and SC-IND-CCA

Intuitively, ciphertext anonymity captures identity privacy and indistinguishability against chosen ciphertext attacks captures message privacy. The goals of ciphertext anonymity and indistinguishability against chosen ciphertext attacks are orthogonal. Formally, Lemmas 1 and 2 proven in the full version of this paper, separate the two notions.

**Lemma 1 (SC-IND-CCA $\not\Rightarrow$ SC-INDK-CCA).** *Let* $\mathsf{SC} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{SignCrypt}, \mathsf{UnSignCrypt})$ *be a signcryption scheme. If the scheme* $\mathsf{SC}$ *satisfies indistinguishability against chosen ciphertext attacks, then it may not satisfy ciphertext anonymity.*

**Lemma 2 (SC-INDK-CCA $\not\Rightarrow$ SC-IND-CCA).** *Let* $\mathsf{SC} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{SignCrypt}, \mathsf{UnSignCrypt})$ *be a signcryption scheme. If the scheme* $\mathsf{SC}$ *satisfies ciphertext anonymity, then it may not satisfy indistinguishability against chosen ciphertext attacks.*

## 4.2 Relationship between Key Invisibility and Ciphertext Anonymity

Next, we investigate the relationship between key invisibility and ciphertext anonymity. We shall use anonymity instead of ciphertext anonymity in our analysis, as these two are equivalent by Theorem 1.

**Theorem 2 (SC-INVK-CCA $\Rightarrow$ SC-ANON-CCA).** *Let* $\mathsf{SC}$ *be a signcryption scheme. If the scheme* $\mathsf{SC}$ *satisfies key invisibility, then it satisfies anonymity.*

Proof of Theorem 2 is presented in the full version of this paper. $\square$

Note that Libert and Quisquater [15] were only able to prove implication of ciphertext anonymity by key invisibility for a class of signcryption schemes satisfying a particular property, namely that for a given message and a given sender's private key, the output of the signcryption algorithm must be uniformly distributed in the ciphertext space when the receiver's public key is random. Our results in Theorems 1 and 2 lift this restriction.

## 4.3 Relationship between Key Invisibility and SC-IND-CCA

Our next result shows that key invisibility, which originally was viewed as a notion for protecting privacy of user identities [15], is in fact a much stronger notion that implies indistinguishability against chosen ciphertext attacks.

**Theorem 3 (SC-INVK-CCA $\Rightarrow$ SC-IND-CCA).** *Let* $\mathsf{SC}$ *be a signcryption scheme. If the scheme* $\mathsf{SC}$ *satisfies key invisibility, then it satisfies indistinguishability against chosen ciphertext attacks.*

Proof of Theorem 3 is presented in the full version of this paper. □

From Theorem 1, Lemma 1, Lemma 2, Theorem 2 and Theorem 3, we can safely conclude that key invisibility is strictly stronger than both indistinguishability against chosen ciphertext attacks and ciphertext anonymity.

## 5 Sign-then-Encrypt Generic Construction

In this section, we revisit the generic construction of signcryption schemes based on the sign-then-encrypt method [1,2]. We show that the resulting signcryption schemes can achieve key invisibility when appropriate encryption schemes are employed.

### 5.1 Scheme

Let $\mathcal{S} = (\mathsf{Setup}, \mathsf{KGen}, \mathsf{Sig}, \mathsf{Ver})$ be a signature scheme and $\mathcal{E} = (\mathsf{Setup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ be a public key encryption scheme. Signcryption schemes based on the sign-then-encrypt method can be constructed as follows:

- $\mathsf{Setup}(1^k)$: On input a security parameter $k$, this algorithm runs $\lambda_{\mathcal{S}} \leftarrow \mathcal{S}.\mathsf{Setup}(1^k)$ and $\lambda_{\mathcal{E}} \leftarrow \mathcal{E}.\mathsf{Setup}(1^k)$, respectively. The public parameters are set as $\lambda_{sc} := (\lambda_{\mathcal{S}}, \lambda_{\mathcal{E}})$.
- $\mathsf{KeyGen}(\lambda_{sc})$: The user $U_i$ runs $\mathcal{S}.\mathsf{KGen}(\lambda_{\mathcal{S}}) \rightarrow (sk_i, vk_i)$ and $\mathcal{E}.\mathsf{KGen}(\lambda_{\mathcal{E}}) \rightarrow (dk_i, ek_i)$, respectively. The secret and public key pair is set as $(sk_{U_i}, pk_{U_i}) := ((sk_i, dk_i), (vk_i, ek_i))$.
- $\mathsf{SignCrypt}(m, sk_{U_i}, pk_{U_j})$: To signcrypt a message $m$ for the receiver $U_j$, $U_i$ first produces a signature $\sigma$ on $m||pk_{U_j}$, i.e., $\sigma \leftarrow \mathcal{S}.\mathsf{Sig}_{sk_i}(m||pk_{U_j})$, and then encrypts $m||\sigma||pk_{U_i}$ under receiver $U_j$'s encryption key, i.e. $c \leftarrow \mathcal{E}.\mathsf{Enc}_{ek_j}(m||\sigma||pk_{U_i})$. The signcryption ciphertext is set as $C := c$.
- $\mathsf{UnSignCrypt}(sk_{U_j}, C)$: On receiving a signcryption ciphertext $C$, receiver $U_j$ firstly decrypts it using its own decryption key $dk_j$, i.e., $m||\sigma||pk_{U_i} \leftarrow \mathcal{E}.\mathsf{Dec}_{dk_j}(C)$, and then checks if $\mathcal{S}.\mathsf{Ver}_{vk_i}\ (m||pk_{U_j}, \sigma) = \top$. If so, it outputs $(m, s, pk_{U_i})$ where $s = (pk_{U_j}, \sigma)$; otherwise, it returns $\perp$.
- $\mathsf{Verify}(m, s, pk_{U_i})$: This algorithm parses $s$ and $pk_{U_i}$ as $(pk_{U_j}, \sigma)$ and $(vk_i, ek_i)$, respectively, and outputs $\mathcal{S}.\mathsf{Ver}_{vk_i}(m||pk_{U_j}, \sigma)$.

### 5.2 Security of the Generic Construction

From the relations discussed in Section 4, we only need to show that the above generic construction results in signcryption schemes that are existentially unforgeable against chosen-message attacks and satisfy key invisibility. The former requirement has already been proven in [1], who stated the following theorem:

**Theorem 4 ([1]).** *Let* $\mathsf{SC}$ *be the above generic signcrypiton scheme. If the signature scheme* $\mathcal{S}$ *is* $\mathsf{UF\text{-}CMA}$*-secure, then* $\mathsf{SC}$ *is existentially unforgeable against chosen-message attacks.*

We thus focus on key invisibility, for which we need to specify how to uniformly sample signcryption ciphertexts from the ciphertext space. Here we will adopt the very natural method for uniform sampling, i.e., uniformly and independently choosing a message $m \in M$, a sender's secret key $sk_{U_i}$, and a receiver's public key $pk_{U_j}$, and returning a signcryption ciphertext $C \leftarrow \mathsf{SC.SignCrypt}(m, sk_{U_i}, pk_{U_j})$.

**Theorem 5.** *Let $\mathcal{S}$ be a signature scheme, $\mathcal{E}$ be a public-key encryption scheme that is both* IND-CCA*-secure and* IK-CCA*-secure. Then the above generic signcrypiton scheme* SC *satisfies key invisibility.*

*Proof.* To show the security, we first define two games, and then show in Claims 1 and 2 that no adversary $\mathcal{A}$ can break the key invisibility property of SC.

**Game 0.** This is the real experiment between the challenger and an adversary $\mathcal{A}$. This means that the challenger firstly correctly generates the target receiver's key pairs $(sk_R, pk_R) := ((sk_0, dk_0), (vk_0, ek_0))$, forwards $pk_R$ to the adversary $\mathcal{A}$, and then provides accesses to signcryption oracle $\mathsf{SC.S}_{sk_R}(\cdot, \cdot)$ and unsigncryption oracle $\mathsf{SC.D}_{sk_R}(\cdot)$. In the challenge phase, after $\mathcal{A}$ submits $(m^*, sk_S = (sk_1, dk_1))$, the challenger randomly flips a coin $b \in \{0, 1\}$. If $b = 0$, the challenger produces a signature $\sigma_0$ on $m^*||pk_R$ under the signing key $sk_1$, i.e., $\sigma_0 \leftarrow \mathcal{S}.\mathsf{Sig}_{sk_1}(m^*||pk_R)$, encrypts $m^*||\sigma_0||pk_S$ under the receiver's encryption key, i.e. $C_0 \leftarrow \mathcal{E}.\mathsf{Enc}_{ek_0}(m^*||\sigma_0||pk_S)$, and returns $C_0$ to $\mathcal{A}$. If $b = 1$, the challenger independently and uniformly chooses $m' \in \mathcal{M}$, a sender's secret key $sk'_S := (sk'_1, dk'_1)$ and a receiver's public key $pk'_R := (vk'_0, ek'_0)$, produces a signature $\sigma_1$ on $m'||pk'_R$, i.e., $\sigma_1 \leftarrow \mathcal{S}.\mathsf{Sig}_{sk'_1}(m'||pk'_R)$, encrypts $m'||\sigma_1||pk'_S$ under the receiver's encryption key, i.e. $C_1 \leftarrow \mathcal{E}.\mathsf{Enc}_{ek'_0}(m'||\sigma_1||pk'_S)$, and returns $C_1$ to $\mathcal{A}$. Besides, the challenger provides access to signcryption oracle $\mathsf{SC.S}_{sk_R}(\cdot, \cdot)$ and unsigncryption oracle $\mathsf{SC.D}_{sk_R}(\cdot)$.

**Game 1.** This is the same as Game 0, with the exception that in the challenge phase, the challenger computes $C_1 \leftarrow \mathcal{E}.\mathsf{Enc}_{ek_0}(m'||\sigma_1||pk'_S)$, and returns $C_1$ to $\mathcal{A}$ when $b = 1$.

Next we link the probability that $\mathcal{A}$ wins in Game 0 and Game 1. Let $\mathsf{S}_1$ be the advantage that $\mathcal{A}$ wins in Game 1. Thus $\Pr[S_1] = |\Pr[\mathsf{Exp}_{\mathsf{SC},\mathcal{A}}^{\textbf{Game 1},0}(k) = 1] - \Pr[\mathsf{Exp}_{\mathsf{SC},\mathcal{A}}^{\textbf{Game 1},1}(k) = 1]|$, where $\mathsf{Exp}_{\mathsf{SC},\mathcal{A}}^{\textbf{Game 1},b}(k)$ is the output of $\mathcal{A}$ in Game 1 when the challenge ciphertext is $C_b$.

**Claim 1**

$$|\mathsf{Adv}_{\mathsf{SC},\mathcal{A}}^{\mathsf{INVK\text{-}CCA}}(k) - \Pr[S_1]| = 2 \cdot \mathsf{Adv}_{\mathcal{E},\mathcal{B}}^{\mathsf{IK\text{-}CCA}}(k), \tag{1}$$

where $\mathsf{Adv}_{\mathcal{E},\mathcal{B}}^{\mathsf{IK\text{-}CCA}}(k)$ is the advantage of an adversary $\mathcal{B}$ that breaks the IK-CCA security of the encryption scheme $\mathcal{E}$.

We show that any difference between $\mathsf{Adv}_{\mathsf{SC},\mathcal{A}}^{\mathsf{INVK\text{-}CCA}}(k)$ and $\Pr[S_1]$ can be parlayed into an algorithm $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ that breaks the IK-CCA security of the encryption scheme $\mathcal{E}$. Recall that $\mathcal{B}_1$ gets $(\lambda_{\mathcal{E}}, ek, ek')$ as input and has access to decryption oracles $\mathcal{D}_{dk}(\cdot)$ and $\mathcal{D}_{dk'}(\cdot)$. $\mathcal{B}_1$ runs $\lambda_{\mathcal{S}} \leftarrow \mathcal{S}.\mathsf{Setup}(1^k)$, $\mathcal{S}.\mathsf{KGen}(\lambda_{\mathcal{S}}) \rightarrow$

$(sk_0, vk_0)$ and sets $\lambda_{sc} := (\lambda_{\mathcal{S}}, \lambda_{\mathcal{E}})$ and $pk_R := (vk_0, ek)$. $\mathcal{B}_1$ runs $\mathcal{A}_1$ as a subroutine by forwarding $(\lambda_{sc}, pk_R)$.

When $\mathcal{A}_1$ makes a signcryption query $(m, pk_U = (vk_U, ek_U))$ to $\mathsf{SC.S}_{sk_R}(\cdot,\cdot)$, $\mathcal{B}_1$ first produces a signature $\sigma$ on $m||pk_U$ under the signing key $sk_0$, i.e., $\sigma \leftarrow \mathcal{S}.\mathsf{Sig}_{sk_0}(m||pk_U)$, and then encrypts $m||\sigma||pk_R$ under the encryption key $ek_U$, i.e. $c \leftarrow \mathcal{E}.\mathsf{Enc}_{ek_U}(m||\sigma||pk_R)$. The signcryption ciphertext is set as $C := c$, and returned to $\mathcal{A}_1$ as the reply. When $\mathcal{A}_1$ makes a unsigncryption query $C$ to $\mathsf{SC.D}_{sk_R}(\cdot)$, $\mathcal{B}_1$ submits $C$ to its own decryption oracle $\mathcal{D}_{dk}(\cdot)$. If the reply is not of the form $m||\sigma||pk_U$ where $pk_U$ is a public key, then $\mathcal{B}_1$ returns $\perp$ to $\mathcal{A}_1$. Otherwise, $\mathcal{B}_1$ decomposes $pk_U$ as $(vk_U, ek_U)$, and further checks whether $\mathcal{S}.\mathsf{Ver}_{vk_U}(m||pk_R, \sigma) = \top$. If so, $\mathcal{B}_1$ returns $(m, (pk_R, \sigma), pk_U)$ to $\mathcal{A}_1$, and otherwise $\perp$ is returned.

At some time, $\mathcal{A}_1$ submits $(m^*, sk_S = (sk_1, dk_1))$. $B_1$ randomly flips a coin $\tilde{b} \in \{0,1\}$. If $\tilde{b} = 0$, $\mathcal{B}$ first produces a signature $\sigma_0$ on $m^*||pk_R$ under the signing key $sk_1$, i.e., $\sigma_0 \leftarrow \mathcal{S}.\mathsf{Sig}_{sk_1}(m^*||pk_R)$, encrypts $m^*||\sigma_0||pk_S$ under the receiver's encryption key, i.e. $C_0 \leftarrow \mathcal{E}.\mathsf{Enc}_{ek}(m^*||\sigma_0||pk_S)$, and returns $C_0$ to $\mathcal{A}$. If $\tilde{b} = 1$, $\mathcal{B}$ independently and uniformly chooses $m' \in \mathcal{M}$, a sender's secret key $sk'_S := (sk'_1, dk'_1)$ and a public verification key $vk'_0$, sets $pk'_R := (vk'_0, ek')$, produces a signature $\sigma_1$ on $m'||pk'_R$ using the signing key $sk'_1$, i.e., $\sigma_1 \leftarrow \mathcal{S}.\mathsf{Sig}_{sk'_1}(m'||pk'_R)$, and submits $m'||\sigma_1||pk'_S$ where $pk'_S$ is the corresponding public key of $sk'_S$ to its own challenger. Let $C_1$ denote the reply of $\mathcal{B}$'s own challenger. $\mathcal{B}$ returns $C_1$ to $\mathcal{A}$. $\mathcal{B}_2$ simulates the oracles in the same way as $\mathcal{B}_1$ did.

Note that $\mathcal{A}_2$ never makes an unsigncryption query $C_b$ where $b \in \{0,1\}$ to $\mathsf{SC.D}_{sk_R}(\cdot)$, thus $\mathcal{B}_2$ does not make the query $C_b$ to its decryption oracles $\mathcal{D}_{dk}(\cdot)$ or $\mathcal{D}_{dk'}(\cdot)$. Finally $\mathcal{A}_2$ outputs a bit $d$. $\mathcal{B}_2$ outputs $d$ when $\tilde{b} = 1$, and returns failure when $\tilde{b} = 0$. When $C_1$ is the encryption of $m'||\sigma_1||pk'_S$ under $ek$, the environment simulated by $\mathcal{B}$ is exactly the same as in Game 1. While $C_1$ is the encryption of $m'||\sigma_1||pk'_S$ under $ek'$, the environment simulated by $\mathcal{B}$ is exactly the same as in Game 0. Thus we have

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{IK\text{-}CCA}}_{\mathcal{E},\mathcal{B}}(k) &= \left|\Pr[\mathsf{Exp}^{\mathsf{IK\text{-}CCA},0}_{\mathcal{E},\mathcal{B}}(k) = 1] - \Pr[\mathsf{Exp}^{\mathsf{IK\text{-}CCA},1}_{\mathcal{E},\mathcal{B}}(k) = 1]\right| \\
&= \left|\Pr[\tilde{b} = 1] \cdot \Pr[\mathsf{Exp}^{\textbf{Game 1},1}_{\mathsf{SC},\mathcal{A}}(k) = 1]\right. \\
&\quad \left. - \Pr[\tilde{b} = 1] \cdot \Pr[\mathsf{Exp}^{\mathsf{INVK\text{-}CCA},1}_{\mathsf{SC},\mathcal{A}}(k) = 1]\right| \\
&= \left|(\frac{1}{2} \cdot \Pr[\mathsf{Exp}^{\textbf{Game 1},0}_{\mathsf{SC},\mathcal{A}}(k) = 1] - \frac{1}{2} \cdot \Pr[\mathsf{Exp}^{\textbf{Game 1},1}_{\mathsf{SC},\mathcal{A}}(k) = 1])\right. \\
&\quad - \frac{1}{2} \cdot \Pr[\mathsf{Exp}^{\mathsf{INVK\text{-}CCA},0}_{\mathsf{SC},\mathcal{A}}(k) = 1] \\
&\quad \left. + \frac{1}{2} \cdot \Pr[\mathsf{Exp}^{\mathsf{INVK\text{-}CCA},1}_{\mathsf{SC},\mathcal{A}}(k) = 1]\right| \qquad (2) \\
&= \frac{1}{2} \cdot \left|\Pr[S_1] - \mathsf{Adv}^{\mathsf{INVK\text{-}CCA}}_{\mathsf{SC},\mathcal{A}}(k)\right|.
\end{aligned}$$

Equation (2) follows from the fact that $\Pr[\mathsf{Exp}^{\textbf{Game 1},0}_{\mathsf{SC},\mathcal{A}}(k) = 1]$ equals to $\Pr[\mathsf{Exp}^{\mathsf{INVK\text{-}CCA},0}_{\mathsf{SC},\mathcal{A}}(k) = 1]$, as the experiments are exactly the same.

**Claim 2**

$$\Pr[S_1] \leq \mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{E},\mathcal{C}}(k), \tag{3}$$

where $\mathsf{Adv}^{\mathsf{IK\text{-}CCA}}_{\mathcal{E},\mathcal{C}}(k)$ is the advantage of an adversary $\mathcal{C}$ that breaks the IND-CCA security of the encryption scheme $\mathcal{E}$.

To show this, we build an algorithm $\mathcal{C}$ that employs the adversary $\mathcal{A}$ in Game 1 to break the IND-CCA security of the encryption scheme $\mathcal{E}$. Recall that $\mathcal{C}$ gets $(\lambda_{\mathcal{E}}, ek)$ as input and has access to a decryption oracle $\mathcal{D}_{dk}(\cdot)$. $\mathcal{C}$ runs $\lambda_{\mathcal{S}} \leftarrow \mathcal{S}.\mathsf{Setup}(1^k)$, $\mathcal{S}.\mathsf{KGen}(\lambda_{\mathcal{S}}) \rightarrow (sk_0, vk_0)$ and sets $\lambda_{sc} := (\lambda_{\mathcal{S}}, \lambda_{\mathcal{E}})$ and $pk_R := (vk_0, ek)$. $\mathcal{C}$ runs $\mathcal{A}_1$ as a subroutine by forwarding $(\lambda_{sc}, pk_R)$.

When $\mathcal{A}_1$ makes a signcryption query $(m, pk_U = (vk_U, ek_U))$ to $\mathsf{SC.S}_{sk_R}(\cdot,\cdot)$, $\mathcal{C}$ first produces a signature $\sigma$ on $m||pk_U$, i.e., $\sigma \leftarrow \mathcal{S}.\mathsf{Sig}_{sk_0}(m||pk_U)$, and then encrypts $m||\sigma||pk_R$ under the encryption key $ek_U$, i.e. $c \leftarrow \mathcal{E}.\mathsf{Enc}_{ek_U}(m||\sigma||pk_R)$. The signcryption ciphertext is set as $C := c$, and returned to $\mathcal{A}_1$ as the reply. When $\mathcal{A}_1$ makes a unsigncryption query $C$ to $\mathsf{SC.D}_{sk_R}(\cdot)$, $\mathcal{C}$ submits $C$ to its own decryption oracle $\mathcal{D}_{dk}(\cdot)$. If the reply is not of the form $m||\sigma||pk_U$ where $pk_U$ is a public key, then $\mathcal{C}$ returns $\perp$ to $\mathcal{A}_1$. Otherwise, $\mathcal{C}$ decomposes $pk_U$ as $(vk_U, ek_U)$, and further checks whether $\mathcal{S}.\mathsf{Ver}_{vk_U}(m||pk_R, \sigma) = \top$. If so, $\mathcal{C}$ returns $(m, (pk_R, \sigma), pk_U)$ to $\mathcal{A}_1$, and otherwise $\perp$ is returned.

At some time, $\mathcal{A}_1$ submits $(m^*, sk_S = (sk_1, dk_1))$. $\mathcal{C}$ first produces a signature $\sigma_0$ on $m^*||pk_R$ under the signing key $sk_1$, i.e., $\sigma_0 \leftarrow \mathcal{S}.\mathsf{Sig}_{sk_1}(m^*||pk_R)$. Then $\mathcal{C}$ independently and uniformly chooses $m' \in \mathcal{M}$, a sender's secret key $sk'_S := (sk'_1, dk'_1)$ and a receiver's public $pk'_R$, produces a signature $\sigma_1$ on $m'||pk'_R$ under the signing key $sk'_1$, i.e., $\sigma_1 \leftarrow \mathcal{S}.\mathsf{Sig}_{sk'_1}(m'||pk'_R)$. $\mathcal{C}$ sets $\bar{m}_0 := m^*||\sigma_0||pk_S$, $\bar{m}_1 := m'||\sigma_1||pk'_S$ where $pk_S$ and $pk'_S$ are the corresponding public keys of $sk_S$ and $sk'_S$ respectively, and submits $\bar{m}_0$ and $\bar{m}_1$ to its own challenger. Let $C_b$ denote the reply of $\mathcal{C}$'s own challenger. $\mathcal{C}$ returns $C_b$ to $\mathcal{A}$. $\mathcal{C}$ then simulates the oracles in the same way as it did before.

Note that $\mathcal{A}_2$ never makes an unsigncryption query $C_b$ to $\mathsf{SC.D}_{sk_R}(\cdot)$, thus $\mathcal{B}_2$ does not make the query $C_b$ to its decryption oracle $\mathcal{D}_{dk}(\cdot)$. Finally $\mathcal{A}_2$ outputs a bit $d$. $\mathcal{C}$ outputs $d$. The environment simulated by $\mathcal{C}$ is exactly the same as in Game 1. Thus we have $\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{E},\mathcal{C}}(k) = \Pr[S_1]$.

As a sequence of equations (1), (3) gained above, we have $\mathsf{Adv}^{\mathsf{INVK\text{-}CCA}}_{\mathsf{SC},\mathcal{A}}(k) \leq 2 \cdot \mathsf{Adv}^{\mathsf{IK\text{-}CCA}}_{\mathcal{E},\mathcal{B}}(k) + \mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{E},\mathcal{C}}(k)$ . This concludes the proof. □

## 6 Conclusion

In this paper, we first revisited the existing privacy notions of signcryption schemes, namely indistinguishability against chosen ciphertext attacks, ciphertext anonymity and key invisibility. We demonstrated the separation between indistinguishability against chosen ciphertext attacks and ciphertext anonymity, and showed that both notions are implied by key invisibility. Finally we proposed the first generic construction for key invisible signcryption schemes in the standard model.

## References

1. An, J.H., Dodis, Y., Rabin, T.: On the security of joint signature and encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 83–107. Springer, Heidelberg (2002)
2. Au, J.H., Rabin, T.: Security for Signcryption: The Two-User Model. In: Dent, A., Zheng, Y. (eds.) Practical Signcryption, Information Security and Cryptography. Springer (2010)
3. Baek, J., Steinfeld, R., Zheng, Y.: Formal proofs for the security of signcryption. In: Naccache, D., Paillier, P. (eds.) PKC 2002. LNCS, vol. 2274, pp. 80–98. Springer, Heidelberg (2002)
4. 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)
5. Bellare, M., Boldyreva, A., Desai, A., Pointcheval, D.: Key-privacy in public-key encryption. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 566–582. Springer, Heidelberg (2001)
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. Boyen, X.: Multipurpose identity-based signcryption – a Swiss Army knife for identity-based cryptography. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 383–399. Springer, Heidelberg (2003), `http://www.cs.stanford.edu/~xb/crypto03/`
8. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998)
9. Dent, A.W., Zheng, Y. (eds.): Practical Signcryption. Springer (2010)
10. Dent, A.W., Fischlin, M., Manulis, M., Stam, M., Schröder, D.: Confidential Signatures and Deterministic Signcryption. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 462–479. Springer, Heidelberg (2010)
11. Dodis, Y., Freedman, M.J., Jarecki, S., Walfish, S.: Optimal Signcryption from Any Trapdoor Permutation. Cryptology ePrint Archive, Report 2004/020 (2004), `http://eprint.iacr.org/`
12. 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)
13. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM J. Comput. 17(2), 281–308 (1988)
14. Li, C.K., Yang, G., Wong, D.S., Deng, X., Chow, S.S.M.: An efficient signcryption scheme with key privacy. In: López, J., Samarati, P., Ferrer, J.L. (eds.) EuroPKI 2007. LNCS, vol. 4582, pp. 78–93. Springer, Heidelberg (2007)
15. Libert, B., Quisquater, J.-J.: Efficient Signcryption with Key Privacy from Gap Diffie-Hellman Groups. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 187–200. Springer, Heidelberg (2004)
16. Libert, B., Quisquater, J.-J.: Improved Signcryption from $q$-Diffie-Hellman Problems. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 220–234. Springer, Heidelberg (2005)
17. Malone-Lee, J.: A General Construction for Simultaneous Signing and Encrypting. In: Smart, N.P. (ed.) Cryptography and Coding 2005. LNCS, vol. 3796, pp. 116–135. Springer, Heidelberg (2005)

18. Malone-Lee, J., Mao, W.: Two Birds One Stone: Signcryption Using RSA. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 211–225. Springer, Heidelberg (2003)
19. Pieprzyk, J., Pointcheval, D.: Parallel Authentication and Public-Key Encryption. In: Safavi-Naini, R., Seberry, J. (eds.) ACISP 2003. LNCS, vol. 2727, pp. 387–401. Springer, Heidelberg (2003)
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. Shin, J.-B., Lee, K., Shim, K.: New DSA-Verifiable Signcryption Schemes. In: Lee, P.J., Lim, C.H. (eds.) ICISC 2002. LNCS, vol. 2587, pp. 35–47. Springer, Heidelberg (2003)
22. Steinfeld, R., Zheng, Y.: A Signcryption Scheme Based on Integer Factorization. In: Okamoto, E., Pieprzyk, J.P., Seberry, J. (eds.) ISW 2000. LNCS, vol. 1975, pp. 308–322. Springer, Heidelberg (2000)
23. Tan, C.-H.: On the security of signcryption scheme with key privacy. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. E88-A(4), 1093–1095 (2005)
24. Yang, G., Wong, D.S., Deng, X.: Analysis and improvement of a signcryption scheme with key privacy. In: Zhou, J., López, J., Deng, R.H., Bao, F. (eds.) ISC 2005. LNCS, vol. 3650, pp. 218–232. Springer, Heidelberg (2005)
25. Yum, D.H., Lee, P.J.: New Signcryption Schemes Based on KCDSA. In: Kim, K.-C. (ed.) ICISC 2001. LNCS, vol. 2288, pp. 305–317. Springer, Heidelberg (2002)
26. 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)

# Injective Encodings to Elliptic Curves

Pierre-Alain Fouque[1], Antoine Joux[2], and Mehdi Tibouchi[3]

[1] University of Rennes
pierre-alain.fouque@ens.fr
[2] CryptoExperts and Université de Versailles–Saint-Quentin
antoine.joux@m4x.org
[3] NTT Secure Platform Laboratories
tibouchi.mehdi@lab.ntt.co.jp

**Abstract.** For a number of elliptic curve-based cryptographic protocols, it is useful and sometimes necessary to be able to encode a message (a bit string) as a point on an elliptic curve in such a way that the message can be efficiently and uniquely recovered from the point. This is for example the case if one wants to instantiate CPA-secure ElGamal encryption directly in the group of points of an elliptic curve. More practically relevant settings include Lindell's UC commitment scheme (EUROCRYPT 2011) or structure-preserving primitives.

It turns out that constructing such an encoding function is not easy in general, especially if one wishes to encode points whose length is large relative to the size of the curve. There is a probabilistic, "folklore" method for doing so, but it only provably works for messages of length less than half the size of the curve.

In this paper, we investigate several approaches to injective encoding to elliptic curves, and in particular, we propose a new, essentially optimal geometric construction for a large class of curves, including Edwards curves; the resulting algorithm is also quite efficient, requiring only one exponentiation in the base field and simple arithmetic operations (however, the curves for which the map can be constructed have a point of order two, which may be a limiting factor for possible applications). The new approach is based on the existence of a covering curve of genus 2 for which a bijective encoding is known.

**Keywords:** Elliptic Curve Cryptography, Injective Encoding, Algebraic Curves.

## 1 Introduction

Various cryptographic protocols based on the hardness of Diffie-Hellman-like problems in a group $\mathbb{G}$, such as ElGamal encryption [7] or Lindell's recent universally-composable commitment scheme [14], assume the existence of an efficient (possibly randomized) algorithm $f$ mapping messages $m \in \{0,1\}^\ell$ to elements of $\mathbb{G}$, in such a way that $m$ can also be recovered efficiently from $f(m)$.

For example, ElGamal encryption is *a priori* defined on group elements, so that a message needs to be mapped to an element of $\mathbb{G}$ before encrypting it, and mapped back to a bit string upon decryption. Similarly, such a function $f$ is an important ingredient for structure-preserving cryptography [1]: indeed, inputs and outputs of structure-preserving primitives are all group elements; this offers convenient composability properties, but to use e.g. commitments or encryption on actual bit strings, a way to map strings to the group and conversely is required.

Moreover, the size $\ell$ of supported bit strings should preferably be as close as possible to the bit size of $\mathbb{G}$ to optimize bandwidth. We call such an algorithm $f$ an injective encoding.

For certain groups $\mathbb{G}$, like multiplicative groups of finite fields or some supersingular elliptic curves, it is not difficult to construct injective encodings achieving the optimal value of $\ell$. On the other hand, for a general group $\mathbb{G}$, it is not obvious how to construct a function $f$ with $\ell$ even super-logarithmic in the size of $\mathbb{G}$. In §2.3, we prove that this is not possible with a deterministic generic group algorithm.

When $\mathbb{G}$ is the group of points of any elliptic curve over a finite field, one can construct a probabilistic injective encoding with $\ell$ equal to about half of the size of $\mathbb{G}$, as we show in §2.4, but we do not know of provable constructions achieving a better $\ell$ in general. Works on deterministic hashing to elliptic curves, such as [17,11], typically do *not* help addressing this problem, as the functions they construct are not injective, and it is not clear how to find a convenient subset of their domain on which they become injective. Recently, however, a solution was proposed by Farashahi [8] in the special case of Hessian elliptic curves over finite fields $\mathbb{F}_q$ with $q \equiv 2 \pmod 3$.

In §3, we propose an essentially optimal construction for all ordinary elliptic curves over fields $\mathbb{F}_q$ with $q \equiv 3 \pmod 4$ with group order divisible by 4; this includes the well-known Edwards curves studied by Edwards and Bernstein–Lange [2], as well as twisted Huff curves, as studied by Joye et. al. [13]. Our construction is based on the bijective encoding from [10] to certain hyperelliptic curves of genus 2, and on the observation from [12] that those curves are quadratic covers of elliptic curves.

## 2 Injective Encodings

### 2.1 Definition

To fix ideas, and although it is not essential for our main purpose, let us first give a formal definition of what we mean by an "injective encoding".

Let us say that a *cyclic group family* $(\mathbb{G}_k)_{k\in\mathbb{N}}$ consists in the data of a sequence of integers $n_k \geq 1$ converging to infinity, a sequence of integers $s_k \geq 0$ that is at most polynomial in $\log n_k$, and for each $k$, an efficiently computable bijection $\sigma_k$ between the cyclic group $\mathbb{Z}/n_k\mathbb{Z}$ of order $n_k$ and a set $\mathbb{G}_k \subset \{0,1\}^{s_k}$ of bit strings of length $s_k$, as well as efficient algorithms:

$$\oplus_k \colon \{0,1\}^{s_k} \times \{0,1\}^{s_k} \to \{0,1\}^{s_k} \cup \{\bot\} \qquad \ominus_k \colon \{0,1\}^{s_k} \to \{0,1\}^{s_k} \cup \{\bot\}$$

which induce on the $\mathbb{G}_k$ the group addition and negation obtained by transport of structure via $\sigma_k$. Here, "efficient" means with a time complexity polynomial in $\log n_k$ (or equivalently, in $s_k$).

For example, if $q_k$ is an increasing sequence of positive primes, we can construct a cyclic group family $\mathbb{G}_k = \mathbb{F}_{q_k}^*$ with $n_k = q_k - 1$ and $s_k = O(\log q_k)$ by representing invertible elements in $\mathbb{F}_{q_k}$ as integers in $\{1, \ldots, q_k\}$ (themselves regarded as bit strings). Similarly, if $E$ is an elliptic curve over $\mathbb{Z}[1/N]$ with $N$ coprime with the $q_k$'s such that $E(\mathbb{F}_{q_k})$ is cyclic for all $k$, we have a cyclic group family $\mathbb{G}_k = E(\mathbb{F}_{q_k})$ with $n_k = q_k + O(\sqrt{q_k})$ and $s_k = O(\log q_k)$ obtained by representing curve points in e.g. affine coordinates (with a special string for the point at infinity).

Given such a cyclic group family $(\mathbb{G}_k)$ and a sequence of non negative integers $\ell_k$, we define an *$\ell_k$-injective encoding* to $(\mathbb{G}_k)$ be the data consisting of a pair of efficient, possibly randomized algorithms:

$$\mathscr{F}_k\colon \{0,1\}^{\ell_k} \to \mathbb{G}_k \subset \{0,1\}^{s_k} \qquad \mathscr{I}_k\colon \{0,1\}^{s_k} \to \{0,1\}^{\ell_k} \cup \{\bot\}$$

for all $k$, which satisfy $\mathscr{I}_k(\mathscr{F}_k(m)) = m$ for all $m \in \{0,1\}^{\ell_k}$ with overwhelming probability over the randomness involved. We will typically express $\ell_k$ in terms of $\nu_k = \lfloor \log_2 n_k \rfloor$, which is the optimal bound, in the sense that we clearly have $\ell_k \leq \nu_k$ for all $k$ by injectivity.

In what follows, the indices $k$, as well as references to sequences of integers and groups, will be omitted most of the time for simplicity's sake.

## 2.2 Some Simple, Optimal Examples

Let $p$ be an odd prime number. The bijection $[1, p-1] \to \mathbb{F}_p^*$ yields an obvious injective encoding to the multiplicative group $\mathbb{G} = \mathbb{F}_p^*$ which is optimal, in the sense that $\ell = \nu$.

Similarly, we obtain an optimal injective encoding to the group of squares $\mathbb{G} = (\mathbb{F}_p^*)^2 \subset \mathbb{F}_p^*$ from the bijection $[1, \frac{p-1}{2}] \to (\mathbb{F}_p^*)^2$ given by $x \mapsto x^2$. The inversion algorithm $\mathscr{I}$ then computes the unique square root of an element in $(\mathbb{F}_p^*)^2$ contained in $[1, \frac{p-1}{2}]$. This is sufficient to obtain IND–CPA ElGamal encryption in the group $(\mathbb{F}_p^*)^2$ when $p$ is a safe prime, assuming the Decisional Diffie–Hellman assumption in that group (though one typically wouldn't want to use it for efficiency reasons). On the other hand, it is not clear how to construct a close to optimal injective encoding to the subgroup of prime order $q$ in $\mathbb{F}_p^*$ when $p$ is a Diffie–Hellman prime $p = 2r \cdot q + 1$.

Some elliptic curve groups also have optimal injective encodings. This is for example the case for the supersingular elliptic curves given by an equation of the form:

$$E\colon y^2 = x^3 + b$$

over a field $\mathbb{F}_q$ with $q \equiv 2 \pmod 3$. Then, as observed e.g. by Boneh and Franklin [4], the map $\mathbb{F}_q \to E(\mathbb{F}_p) \setminus \{\infty\}$ given by $u \mapsto \big((u^2 - b)^{1/3}, u\big)$ is an efficient bijection, and its inverse is clearly efficient as well. This gives, again, an optimal injective encoding to $\mathbb{G} = E(\mathbb{F}_q)$. Similarly, the genus 1 case of the construction proposed in [10] provides an optimal injective encoding to supersingular elliptic curves of the form:

$$E\colon y^2 = x^3 + ax$$

over fields $\mathbb{F}_q$ with $q \equiv 3 \pmod 4$. However, we are not aware of any strictly optimal injective encoding to groups of points of ordinary elliptic curves, or even supersingular curves of embedding degree greater than 2.

## 2.3 Generic Injective Encodings

It is easy to construct $\ell_k$-injective encodings to any cyclic group family $(\mathbb{G}_k)$ provided that $\ell_k = O(\log \nu_k)$ (and of course $\ell_k \leq \nu_k$ for all $k$). Indeed, in that case, the set $\{0,1\}^{\ell_k}$ of elements to be encoded contains only polynomially many elements: therefore, $\mathscr{F}_k$ and $\mathscr{I}_k$ can be defined as mutually inverse dictionary lookups for each $k$, and still be efficient. For example, we can define $\mathscr{F}_k$ to be the restriction of $\sigma_k$ to $\{0,1,\dots,2^{\ell_k}-1\} \subset \mathbb{Z}/n_k\mathbb{Z}$ (coded as bit strings in the obvious way), and $\mathscr{I}_k$ as a series of $2^{\ell_k}$ successive comparisons. Moreover, $\mathscr{F}_k$ and $\mathscr{I}_k$ are generic, in the sense that they only require black-box access to $\mathbb{G}_k$ (see [15]).

On the other hand, if $\ell_k = \omega(\log \nu_k)$, then it is easy to see that $\mathscr{F}_k$ and $\mathscr{I}_k$ cannot be both generic for all $k$ if the $\mathscr{F}_k$'s are deterministic. Indeed, suppose that it were the case. Since it doesn't take any group element as input, $\mathscr{F}_k$ must be of the form:

$$\mathscr{F}_k(m) = \sigma_k\big(f_k(m)\big)$$

for some efficiently computable function $f_k\colon \{0,1\}^{\ell_k} \to \mathbb{Z}/n_k\mathbb{Z}$. Then, let $S = \mathscr{F}_k(\{0,1\}^{\ell_k})$ be the image of $\mathscr{F}_k$. In the terminology of Shoup [18], the algorithm $f_k \circ \mathscr{I}_k$ is a generic group algorithm for $\mathbb{Z}/n_k\mathbb{Z}$ on $\{0,1\}^{s_k}$ that computes the discrete logarithm $\sigma_k^{-1}(g)$ of any element $g \in S$ with overwhelming probability in $\mathrm{poly}(\nu_k)$ steps, regardless of the choice of the bijection $\sigma_k$. As a result, by Shoup's argument in op. cit., we must have $\#S = \mathrm{poly}(\nu_k)$: a contradiction.

This means that deterministic injective encodings from sets of superlogarithmic bit size must use the particular representation of individual group elements. We conjecture that no *probabilistic* generic $\omega(\log \nu)$-injective encoding exists either, although this seems less easy to establish.

## 2.4 Injective Encodings to Elliptic Curves

For groups of points of arbitrary (even ordinary) elliptic curves over finite prime fields, it is possible to construct $\ell$-injective encodings for much larger values $\ell$ than in the generic case. We describe one such construction here, which is more or less folklore.

Let $E$ be an elliptic curve over $\mathbb{F}_p$ ($p \geq 5$) in short Weierstrass form, and $\ell$ an integer such that $\ell \leq (1/2 - \varepsilon)\log_2 p$ for some fixed constant $\varepsilon \in (0, 1/2)$. We define the encoding algorithm $\mathscr{F}\colon \{0,1\}^{\ell} \to E(\mathbb{F}_p)$ as follows (this encoding is probabilistic: it is not a map). To compute $\mathscr{F}(m)$, pick a random integer $x$ in $[0, p-1]$ whose least significant $\ell$ bits coincide with $m$. If there are points in

$E(\mathbb{F}_p)$ of abscissa $x \bmod p$, return one of those (at most two) points; otherwise, start over. The inversion algorithm $\mathscr{I}$ then simply maps a point $(x, y) \in E(\mathbb{F}_p)$ to the bit string $m$ formed by the $\ell$ least significant bits of $x$.

To prove that this method works, it suffices to show that $\mathscr{F}$ terminates in expected polynomial time on any input $m$. We obtain the following, more precise result.

**Theorem 1.** *If $p$ is large enough, the expected number of iterations in $\mathscr{F}$ on any input is less than* $3$.

*Proof.* Let $P(m)$ be the success probability of $\mathscr{F}$ on input $m$ after a single iteration; in other words, $P(m)$ is the probability that a random integer $x$ in $[0, p-1]$ whose least significant $\ell$ bits coincide with $m$ is the abscissa of a point in $E(\mathbb{F}_p)$. Since for each such $x$ there are at most two corresponding points in $E(\mathbb{F}_p)$, we have:

$$P(m) \geq \frac{1}{2} \cdot \frac{\#\{(x,y) \in E(\mathbb{F}_p) \mid \textsc{lsb}_\ell(x) = m\}}{\#\{x \in [0, p-1] \mid \textsc{lsb}_\ell(x) = m\}} \tag{1}$$

where $\textsc{lsb}_\ell(x)$ denotes the bit string formed by the $\ell$ least significant bits of $x$. Clearly we have

$$\#\{x \in [0, p-1] \mid \textsc{lsb}_\ell(x) = m\} \leq 2^{-\ell} \cdot p.$$

On the other hand, the value $\#\{(x,y) \in E(\mathbb{F}_p) \mid \textsc{lsb}_\ell(x) = m\}$ can be estimated as in [9, §6]. It is the number of $\mathbb{F}_p$-points $(x, y)$ of $E$ such that $x/p$ is in a certain interval of $\mathbb{R}/\mathbb{Z}$ of length $\geq 2^{-\ell} \cdot (1 - 2/p)$ (because $x$ can be of the form $m + 2^\ell \cdot r$ at least for any $r \in [0, \lfloor p/2^\ell \rfloor - 1]$). But the values $x/p$ in $\mathbb{R}/\mathbb{Z}$ for $(x, y) \in E(\mathbb{F}_p)$ are close to equidistributed. More precisely, we know from Bombieri's bound on character sums [3] that for any nontrivial additive character $\psi$ of $\mathbb{F}_p$, we have:

$$T(\psi) = \left| \sum_{(x,y) \in E(\mathbb{F}_p) \setminus \{\infty\}} \psi(x) \right| \leq 4\sqrt{p}. \tag{2}$$

As a result, the (1-dimensional) Erdős–Turán–Koksma inequality [6, Th. 1.21] gives, for any interval $I \subset \mathbb{R}/\mathbb{Z}$ of length $L$ and any positive integer $H < p$:

$$\left| \frac{\#\{(x,y) \in E(\mathbb{F}_p) \setminus \{\infty\} \mid \frac{x}{p} \in I\}}{N} - L \right| \leq \frac{3}{H+1} + \frac{3}{N} \sum_{h=1}^{H} \frac{T(\psi_h)}{h}$$

where $\psi_h$ is the additive character $x \mapsto e^{2i\pi hx/p}$ and $N = \#E(\mathbb{F}_p) \setminus \{\infty\}$. Setting $H = \sqrt{p} - 1$, we get, in view of (2):

$$\begin{aligned} \#\{(x,y) \in E(\mathbb{F}_p) \setminus \{\infty\} \mid \frac{x}{p} \in I\} &\geq L \cdot N - \frac{3N}{\sqrt{p}} - 3 \cdot 4\sqrt{p} \log \sqrt{p} \\ &\geq L \cdot p - 2L\sqrt{p} - 3\sqrt{p} - 6 - 6\sqrt{p} \log p \\ &\geq L \cdot p - 12\sqrt{p} \log p \end{aligned}$$

since $|N - p| \leq 2\sqrt{p}$ by the Hasse bound. Plugging this estimate back into (1), we finally obtain:

$$P(m) \geq \frac{1}{2} \cdot \frac{2^{-\ell}(1 - 2/p)p - 12\sqrt{p}\log p}{2^{-\ell} \cdot p} = \frac{1}{2} - \frac{1}{p} - \frac{6 \log p}{p^{\varepsilon}}$$

since $\ell \leq (1/2 - \varepsilon)\log_2 p$. Hence, the expected number of iteration in $\mathscr{F}$ is $1/P(m) \leq 3$ for large enough $p$ as required. $\square$

Thus, we can construct $\ell$-injective encodings to elliptic curves over prime fields for $\ell = (1/2 - \varepsilon)\nu$: this is much better than the logarithmic bound we get in the generic case, but this still falls short of optimality by a constant factor greater than 2. It is conceivable that the same algorithm does in fact work with a larger $\ell$ still, possibly as large as $(1 - \varepsilon)\nu$ or even $\nu - \log^{O(1)} \nu$; we doubt that current results on the distribution of points on elliptic curves are sufficient to prove that the algorithm terminates on all inputs on those cases, however (though it might be possible to bound its complexity *on average* over all inputs $m$).

The only injective encoding to ordinary elliptic curves in the literature achieving a better bound is, to our knowledge, the one proposed by Farashahi in [8]. It applies to Hessian curves (i.e. elliptic curves with a rational point of order 3) over fields $\mathbb{F}_q$ with $q \equiv 2 \pmod 3$, and achieves $\ell = \nu - 1$, a single bit short of optimal. In the next sections, we construct a similar deterministic injective encoding to all ordinary elliptic curves over fields $\mathbb{F}_q$ with $q \equiv 3 \pmod 4$ with group order divisible by 4, also achieving $\ell = \nu - 1$.

## 3 Our New Elliptic Curve Encoding

### 3.1 Main Construction

As mentioned in the introduction, we now construct a new injective encoding for a large family of elliptic curves that are covered by certain hyperelliptic curves of genus 2.

More precisely, fix some finite field $\mathbb{F}_q$ with $q \equiv 3 \pmod 4$, and constants $c \in \mathbb{F}_q \setminus \{-1, 0, 1\}$, $\delta = \pm 1$. We consider the following hyperelliptic curve of genus 2:

$$H_c^{\delta} \colon y^2 = f(x) = \delta x^5 + \left(c^2 + \frac{1}{c^2}\right) \cdot x^3 + \delta x. \tag{3}$$

The main result of this paper can then be stated as follows.

**Theorem 2.** *The following properties hold.*

1. *In addition to the hyperelliptic involution $\tau\colon (x, y) \mapsto (x, -y)$, $H_c^{\delta}$ admits an additional involution $\sigma$ defined over $\mathbb{F}_q$ and given by $\sigma(x, y) = (1/x, y/x^3)$.*

2. *The quotient curve $H_c^\delta/\langle\sigma\rangle$ is an elliptic curve isomorphic to:*

$$E_c^\delta\colon y^2 = x^3 - 4\delta x^2 + \delta(c+\delta/c)^2 x. \tag{4}$$

*The quotient map $G\colon H_c^\delta \to E_c^\delta$ commutes with hyperelliptic involutions, in the sense that if we denote by $\tau'$ the involution of $E_c^\delta$ given by $(x,y)\mapsto(x,-y)$, we have $G\circ\tau = \tau'\circ G$.*

3. *There is a well-defined map $F\colon \mathbb{F}_q \to H_c^\delta(\mathbb{F}_q)$ given by*

$$F(t) = \Big(\chi_q(f(t))\cdot t,\ \chi_q\big(ct+\delta t^3/c\big)\sqrt{\chi_q(f(t))\cdot f(t)}\Big), \tag{5}$$

*where $\chi_q(\cdot)$ is the quadratic character of $\mathbb{F}_q^*$ (extended by zero to all of $\mathbb{F}_q$) and $\sqrt{\cdot}$ is the usual square root on the squares in $\mathbb{F}_q$ (namely exponentiation by $(q+1)/4$). This map $F$ satisfies, for all $t\in\mathbb{F}_q^*$:*

$$F(1/t) = \sigma\big(F(t)\big) \quad \text{and} \quad F(-t) = \tau\big(F(t)\big).$$

4. *Fix $I\subset\mathbb{F}_q$ a subset of $\mathbb{F}_q$ with $(q-1)/2$ elements such that $I\cap(-I)=\varnothing$ and $-1\notin I$, and let $I_0$ be the set obtained from $I$ by removing all elements of the form $\frac{1-t}{1+t}$ for $t$ a root of $f$, and adding $0$ and $1$. Then, the restriction to $I_0$ of the map $F_{\mathrm{inj}}\colon u\mapsto G\Big(F\big(\frac{1-u}{1+u}\big)\Big)$ is injective, and can be computed very efficiently as can its inverse (computing either of them costs one exponentiation in the base field and a few multiplications and divisions).*

*Proof.* The first claim is clear. To prove the second claim, the idea is to write the equation of $H_c^\delta$ in terms of a rational function that transforms in a simple way under $\sigma$, such as $t=\frac{1+x}{1-x}$, which satisfies $\sigma^* t = -t$. Concretely, we observe that (when $t$ is regarded as an indeterminate over $\mathbb{F}_q$):

$$\begin{aligned}(1+t)^6 f\left(\frac{1-t}{1+t}\right) &= \delta(1-t)^5(1+t) + \omega(1-t^3)(1+t)^3 + \delta(1-t)(1+t)^5\\ &= -(2\delta+\omega)t^6 - (10\delta-3\omega)t^4 + (10\delta-3\omega)t^2 + (2\delta+\omega).\end{aligned}$$

where $\omega = c^2+1/c^2$. From this relation, it is easily verified that $H_c^\delta$ is isomorphic to the curve:

$$H'\colon v^2 = -(2\delta+\omega)t^6 - (10\delta-3\omega)t^4 + (10\delta-3\omega)t^2 + (2\delta+\omega),$$

a pair of mutually inverse rational maps between $H_c^\delta$ and $H'$ being given by:

$$\begin{aligned} H_c^\delta &\longrightarrow H'\\ (x,y) &\longmapsto \left(\frac{1-x}{1+x},\ y\Big(\frac{2}{1+x}\Big)^3\right);\\ \left(\frac{1-t}{1+t},\ \frac{v}{(1+t)^3}\right) &\longleftarrow\!\shortmid (t,v).\end{aligned}$$

Moreover, the involution $\sigma$ on $H_c^\delta$ corresponds, under this isomorphism, to the involution $\sigma'\colon (x,y) \mapsto (-x,y)$ of $H'$, and hence $H_c^\delta/\langle\sigma\rangle \cong H'/\langle\sigma'\rangle$ is isomorphic to:

$$E'\colon v^2 = h(u) = -(2\delta+\omega)u^3 - (10\delta - 3\omega)u^2 + (10\delta - 3\omega)u + (2\delta+\omega).$$

Now, since $h(1-u) = (2\delta+\omega)x^3 - 16\delta x^2 + 16\delta x$, we see by applying the change of coordinates $(u,v) \mapsto (1-u,v)$ and then the scaling $(u,v) \mapsto \big((2\delta+\omega)u/4, (2\delta+\omega)v/8\big)$ that $E'$ is itself isomorphic to

$$E_c^\delta\colon y^2 = x^3 - 4\delta x^2 + \delta(2\delta+\omega)x$$

as required (this is the same as (4) since $2\delta+\omega = (c+\delta/c)^2$). Furthermore, the discriminant of this curve is:

$$\varDelta = 16(c+\delta/c)^4 \cdot \big(16 - 4\delta(c+\delta/c)^2\big) = -64\delta(c+\delta/c)^4 \cdot (c-\delta/c)^2,$$

which is necessarily non-zero since $c \neq \pm 1$ and $-1$ is a non quadratic residue. It follows that $E_c^\delta$ is indeed an elliptic curve. Finally, each of the maps in the diagram $H_c^\delta \to H' \to E' \to E_c^\delta$ commutes with hyperelliptic involutions, so the compose map $G$ does as well.

We now turn to the third claim. For any $t \in \mathbb{F}_q$, $\chi_q(f(t)) \cdot f(t)$ has a square image under $\chi_q$ so it is itself a square, and thus equation (5) correctly defines $F(t) = (x,y)$ as a point in $\mathbb{F}_q^2$. We have to check that it lies in $H_c^\delta(\mathbb{F}_q)$. Suppose first that $f(t) \neq 0$. In that case, we cannot have $ct + \delta t^3/c = 0$ since:

$$(ct + \delta t^3/c)\cdot(ct^2 + \delta/c) = c^2t^3 + \delta t + \delta t^5 + t^3/c^2 = f(t) \neq 0.$$

Therefore, the first factor in $y$ is $\pm 1$, and thus:

$$y^2 = \chi_q(f(t))\cdot f(t) = f\left(\chi_q(f(t))\cdot t\right) = f(x)$$

so that $F(t) \in H_c^\delta(\mathbb{F}_q)$ as required. On the other hand, if $f(t) = 0$, we get $x = y = 0$ and again $F(t) \in H_c^\delta(\mathbb{F}_q)$.

It remains to show that $F(1/t) = \sigma\big(F(t)\big)$ and $F(-t) = \tau\big(F(t)\big)$ for all $t \neq 0$. The latter is easy:

$$\begin{aligned} F(-t) &= \left(\chi_q(f(-t))\cdot(-t),\ \chi_q\big(-ct - \delta t^3/c\big)\sqrt{\chi_q(f(-t))\cdot f(-t)}\right) \\ &= \left(\chi_q(f(t))\cdot t,\ -\chi_q\big(ct + \delta t^3/c\big)\sqrt{\chi_q(f(t))\cdot f(t)}\right) = \tau\big(F(t)\big). \end{aligned}$$

To obtain the former, note that $f(1/t) = f(t)/t^6$. In particular, $f(t)$ and $f(1/t)$ have the same quadratic residue. Moreover, if we let $\alpha(t) = \chi_q\big(ct + \delta t^3/c\big)$, we have:

$$\alpha(t)\cdot\alpha(1/t) = \chi_q\big((ct + \delta t^3/c)\cdot(ct^2 + \delta/c)/t^3\big) = \chi_q\big(f(t)/t^3\big).$$

Now write $F(t) = (x, y)$ and $F(1/t) = (x', y')$. We have:

$$\begin{aligned}
x' &= \chi_q(f(t)) \cdot \frac{1}{t} = \frac{1}{x} \\
y' &= \alpha(1/t) \cdot \sqrt{\chi_q(f(t))\,\frac{f(t)}{t^6}} \\
&= \alpha(1/t) \cdot \sqrt{\chi_q(f(t))\,f(t)} \cdot \frac{1}{t^3}\chi_q\bigl(1/t^3\bigr) \\
&= \alpha(1/t) \cdot \alpha(t) y \cdot \frac{\chi_q\bigl(t^3\bigr)}{t^3} = y \cdot \chi_q(f(t))\,\frac{1}{t^3} = \frac{y}{x^3},
\end{aligned}$$

hence $F(1/t) = \sigma\bigl(F(t)\bigr)$ as required.

Regarding the fourth assertion of the theorem, the injectivity claim is a direct consequence of Lemma 1 below. The efficiency claim for $F_{\text{inj}}$ follows from the fact that $F$ can be computed at the cost of one exponentiation in the base field, some quadratic character evaluations[1] and a few multiplications, while $G$ is the simple rational function described explicitly above. Similarly, computing $G^{-1}$ costs one square root to lift a point from $E'(\mathbb{F}_q)$ back to $H'(\mathbb{F}_q)$ and a few arithmetic operations for the isomorphisms $E_c^\delta \cong E'$ and $H' \cong H_c^\delta$, whereas the inverse of $F$ (outside of the Weierstrass points of $H_c^\delta$) admits the following simple expression:

$$F^{-1}(x, y) = \alpha(x) \cdot \chi_q(y) \cdot x.$$

Indeed, if $(x, y) = F(t)$, we have $\alpha(x)\chi_q(y) = \alpha(x)\alpha(t) = \chi_q(xt) \cdot \chi_q\bigl(c + \delta x^2/c\bigr)^2 = \chi_q(xt)$ since $t^2 = x^2$. Hence the claim on the efficiency of $F_{\text{inj}}^{-1}$. □

**Lemma 1.** *Let $S \subset \mathbb{F}_q$ be any subset of $\mathbb{F}_q$ containing no root of $f$, and such that $S \cap S^{-1} = \varnothing$ (i.e. for all $x \in S$, $1/x \notin S$). Then, the restriction of $G \circ F$ to $S$ is injective. Moreover, the result still holds if we replace $S$ by $S \cup \{0, 1\}$.*

*Proof.* Consider $t, t' \in S$ such that $G\bigl(F(t)\bigr) = G\bigl(F(t')\bigr)$. We must have either $F(t) = F(t')$ or $F(t) = \sigma\bigl(F(t')\bigr) = F(1/t')$.

In the latter case, we see in particular that the first coordinates of $F(t)$ and $F(1/t')$ coincide, so that $t = \pm 1/t'$. By definition of $S$, $t = 1/t'$ is excluded, so we must have $t = -1/t'$. Now since $G$ commutes with hyperelliptic involutions, we can write:

$$G\bigl(F(t')\bigr) = G\bigl(F(-1/t')\bigr) = G\bigl(\tau F(1/t')\bigr) = \tau' G\bigl(\sigma F(t')\bigr) = \tau' G\bigl(F(t')\bigr).$$

Therefore, $G\bigl(F(t')\bigr)$ is a Weierstrass point on $E_c^\delta$. Given the expression of $G$, this implies that $F(t')$ is a Weierstrass point on $H_c^\delta$, and hence that $t'$ is a root of $f$, which is a contradiction.

If on the other hand $F(t) = F(t')$, we see in particular by comparing the first coordinates of $F(t)$ and $F(t')$ that $t' = \pm t$. But since $S$ contains no root of $f$, $F(t)$ is not a Weierstrass point, so it is not equal to its image $F(-t)$ under the

[1] In fact, using the techniques from [10], they can be optimized away.

hyperelliptic involution $\tau$. Hence $t' = -t$ is impossible, and we must have $t = t'$ as required.

Turning to the second claim, we compute the images of 0 and 1 under $G \circ F$. We have $G \circ F(0) = G\big((0,0)\big) = (0,0) \in E_c^\delta(\mathbb{F}_q)$, and similarly, since $f(1) = (c+\delta/c)^2$, we find that $G \circ F(1) = G\big((1, c+\delta/c)\big) = \big((c+\delta/c)^2/4, (c+\delta/c)^3/8\big) \in E_c^\delta(\mathbb{F}_q)$. In particular, these images are distinct. Moreover, it follows from the above that for all $t \in S$, $G\big(F(t)\big)$ is never a Weierstrass point on $E_c^\delta$, and hence is always distinct from $G\big(F(0)\big)$. Finally, if there was some $t \in S$ such that $G\big(F(t)\big) = G\big(F(1)\big)$, then, using the same argument as above, we would have $t = \pm 1$ (or $1/t = \pm 1$, which is equivalent), and this is impossible since $S \cap S^{-1} = \varnothing$. □

## 3.2 Description of the Target Curves

The result of Theorem 2 is an injective encoding $F_{\text{inj}}$ to any elliptic curve of the form $E_c^\delta$. Its range $I_0$ is of cardinality exactly $(q-1)/2 + \delta$. Indeed, we can write $f(x)$ for $x \neq 0$ as $x^3 \cdot (cx^2 + \delta/c) \cdot (c/x^2 + \delta/c)$. When $\delta = +1$, none of the factors can vanish for $x \neq 0$, so 0 is the only root of $f$. Therefore, the range $I_0$ of $F_{\text{inj}}$ is of cardinality $(q+1)/2$; when $q$ is prime, we can take the interval $[0, (q-1)/2]$. On the other hand, when $\delta = -1$, the roots of $f$ are $0, \pm c, \pm 1/c$, and $I_0$ is then of cardinality $(q+1)/2 - 2 = (q-3)/2$; when $q$ is prime, it is the interval $[0, (q-1)/2]$ from which one has removed $\pm t, \pm 1/t$ where $t = \frac{1-c}{1+c}$.

In both cases, we see that the size of the set from which we encode is a single bit less than the cardinality $\#E_c^\delta(\mathbb{F}_q) = q + O(\sqrt{q})$ of the target group. Hence, we do get a deterministic $(\nu - 1)$-injective encoding as stated.

It is desirable to have a simple description of the class of curves $E_c^\delta$ for which we obtain this encoding. It is given by the following theorem.

**Theorem 3.** *Denoting $E_c^\delta$ by $E_c^+$ or $E_c^-$ for $\delta = 1$ and $\delta = -1$ respectively, the following hold:*

1. *The point $(0,0)$ is the only rational point of exact order 2 on $E_c^+$ and it is divisible by 2. In particular, the rational 4-torsion subgroup of $E_c^+$ is equal to $\mathbb{Z}/4\mathbb{Z}$.*
2. *All three points of exact order 2 on $E_c^-$ are rational, but $(0,0)$ is not divisible by 2. In particular, $E_c^-$ has full rational 2-torsion, and the rational 4-torsion of is equal to $(\mathbb{Z}/2\mathbb{Z})^2$ or $\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/4\mathbb{Z}$, depending on whether one of the other points of order 2 is divisible by 2.*
3. *Any ordinary elliptic curve over $\mathbb{F}_q$ with rational 4-torsion equal to $\mathbb{Z}/4\mathbb{Z}$ is isomorphic to $E_c^+$* or to its twist *for some $c$.*
4. *Any ordinary elliptic curve over $\mathbb{F}_q$ with full rational 2-torsion is isomorphic to $E_c^-$* or to its twist *for some $c$.*

*Proof. Statements 1 and 2.* The curve $E_c^\delta$ obviously has a rational point of exact order 2, namely $(0,0)$. When $\delta = +1$, it is the only one; indeed, the trinomial $x^2 - 4x + (c+1/c)^2$ has discriminant $16 - 4(c+1/c)^2 = -4(c-1/c)^2$ which is a non quadratic residue. On the other hand, if $\delta = -1$, all three points of exact order 2 are rational, since $x^2+4x-(c-1/c)^2$ has discriminant $16+4(c-1/c)^2 = 4(c+1/c)^2$ which is a square.

Furthermore, there is a rational point $P$ such that $[2]P = (0,0)$ if and only if $\delta = +1$. To see that, it suffices to show that there is a line through $(0,0)$ which is tangent to the curve, since the intersection point will clearly satisfy the requirement. Now if $y = tx$ is a line through $(0,0)$, the other intersection points with the curve have their abscissa given by $t^2x = x^2 - 4\delta x + \delta(c+\delta/c)^2$, and the line is tangent when the discriminant of this quadratic equation vanishes, i.e. when $t$ satisfies:

$$(4\delta + t^2)^2 = 4\delta\left(c + \frac{\delta}{c}\right)^2 .$$

There is no solution when $\delta = -1$ since the right-hand side is not square. On the other hand, when $\delta = +1$, this is equivalent to:

$$t^2 = -4 \pm 2\left(c + \frac{1}{c}\right)$$

and this equation has a solution for one of the two possible signs, because $\bigl(-4 + 2(c+1/c)\bigr)\cdot\bigl(-4 - 2(c+1/c)\bigr) = -4(c-1/c)^2$ is a non quadratic residue, and hence exactly one of the factors must be square.

Thus, in all cases, we see that the curve admits a rational subgroup of order 4. In fact, the rational 4-torsion is of order 4 or 8: namely $\mathbb{Z}/4\mathbb{Z}$ when $\delta = +1$, and $(\mathbb{Z}/2\mathbb{Z})^2$ or $\mathbb{Z}/2\mathbb{Z}\times\mathbb{Z}/4\mathbb{Z}$ when $\delta = -1$ depending on whether one of the points of order 2 other than $(0,0)$ is divisible by two. This completes the proof of statements 1 and 2.

*Statement 3.* Conversely, we now prove that, *up to a quadratic twist*, any ordinary elliptic curve over $\mathbb{F}_q$ with a point of order 4 and only one point of order 2 is isomorphic to $E_c^+$ for some $c$. Indeed, let $E$ be any such elliptic curve. We can put $E$ in Weierstrass form, translate so that the point of order 2 is $(0,0)$, and scale the coordinates to get an equation of the form:

$$E\colon y^2 = x^3 \pm 4x^2 + ax$$

for some $a \in \mathbb{F}_q$, with $a \neq 0, 4$ since the right-hand side must have no double root. Note that the nontrivial quadratic twist of $E$ has the same equation, only with the sign of the coefficient of $x^2$ reversed.

Since there is a single point of order 2, the discriminant $16-4a$ of the trinomial $x^2 \pm 4x + a$ must be a non quadratic residue. Hence, $a-4$ is a square. Moreover, $(0,0)$ is divisible by two: therefore, there exists a $t$ such that the line of slope $t$ through $(0,0)$ is tangent to the curve. This $t$ is such that the equation $t^2x = x^2 \pm 4x + a$ has a double root, so we must have $(-t^2 \pm 4)^2 - 4a = 0$, hence $a$ is a square. And the discriminant of the trinomial $c^2 - \sqrt{a}\cdot c + 1$ is $a - 4$, so there is a $c \in \mathbb{F}_q \setminus \{0, \pm 1\}$ such that $a = (c+1/c)^2$. This shows that $E$ is either $E_c^+$ or its quadratic twist as required.

*Statement 4.* Finally, consider any elliptic curve $E$ over $\mathbb{F}_q$ with full rational 2-torsion. As above, we can put $E$ in the form:

$$E\colon y^2 = x^3 \pm 4x^2 + ax \tag{6}$$

for some $a \in \mathbb{F}_q$ with $a \neq 0, 4$, and since the right-hand side splits in linear factors, $4 - a$ is a square. Assume for the moment that $a$ is a non quadratic residue, then $E$ is isomorphic to either $E_c^-$ or its twist. Indeed, $-a$ is then a square, and the discriminant of the trinomial $c^2 - \sqrt{-a} \cdot c - 1$ is $-a + 4$ which is a square as well; hence, we can find $c \in \mathbb{F}_q \setminus \{0, \pm 1\}$ such that $a = -(c - 1/c)^2$, as required.

To complete the proof, we need to show that we can always find a Weierstrass equation (6) for $E$ such that $a$ is a non quadratic residue. To see this, first observe that if we start from a Weierstrass equation of the form

$$y^2 = x(x - \lambda)(x - \mu) \tag{7}$$

for $E$ (which certainly exists) and scale the coefficients to get (6), the scaling factor $s$ satisfies $\lambda + \mu = \pm 4s^2$ and $\lambda\mu = as^4$, so that:

$$a = \frac{16\lambda\mu}{(\lambda + \mu)^2}.$$

Now clearly, starting from (7), we can translate the origin to one of the other two points of order 2, and get one of the other two Weierstrass forms:

$$y^2 = x'(x' + \lambda)(x' + \lambda - \mu) \quad \text{or} \quad y^2 = x''(x'' + \mu)(x'' + \mu - \lambda).$$

These correspond to the canonical form (6) with the coefficient of $x$ equal to:

$$a' = \frac{16(-\lambda)(-\lambda + \mu)}{(-2\lambda + \mu)^2}, \quad \text{resp.} \quad a'' = \frac{16(-\mu)(-\mu + \lambda)}{(-2\mu + \lambda)^2}.$$

But at least one of $a$, $a'$ and $a''$ must be a non quadratic residue, since:

$$\chi_q(a \cdot a' \cdot a'') = \chi_q(\lambda\mu \cdot (-\lambda)(-\lambda + \mu) \cdot (-\mu)(-\mu + \lambda)) = -1.$$

This concludes the proof. □

Note that elliptic curves with rational 4-torsion equal to $\mathbb{Z}/4\mathbb{Z}$ are birational to Edwards curves $x^2 + y^2 = 1 + dx^2y^2$ with non square $d$ [2]. Bernstein and Lange showed that these curves are quite interesting for computation and cryptography, as they admit a complete addition law, and admit the fastest arithmetic known to date. Similarly, curves with full rational 2-torsion are also isomorphic to curves with fast arithmetic and unified addition laws, namely twisted Huff curves [13]. Together, they comprise all ordinary curves with order divisible by 4.

### 3.3 Mapping to the Twist

The previous paragraph suggests that if $E$ is an elliptic curve with order divisible by 4, then we know an injective encoding to *either* $E(\mathbb{F}_q)$ itself or to its nontrivial quadratic twist. But we can in fact do better and map to $E(\mathbb{F}_q)$ itself.

Indeed, it is classical (see e.g. [16] or [5, Ch. 14]) that $H_c^\delta$ does not only cover the elliptic curve $E'\colon v^2 = h(u)$ given by the quotient by $\sigma$ (using the notations of the proof of Theorem 2), but also $v^2 = u^3h(1/u)$ given by the quotient by $\sigma\tau$. Moreover, we have $u^3h(1/u) = -h(u)$, so that $H_c^\delta/\langle\sigma\tau\rangle$ is the nontrivial quadratic twist of $E_c^\delta$.

It is easy to adapt the construction of Theorem 2 to obtain a similar injective function to $H_c^\delta/\langle\sigma\tau\rangle$, and hence a $(\nu-1)$-injective encoding to the twists of the curves $E_c^\delta$. We conclude:

**Theorem 4.** *Let $E$ be an ordinary elliptic curve over a finite field $\mathbb{F}_q$ with $q \equiv 3 \pmod 4$, such that 4 divides $\#E(\mathbb{F}_q)$. Then there is an efficient, efficiently invertible injective encoding to $E(\mathbb{F}_q)$ from an interval of cardinality $q/2 + O(1)$ (i.e. a $(\nu-1)$-injective encoding, in the terminology of §2.1). Both the encoding and its inverse can be computed with one exponentiation in $\mathbb{F}_q$ and a few multiplications and divisions.*

In Appendix A, we give pseudocode for the encoding to and decoding from $E_c^+$. The other cases (viz. $E_c^-$ and the twists of $E_c^\pm$) are treated similarly.

## 4 Conclusion

In this paper, we proposed an efficient injective encoding with almost optimally large image for a new class of elliptic curves including important examples like Edwards curves. The only previous construction in that direction was for Hessian curves.

Note that, from a cryptographic perspective, this does not completely solve the problem of constructing an encoding for ElGamal encryption, as the curves we encode to have a small subgroup which can reveal information about the message (i.e. ElGamal is one-way but not semantically secure in this setting). This is similar to the situation of ElGamal in multiplicative groups $\mathbb{F}_p^*$ when $p$ is not a safe prime. Similarly, since Lindell's UC commitment scheme works in prime order groups, our construction is *a priori* not applicable to that setting.

However, we believe that the possibility of encoding messages as elliptic curve points can be of sufficient interest to protocol designers that designing around this cofactor limitation might be worthwhile.

## References

1. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-preserving signatures and commitments to group elements. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 209–236. Springer, Heidelberg (2010)

2. Bernstein, D.J., Lange, T.: Faster addition and doubling on elliptic curves. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 29–50. Springer, Heidelberg (2007)
3. Bombieri, E.: On exponential sums in finite fields. In: Les Tendances Géom. en Algèbre et Théorie des Nombres, pp. 37–41. Éditions du CNRS (1966)
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. Cassels, J., Flynn, E.: Prolegomena to a middlebrow arithmetic of curves of genus 2. London Mathematical Society Lecture Note Series, vol. 230. Cambridge University Press (1996)
6. Drmota, M., Tichy, R.F.: Sequences, discrepancies and applications. Springer (1997)
7. El Gamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory 31(4), 469–472 (1985)
8. Farashahi, R.R.: Hashing into Hessian curves. In: Nitaj, A., Pointcheval, D. (eds.) AFRICACRYPT 2011. LNCS, vol. 6737, pp. 278–289. Springer, Heidelberg (2011)
9. Farashahi, R.R., Fouque, P.-A., Shparlinski, I.E., Tibouchi, M., Voloch, J.F.: Indifferentiable deterministic hashing to elliptic and hyperelliptic curves. Math. Comp. 82, 491–512 (2013)
10. Fouque, P.-A., Tibouchi, M.: Deterministic encoding and hashing to odd hyperelliptic curves. In: Joye, M., Miyaji, A., Otsuka, A. (eds.) Pairing 2010. LNCS, vol. 6487, pp. 265–277. Springer, Heidelberg (2010)
11. Icart, T.: How to hash into elliptic curves. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 303–316. Springer, Heidelberg (2009)
12. Joux, A., Vitse, V.: Cover and decomposition index calculus on elliptic curves made practical. Application to a previously unreachable curve over $\mathbb{F}_{p^6}$. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 9–26. Springer, Heidelberg (2012)
13. Joye, M., Tibouchi, M., Vergnaud, D.: Huff's model for elliptic curves. In: Hanrot, G., Morain, F., Thomé, E. (eds.) ANTS-IX. LNCS, vol. 6197, pp. 234–250. Springer, Heidelberg (2010)
14. Lindell, Y.: Highly-efficient universally-composable commitments based on the DDH assumption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 446–466. Springer, Heidelberg (2011)
15. Maurer, U.M.: Abstract models of computation in cryptography. In: Smart, N.P. (ed.) Cryptography and Coding 2005. LNCS, vol. 3796, pp. 1–12. Springer, Heidelberg (2005)
16. Paulhus, J.: Decomposing Jacobians of curves with extra automorphisms. Acta Arith. 132(3), 231–244 (2008)
17. Shallue, A., van de Woestijne, C.E.: Construction of rational points on elliptic curves over finite fields. In: Hess, F., Pauli, S., Pohst, M. (eds.) ANTS 2006. LNCS, vol. 4076, pp. 510–524. Springer, Heidelberg (2006)
18. 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)

## A Pseudocode for the Encoding to $E_c^+$

We give formulas for computing the injective encoding to $E_c^+(\mathbb{F}_q)$ both in short Weierstrass form and in Edwards form. Fix a subset $I_0 \subset \mathbb{F}_q \backslash \{-1\}$ of cardinality $(q+1)/2$ such that $I \cap (-I) = \{0\}$. We can for example pick $I_0 = [0, (q-1)/2]$ if $q$ is prime.

*Short Weierstrass form.* The image $(x, y)$ of $u \in I_0$ is obtained with the $\textsc{Encode}_{E_c^+}$ algorithm below, and decoding in carried out with the inverse algorithm $\textsc{Decode}_{E_c^+}$.

```
function Encode_{E_c^+}(u)
    $t \leftarrow (1-u)/(1+u)$
    $f \leftarrow t^5 + (c^2 + 1/c^2) \cdot t^3 + t$
    $\varepsilon \leftarrow \chi_q(f)$
    $\alpha \leftarrow \chi_q\big(ct + t^3/c\big)$
    $x_H \leftarrow \varepsilon \cdot t$
    $y_H \leftarrow \alpha \cdot \sqrt{\varepsilon \cdot f}$
    $u_{E'} \leftarrow \big(\frac{1-x_H}{1+x_H}\big)^2$
    $v_{E'} \leftarrow y_H\big(\frac{2}{1+x_H}\big)^3$
    $x \leftarrow (c + 1/c)^2 \cdot (1 - u_{E'})/4$
    $y \leftarrow (c + 1/c)^2 \cdot v_{E'}/8$
    return $(x, y)$
```

```
function Decode_{E_c^+}(x, y)
    $u_{E'} \leftarrow 1 - 4x/(c + 1/c)^2$
    $v_{E'} \leftarrow 8y/(c + 1/c)^2$
    $u_{H'} \leftarrow \sqrt{u_{E'}}$
    $x_H \leftarrow \frac{1-u_{H'}}{1+u_{H'}}$
    $y_H \leftarrow \frac{v_{E'}}{(1+u_{H'})^3}$
    $\alpha \leftarrow \chi_q\big(cx_H + x_H^3/c\big)$
    $t \leftarrow \alpha \cdot \chi_q(y_H) \cdot x_H$
    $u \leftarrow (1-t)/(1+t)$
    if $u \notin I_0$ then
        $u \leftarrow -u$
    return $u$
```

A number of optimizations of these algorithms are possible: for example, the decoding function only uses the quadratic character of $y_H$, so it is not necessary to compute $y_H$ in full; similarly, one can speed up the first part of the encoding algorithm by using the implementation techniques from [10] and noticing that $\alpha = \chi_q(c/t + t/c)$. Such improvements, however, only marginally affect the running time, which is dominated in both cases by the square root evaluation, so we chose to closely follow the steps of §3.1.

*Edwards form.* Clearly, $E_c^+$ and $E_{-c}^+$ are identical curves, so we may assume without loss of generality that $c$ is of the form $2s^2$. Then, consider the birational transformation $(X, Y) \mapsto (x, y)$ given by:

$$x = \left(c + \frac{1}{c}\right)\frac{1+Y}{1-Y} \qquad \text{and} \qquad \frac{y}{x} = \frac{c-1}{sX}.$$

It maps $E_c^+$ to the curve given by the equation:

$$\begin{aligned}
\left(\frac{c-1}{sX}\right)^2 &= x - 4 + \left(c + \frac{1}{c}\right)^2 \frac{1}{x} \\
2\frac{(c-1)^2}{cX^2} &= \left(c + \frac{1}{c}\right) \cdot \left(\frac{1+Y}{1-Y} + \frac{1-Y}{1+Y}\right) - 4 \\
\frac{(c-1)^2}{X^2} &= (c^2+1)\frac{1+Y^2}{1-Y^2} - 2c = \frac{(c-1)^2 + (c+1)^2 Y^2}{1-Y^2} \\
1 - Y^2 &= X^2 \cdot \left(1 + \left(\frac{c+1}{c-1}\right)^2 Y^2\right)
\end{aligned}$$

which is exactly the Edwards curve $\mathscr{E}_d\colon X^2 + Y^2 = 1 + dX^2Y^2$ for:

$$d = -\left(\frac{c+1}{c-1}\right)^2 .$$

Since we constrained $c$ to be of the form $2s^2$, this shows that about half of all Edwards curves $\mathscr{E}_d$ with non square $d$ are isomorphic to some $E_c^+$ (the other half being isomorphic to the twists of those).

We can easily encode to and decode from $\mathscr{E}_d(\mathbb{F}_q)$ using the birational transformation described above. This gives the following algorithms.

```
1: function Encode_{$\mathscr{E}_d$}($u$)
2:     $(x, y) \leftarrow \textsc{Encode}_{E_c^+}(u)$
3:     $X \leftarrow \frac{c-1}{s} \cdot \frac{x}{y}$
4:     $Y \leftarrow \frac{x+c+1/c}{x-c-1/c}$
5:     return $(X, Y)$
```

```
1: function Decode_{$\mathscr{E}_d$}($X, Y$)
2:     $x \leftarrow (c + 1/c) \cdot \frac{1+Y}{1-Y}$
3:     $y \leftarrow \frac{c-1}{s} \cdot \frac{x}{X}$
4:     $u \leftarrow \textsc{Decode}_{E_c^+}(x, y)$
5:     return $u$
```

# Membership Encryption and Its Applications[⋆]

Fuchun Guo[1], Yi Mu[1], Willy Susilo[1,⋆⋆], and Vijay Varadharajan[2]

[1] Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Wollongong, Australia
{fg278,ymu,wsusilo}@uow.edu.au

[2] Information and Networked Systems Security Research
Department of Computing, Faculty of Science
Macquarie University, Sydney, Australia
vijay.varadharajan@mq.edu.au

**Abstract.** We propose a new encryption primitive called *Membership Encryption*. Let $\mathcal{P}(\mathbf{G})$ be a privacy-preserving token on a group attribute/identity $\mathbf{G}$, such that given $\mathcal{P}(\mathbf{G})$ it is hard to know the attributes in $\mathbf{G}$. In this membership encryption, if an encryption takes as input an attribute $A$ and the token $\mathcal{P}(\mathbf{G})$, the decryption requires holding the membership $A \in \mathbf{G}$, i.e., $A$ belongs to this group attribute. Membership encryption is applicable in constructing membership proof $A \in \mathcal{P}(\mathbf{G})$ with privacy preserving on group attribute and the membership. Membership encryption can be also utilized to construct an efficient two-round $K$-out-of-$N$ oblivious transfer protocol. In this paper, we construct a provably secure membership encryption where the group token $\mathcal{P}(\mathbf{G})$ is constant-size with maximum number accountability on attributes. Using our scheme, the proposed oblivious transfer protocol exhibits the nice feature of $O(1)$ communication cost for any $K$ from receiver to sender, and $O(N)$ communication cost from sender to receiver.

## 1 Introduction

**Membership Proof.** Proving that an attribute $A$ belongs to a group attribute $\mathbf{G}$, denoted by group membership $A \in \mathbf{G}$, is useful and non-trivial especially when privacy protections are essential. Let $\mathcal{P}(O)$ denote privacy protection (e.g. commitment) on the object $O$, such that given $\mathcal{P}(O)$ it is hard to know the object $O$. The privacy-preserving membership proof falls into two different cases:

- $\mathcal{P}(A) \in \mathbf{G}$. The verifier knows the token $\mathcal{P}(A)$ and all attributes in $\mathbf{G}$. The prover wants to prove that the attribute in $\mathcal{P}(A)$ belongs to $\mathbf{G}$ without leaking the real attribute $A$ to the verifier. Assuming that each attribute is an individual, this membership proof is towards privacy protection on the involved individual. We found the technique called set membership proof [11,8,6] is proposed for $\mathcal{P}(A) \in \mathbf{G}$.

[⋆] This work is supported by ARC Discovery Grant DP110101951.
[⋆⋆] W. Susilo is supported by ARC Future Fellowship FT0991397.

– $A \in \mathcal{P}(\mathbf{G})$. The verifier knows the attribute $A$ and the token $\mathcal{P}(\mathbf{G})$. The prover wants to prove that the group attribute in $\mathcal{P}(\mathbf{G})$ contains $A$ without leaking other attributes in $\mathbf{G}$ to the verifier. This membership proof is aiming at protecting the privacy of non-involved individuals. We found the technique called accumulator with witness [3,2,17,14,7,1] can be seen as a membership proof for $A \in \mathcal{P}(\mathbf{G})$.

Membership proof is useful in those privacy-preserving applications (see [11,8,6,3,2,17,14,7,1]), where $\mathcal{P}(O)$ instead of $O$ is certified for privacy purpose, and the prover wants to prove that the certified $\mathcal{P}(O)$ satisfies some membership.

**Motivation.** In this work, we extend the membership by proof to membership by encryption. We are interested in exploring the notion of *membership encryption*. Let $\mathbf{G} = \{A_1, A_2, \cdots, A_k\}$ be a finite set of group attribute, and $\mathcal{P}(\mathbf{G})$ denote the privacy-preserving group $\mathbf{G}$. Following the membership proof $A \in \mathcal{P}(\mathbf{G})$, the membership encryption $A \in \mathcal{P}(\mathbf{G})$ is defined as follows: when the encryption takes as input $A$ and $\mathcal{P}(\mathbf{G})$, the decryption requires holding the membership $A \in \mathbf{G}$.

We focus on the membership encryption $A \in \mathcal{P}(\mathbf{G})$ only as it can be naturally transferred into the membership encryption $\mathcal{P}(A) \in \mathbf{G}$, when $\mathcal{P}(\cdot)$ contains only one attribute. For example, let $\mathbf{G} = \{A_1, A_2\}$. To generate a membership encryption $\mathcal{P}(A) \in \{A_1, A_2\}$, we run membership encryption $A_1 \in \mathcal{P}(A)$ and $A_2 \in \mathcal{P}(A)$ on the same message and $R$, where $R = \{r_1, r_2\}$ and $r_i$ is the randomness for $A_i \in \mathcal{P}(A)$ encryption. It is not hard to verify that this is equivalent to the membership encryption $\mathcal{P}(A) \in \mathbf{G}$.

**Encryption vs Proof.** Membership encryption is more powerful compared to membership proof in terms of three reasons. Firstly, a membership proof $A \in \mathcal{P}(\mathbf{G})$ cannot be converted into a membership encryption, but a successful decryption of membership encryption with $A$ and $\mathcal{P}(\mathbf{G})$ as input naturally implies the membership $A \in \mathcal{P}(\mathbf{G})$. Secondly, given a membership proof, the verifier might be able to compromise the privacy of $\mathcal{P}(\mathbf{G})$ to others by publishing the membership proof $A \in \mathcal{P}(\mathbf{G})$. While the membership proof from membership encryption is non-transferable. Finally, considering the scenario that Alice would send a message to Bob if he can prove the membership $A \in \mathcal{P}(\mathbf{G})$. Using the membership proof, Bob needs to generate the proof first and then Alice sends messages to Bob after checking the proof, which costs two separated steps. Membership encryption combines the two steps into one, which improves the communication efficiency.

Membership encryption is also useful in other applications. One of them is the oblivious transfer protocol [19]. Suppose there are $N$ messages $M_1, M_2, \cdots, M_N$, and a receiver wants to get part of them without leaking her/his choice to the message owner (sender). Using the membership encryption, the receiver generates $\mathcal{P}(C)$ and sends it to sender, where $C \subseteq \{1, 2, \cdots, N\}$ is the receiver's choice. The sender then encrypts message $M_i$ with the index $i$ and $\mathcal{P}(C)$. If $i \in C$, the receiver can decrypt the message $M_i$; otherwise $i \notin C$, the receiver will not be able to extract $M_i$. Suppose the number of choices is accountable from $\mathcal{P}(C)$, we will obtain a two-round $K$-out-of-$N$ protocol for any $K$ from the membership encryption $A \in \mathcal{P}(\mathbf{G})$.

**Contributions.** We propose a new encryption primitive called *membership encryption* where the decryption satisfies the privacy-preserving group membership $A \in \mathcal{P}(\mathbf{G})$. To be precise, in our membership encryption definition, $\mathcal{P}(\mathbf{G})$ is generated from the group attribute $\mathbf{G}$ and a secret token $\mathbf{S}$. If the encryption takes as input $A$ and $\mathcal{P}(\mathbf{G})$, the decryption is successful if and only if the decryptor knows $(\mathbf{G}, \mathbf{S})$ and $A \in \mathbf{G}$ is true.

We construct a provably secure membership encryption, which exhibits the following nice features.

- The group token $\mathcal{P}(\mathbf{G})$ is constant-size and independent of the number of attributes in $\mathbf{G}$.
- The upper bound attribute number in $\mathcal{P}(\mathbf{G})$ is accountable.
- The ciphertext is constant-size and dependent on the length of security parameter only.

We show how to apply membership encryption in constructing a two-round $K$-out-of-$N$ oblivious transfer protocol $\mathsf{OT}_\mathsf{N}^\mathsf{K}$ for any $1 \leq K \leq N$. Our protocol satisfies the security model defined in [10]. In our protocol, messages from receiver to sender are the group token $\mathcal{P}(\mathbf{G})$ only and messages from sender to receiver are $N$ constant-size ciphertexts. Using our proposed scheme, the communication cost from receiver to sender is $O(1)$ or constant-size, and communication cost from sender to receiver is $O(N)$ or linear in $N$. This is the first two-round $\mathsf{OT}_\mathsf{N}^\mathsf{K}$ protocol with the least communication cost compared to existing two-round oblivious transfer protocols [16,18,10,9].

**Roadmap.** The rest of this paper is organized as follows. We give the definition and security models of membership encryption in Section 2. Our construction is proposed in Section 3 with the security proof in Section 4. We show how to apply membership encryption to the construction of two-round $K$-out-of-$N$ oblivious transfer protocols in Section 5. In the final section, we conclude this paper.

## 2 Membership Encryption

### 2.1 Description of Membership Encryption

A membership encryption $A \in \mathcal{P}(\mathbf{G})$ with maximum number accountability on group attribute consists of the following five algorithms:

**Setup:** Taking as input a security parameter $1^\lambda$, an integer $n$ and all attributes $\{A_1, A_2, \cdots, A_n\}$, the setup algorithm generates the system parameter $SP$. Here, $n$ denotes the upper bound attribute number of group tokens.

**GroupGen:** Taking as input the system parameter $SP$ and a group attribute $\mathbf{G} = \{A_1, \cdots, A_k\}$ $(1 \leq k \leq n)$, the group token generation algorithm returns the token $\mathcal{P}(\mathbf{G})$ and the secret key $\mathbf{S}$.

**Verify:** Taking as input $\mathcal{P}(\mathbf{G})$ and an integer $k$, the verification algorithm returns true if the attribute number in $\mathcal{P}(\mathbf{G})$ satisfying $|\mathcal{P}(\mathbf{G})| \leq k$; otherwise, outputs false.

**Encrypt:** Taking as input the system parameter $SP$, an attribute $A$, a group token $\mathcal{P}(\mathbf{G})$ and a message $M$, the encryption algorithm returns a ciphertext $C$ of $M$. We define the ciphertext as $C \leftarrow \mathsf{ME}[A, \mathcal{P}(\mathbf{G}), M]$.

**Decrypt:** Taking as input the attribute $A$, the group attribute $\mathbf{G}$, the secret key $\mathbf{S}$ and the ciphertext $C$, the decryption algorithm returns the message $M$ or $\perp$. We define the decryption as $\{M, \perp\} \leftarrow \mathsf{MD}[C, \mathbf{G}, \mathbf{S}]$.

*Correctness:* The membership encryption must satisfy that for any system parameter $SP$, group token $(\mathcal{P}(\mathbf{G}), \mathbf{G}, \mathbf{S})$ and ciphertext $\mathsf{ME}[A, \mathcal{P}(\mathbf{G}), M]$, if $A \in \mathbf{G}$, we have $\mathsf{MD}[\mathsf{ME}[A, \mathcal{P}(\mathbf{G}), M], \mathbf{G}, \mathbf{S}] = M$; Otherwise, $A \notin \mathbf{G}$, we have $\mathsf{MD}[\mathsf{ME}[A, \mathcal{P}(\mathbf{G}), M], \mathbf{G}, \mathbf{S}] = \perp$ .

## 2.2 Security Models of Membership Encryption

**Definition 1 (Message Security).** *A membership encryption captures the message security if given a ciphertext generated with $A$ and $\mathcal{P}(\mathbf{G})$, it is computationally hard to know the encrypted message when*

- *The decryptor does not have the secret key $\mathbf{S}$ of $\mathcal{P}(\mathbf{G})$, or*
- *The attribute $A$ does not satisfy the membership, i.e., $A \notin \mathbf{G}$.*

We define two games to capture message security. The first game is about *indistinguishability against secret key* and says that if the corresponding secret key $\mathbf{S}$ is unknown, it is indistinguishable to decide the message in a ciphertext for the corresponding token $\mathcal{P}(\mathbf{G})$ and any attribute $A$. The second game is about *indistinguishability against membership* and says that it is indistinguishable to decide the message in a ciphertext for any attribute $A$ and any group token $\mathcal{P}(\mathbf{G})$ if $A \notin \mathbf{G}$ holds.

**Game 1: Indistinguishability against Secret Key**

- **Setup:** The challenger runs the $\mathsf{Setup}$ algorithm to generate the system parameter $SP$, and sends it to the adversary.
- **Phase 1:** The adversary queries group tokens and decryption as follows.
  - For a token query on group attribute $\mathbf{G}_i$ that is adaptively chosen by the adversary, the challenger responds by generating $(\mathcal{P}(\mathbf{G}_i), \mathbf{S}_i)$ and sending $\mathcal{P}(\mathbf{G}_i)$ to the adversary.
  - For a decryption query on a ciphertext $C_i$ for $(A, \mathcal{P}(\mathbf{G}_i))$ where $\mathcal{P}(\mathbf{G}_i)$ is generated by the challenger, if $A \notin \mathbf{G}$, the challenger returns $\perp$ to the adversary; otherwise, the challenger responds by decrypting the ciphertext with $\mathbf{S}_i$, and sending the decryption result to the adversary.

- **Challenge:** The adversary gives the challenger one attribute $A^*$, one group token $\mathcal{P}(\mathbf{G}^*)$ and two messages $M_0, M_1$, where $\mathcal{P}(\mathbf{G}^*)$ was generated in the query phase. The challenger responds by randomly choosing a coin $c \in \{0, 1\}$, generating a ciphertext $C^* \leftarrow \mathsf{ME}[A^*, \mathcal{P}(\mathbf{G}^*), M_c]$, and sending the challenge ciphertext to the adversary.
- **Phase 2:** The adversary can continue the query the same as Phase 1 except no decryption query on the challenge ciphertext $C^*$ for $(A^*, \mathcal{P}(\mathbf{G}^*))$.
- **Win:** The adversary outputs a guess $c'$ of $c$ and wins the game if $c' = c$.

We define the advantage of adversary as $\mathsf{Adv}_{I_1} = \big|\Pr[c' = c] - 1/2\big|$.

**Definition 2.** *A membership encryption generated with a security parameter* $1^\lambda$ *is* $(t, q_k, q_d, \epsilon)$*-secure against secret key if for all t-polynomial time adversaries who make* $q_k$ *token key queries at most and* $q_d$ *decryption queries at most, we have* $\epsilon = \mathsf{Adv}_{I_1}$ *is a negligible function of* $\lambda$.

**Game 2: Indistinguishability against Membership**

- **Setup:** The challenger runs the $\mathsf{Setup}$ algorithm to generate the system parameter $SP$, and sends it to the adversary.
- **Challenge:** The adversary gives the challenger one attribute $A^*$, $\mathcal{P}(\mathbf{G}^*)$, $\mathbf{G}^*$, $\mathbf{S}$ and two messages $M_0, M_1$. The challenger first verifies that $A \notin \mathcal{P}(\mathbf{G}^*)$ with $\mathbf{G}^*$ and $\mathbf{S}$. Then, the challenger responds by randomly choosing a coin $c \in \{0, 1\}$, generating a ciphertext $C^* \leftarrow \mathsf{ME}[A^*, \mathcal{P}(\mathbf{G}^*), M_c]$, and sending the challenge ciphertext to the adversary.
- **Win:** The adversary outputs a guess $c'$ of $c$ and wins the game if $c' = c$.

We define the advantage of adversary as $\mathsf{Adv}_{I_2} = \big|\Pr[c' = c] - 1/2\big|$.

**Definition 3.** *A membership encryption generated with a security parameter* $1^\lambda$ *is* $(t, \epsilon)$*-secure against membership if for all t-polynomial time adversaries, we have* $\epsilon = \mathsf{Adv}_{I_2}$ *is a negligible function of* $\lambda$*. We call the membership encryption selectively secure [4] against membership if the adversary must output* $A^*$ *and* $\mathbf{G}^*$ *before the setup of system parameters.*

**Definition 4 (Privacy).** *A membership encryption preserves the privacy of group attributes if given a group token* $\mathcal{P}(\mathbf{G})$ *and two group attributes* $\mathbf{G}_0 = \{A_1, A_2, \cdots, A_{k_1}\}$ *and* $\mathbf{G}_1 = \{A'_1, A'_2, \cdots, A'_{k_2}\}$*, it is computationally hard to decide whether* $\mathbf{G} = \mathbf{G}_0$ *or* $\mathbf{G} = \mathbf{G}_1$.

A secure membership encryption only guarantees the decryptor has the secret key $\mathbf{S}$ and $A$ belongs to $\mathbf{G}$. To protect the privacy of group tokens, the membership encryption must capture the privacy property defined above. The game playing of privacy is defined as follows.

**Game 3: Privacy**

- **Setup:** The challenger runs the $\mathsf{Setup}$ algorithm to generate the system parameter $SP$, and sends it to the adversary.

- **Challenge:** The adversary gives the challenger two group attributes $\mathbf{G}_0 = \{A_1, A_2, \cdots, A_{k_1}\}$ and $\mathbf{G}_1 = \{A'_1, A'_2, \cdots, A'_{k_2}\}$. The challenger responds by randomly choosing a coin $c \in \{0, 1\}$ and generating $\mathcal{P}(\mathbf{G}_c)$ for $\mathbf{G}_c$. Then, the challenger sends $\mathcal{P}(\mathbf{G}_c)$ to the adversary.
- **Win:** The adversary outputs a guess $c'$ of $c$ and wins the game if $c' = c$.

We define the advantage of adversary as $\mathsf{Adv}_P = \big|\Pr[c' = c] - 1/2\big|$.

**Definition 5.** *A membership encryption generated with a security parameter $1^\lambda$ preserves the privacy of group tokens with $(t, \epsilon)$ if for all $t$-polynomial time adversaries, we have $\epsilon = \mathsf{Adv}_P$ is a negligible function of $\lambda$. We say it unconditionally preserves the privacy of group tokens if $\epsilon = 0$ for any time $t$ and $SP$ is generated by the adversary.*

The properties of message security and privacy are sufficient for the definition of membership encryption. We define the additional maximum number accountability so as to apply it to constructing a flexible $(K, N)$-oblivious transfer protocol for any $K \leq N$.

**Definition 6 (Maximum Number Accountability).** *A membership encryption captures the property of maximum number accountability, if it is computationally hard to generate a group token pair $(\mathcal{P}(\mathbf{G}), \mathbf{S})$ for $\mathbf{G}$ with $k$ attributes, but the verification shows that $|\mathcal{P}(\mathbf{G})| < k$.*

**Game 4: Maximum Number Accountability**

- **Challenge:** The challenger runs the $\mathsf{Setup}$ algorithm to generate the system parameter $SP$, and sends it to the adversary.
- **Win:** The adversary outputs $(\mathcal{P}(\mathbf{G}^*), \mathbf{G}^*, \mathbf{S})$ and wins the game if $\mathbf{G}^*$ contains $k$ numbers of attributes but the verification on $\mathcal{P}(\mathbf{G}^*)$ shows that it contains less than $k$ attributes.

We define the advantage of adversary as $\mathsf{Adv}_A$.

**Definition 7.** *A membership encryption generated with a security parameter $1^\lambda$ is $(t, \epsilon)$-secure with maximum number accountability if for all $t$-polynomial time adversaries, we have $\epsilon = \mathsf{Adv}_A$ is a negligible function of $\lambda$.*

## 3 Our Membership Encryption

### 3.1 Pairing Group

Our membership encryption can be built from any pairing group. Let $\mathcal{G}_B$ be a generator of pairing groups. Taking as input a security parameter $1^\lambda$, it outputs a pairing group $\mathbb{PG} = (\mathbb{G}, \mathbb{G}_T, e, p, g')$, where $\mathbb{G}, \mathbb{G}_T$ are two cyclic groups of prime order $p$, $g'$ is a generator of $\mathbb{G}$, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is the bilinear map. The bilinear map $e$ is a map with the following three properties:

- For all $u, v \in \mathbb{G}, a, b \in \mathbb{Z}_p$, $e(u^a, v^b) = e(u, v)^{ab}$.
- $e(g', g')$ is a generator of $\mathbb{G}_T$.
- It is efficient to compute the bilinear map $e(u, v)$ for any $u, v \in \mathbb{G}$.

## 3.2 The Scheme

Our group token generation is extended from the accumulator scheme in [17] with two secret keys $\alpha$ and $\beta$. Let $u \in \mathbb{G}$ and $u, u^{\alpha}, u^{\alpha^2}, \cdots, u^{\alpha^n}, u^{\beta\alpha}, u^{\beta\alpha^2}, \cdots, u^{\beta\alpha^n}$ be the components of system parameter. The group token $\mathcal{P}(\mathbf{G})$ for $\mathbf{G} = \{A_1, A_2, \cdots, A_k\} \in \mathbb{Z}_p$ is defined as

$$\mathcal{P}(\mathbf{G}) = (w_1, w_2, w_3) = \left(u^{\tau \prod_{i=1}^{k}(\alpha+A_i)},\ u^{\tau\beta \prod_{i=1}^{k}(\alpha+A_i)},\ u^{\tau\beta\alpha^{n-k} \prod_{i=1}^{k}(\alpha+A_i)}\right)$$

where $\mathbf{S} = \tau \in \mathbb{Z}_p$ is randomly chosen, and $w_3$ is the element for attribute number verification. Suppose $u^{\frac{1}{(\alpha+A)(\beta+A)}}$ is also in the system parameters and $r$ is a randomness from $\mathbb{Z}_p$. Our approach for membership is described as follows:

– If $A \in \mathbf{G}$, we have $w_2 w_1^A$ contains $(\alpha + A)(\beta + A)$ such that

$$e\Big(\big(w_2 w_1^A\big)^r, u^{\frac{1}{(\alpha+A)(\beta+A)}}\Big) = e(u,u)^{r \prod_{A_i \in \mathbf{G}/A}(\alpha+A_i)}$$

is computable from $u^r$ and the system parameter.

– Otherwise, $A \notin \mathbf{G}$, we have $w_2 w_1^A = u^{\tau(\beta+A)\prod_{i=1}^{k}(\alpha+A_i)}$ such that

$$e\big((w_2 w_1^A)^r, u^{\frac{1}{(\alpha+A)(\beta+A)}}\big) = e(u,u)^{r \cdot \frac{\tau \prod_{A_i \in \mathbf{G}}(\alpha+A_i)}{\alpha+A}}$$

contains the inversion exponent $\frac{1}{\alpha+A}$, which cannot be computed from $u^r$ and the system parameter.

We use the above two different results to encrypt messages so that the decryption requires $A \in \mathbf{G}$. The detailed construction modified from [15] with security against chosen-plaintext attack ($q_d = 0$ in Game 1) is described as follows.

**Setup:** Taking as input a security parameter $1^\lambda$, let $n$ be the upper bound attribute number in group token generation and let all attributes be $\mathbb{A} = \{A_1, A_2, \cdots, A_n\} \subseteq \mathbb{Z}_p$, the setup algorithm works as follows:

– Choose a pairing group $\mathbb{PG} = (\mathbb{G}, \mathbb{G}_T, e, p, g')$.
– Choose $\alpha, \beta, \gamma \in \mathbb{Z}_p$ and $g, h \in \mathbb{G}$ at random. Compute $e(g^\gamma, h)$ and $g^{\gamma\alpha}$.
– Compute $u_i = h^{\gamma\alpha^i}$ and $v_i = h^{\gamma\beta\alpha^i}$ for all $i = 0, 1, \cdots, n$.
– Randomly choose $s_i$ from $\mathbb{Z}_p$ and compute $d_{A_i}$ for $i = 1, 2, \cdots, n$ as

$$d_{A_i} = \big(g^{\frac{s_i}{(\alpha+A_i)(\beta+A_i)}},\ \ h^{\frac{s_i-1}{\alpha}},\ \ h^{s_i},\ \ h^{s_i\alpha},\ \cdots,\ \ h^{s_i\alpha^{n-2}}\big).$$

The system parameter $SP$ is defined as

$$SP = \Big(\mathbb{PG}, u_0, u_1, u_2, \cdots, u_n, v_0, v_1, v_2, \cdots, v_n, e(g^\gamma, h), g^{\gamma\alpha}, d_{A_1}, d_{A_2}, \cdots, d_{A_n}\Big).$$

**GroupGen:** Taking as input the group attribute $\mathbf{G} = \{A_1, A_2, \cdots, A_k\} \in \mathbb{Z}_p$ for any $k \leq n$, let $F(x) = \prod_{i=1}^{k}\big(x + A_i\big)$ and $F_i$ be the coefficient of $x^i$, the group token generation algorithm randomly chooses $\tau$ from $\mathbb{Z}_p$, sets $\mathbf{S} = \tau$ and computes $\mathcal{P}(\mathbf{G})$ as

$$\begin{aligned}\mathcal{P}(\mathbf{G}) = (w_1, w_2, w_3) &= \Big(h^{\tau\gamma F(\alpha)},\ \ h^{\tau\gamma\beta F(\alpha)},\ \ h^{\tau\gamma\beta\alpha^{n-k}F(\alpha)}\Big)\\ &= \Big(\prod_{i=0}^{k} u_i^{F_i\tau},\ \ \prod_{i=0}^{k} v_i^{F_i\tau},\ \ \prod_{i=0}^{k} v_{n-k+i}^{F_i\tau}\Big).\end{aligned}$$

**Verify:** Taking as input $\mathcal{P}(\mathbf{G})$ and $k$, accept $|\mathcal{P}(\mathbf{G})| \leq k$ if $e(w_2, u_n) = e(w_3, u_k)$.

**Encrypt:** Taking as input an attribute $A \in \mathbb{A}$, a group token $\mathcal{P}(\mathbf{G}) = (w_1, w_2, w_3)$, a message $M \in \mathbb{G}_T$ and the system parameter, the encryption algorithm works as follows:

- Verify that $w_2 = w_1^{\beta}$ by checking $e(w_1, v_1) = e(w_2, v_0)$.
- Randomly choose $r$ from $\mathbb{Z}_p$. Compute the ciphertext on the message $M$ as

$$C = (c_1, c_2, c_3) = \Big(\big(w_2 w_1^{A}\big)^r,\ \ (g^{\gamma\alpha})^r,\ \ e(g^{\gamma}, h)^r \cdot M\Big).$$

**Decrypt:** Taking as input the ciphertext $C$, the secret key $\mathbf{S}$, the attribute $A$, the group attribute $\mathbf{G}$ and the system parameter, the decryption algorithm is described as follows.

- Compute

$$c_1' = c_1^{\frac{1}{\mathbf{S}}} = \Big(h^{r\tau\gamma(\beta+A)F(\alpha)}\Big)^{\frac{1}{\tau}} = h^{r\gamma(\beta+A)F(\alpha)}.$$

- Compute the pairing

$$e_1 = e\Big(c_1',\ g^{\frac{s}{(\alpha+A)(\beta+A)}}\Big) = e(g,h)^{rs\gamma\frac{F(\alpha)}{\alpha+A}}.$$

- If $A \in \mathbf{G}$, we have $A$ is a root of $F(x)$. Let $F_i'$ be the coefficient of $x^i$ in $\frac{F(x)}{x+A}$. Compute the pairing

$$e_2 = e\Big((h^{\frac{s-1}{\alpha}})^{F_0'} \cdot \prod_{i=1}^{k-1}(h^{s\alpha^{i-1}})^{F_i'}, c_2\Big) = e(g,h)^{rs\gamma\frac{F(\alpha)}{\alpha+A} - F_0' r\gamma}.$$

- Compute $M$ by

$$c_3 \cdot \big(e_2 e_1^{-1}\big)^{\frac{1}{F_0'}} = e(g^{\gamma}, h)^r M \cdot \big(e(g,h)^{-F_0' r\gamma}\big)^{\frac{1}{F_0'}} = M.$$

## 3.3 Discussions

The above membership encryption is proposed for security against chosen-plaintext attack (CPA), i.e., the adversary cannot make decryption queries. Let $\mathcal{ME}[M, r]$ be our ciphertext on $M$ encrypted with the randomness $r$. Using the Fujisaki-Okamoto approach [13] in the random oracle model, we can easily extend it to the security against chosen-ciphertext attack (CCA). Let $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p$ and $H_2 : \{0,1\}^* \rightarrow \{0,1\}^{l_m}$ be cryptographic hash functions, where $l_m$ denotes the length of messages. If $\mathcal{ME}[M, r]$ is secure against CPA, the following membership encryption construction for $(A, \mathcal{P}(\mathbf{G}))$ is secure against CCA

$$\mathcal{ME}\big[\sigma,\ H_1(A, \mathcal{P}(\mathbf{G}), \sigma, M)\big],\quad H_2(\sigma) \oplus M.$$

It is not hard to prove CCA security under our security model definition in Game 1 with the proof in [13]. We omit it here.

Our membership encryption captures the following nice features.

– Constant-size $\mathcal{P}(\mathbf{G})$. Our group token $\mathcal{P}(\mathbf{G})$ consists of three group elements only independent of the number of attributes in $\mathbf{G}$.
– Maximum number accountability. According to our setting, we have

$$\mathcal{P}(\mathbf{G}) = (w_1, w_2, w_3) = (w_1, w_1^{\beta}, w_1^{\beta\alpha^{n-k}})$$

for $k$ numbers of attributes. Through the verification, the verifier knows that the exponent of $w_3$ contains $\alpha^{n-k}$. We have $F(\alpha)$ in $w_1 = h^{\tau\gamma F(\alpha)}$ has $k$ degrees at most; otherwise, computing $w_3$ needs $h^{\gamma\beta}, h^{\gamma\beta\alpha^1}, \cdots, h^{\gamma\beta\alpha^{n'}}$ for $n' > n$ and they are not given in the system parameter.
– Constant-size ciphertext. Our ciphertext is constant-size and is composed of two group elements from $\mathbb{G}$ and one group element from $\mathbb{G}_T$. The length of ciphertext depends on the length of security parameter only.

## 4 Proof of Security

In this section, we prove the security of our membership encryption. Before the security analysis, we introduce three hard problems adopted in our reduction proof.

### 4.1 Hard Problems

Our membership encryption uses a pairing group as an ingredient and its security relies on the hardness of three problems that are slightly modified from the GDDHE problem [12], the a-MSE-DDH problem [15] and the DHE problem [7]. The new hard problems are $(f, n)$-GDDHE problem that is adopted to prove message security against secret key (Game 1), $(f, g, n)$-aMSE-DDHE problem which is used to prove message security against membership (Game 2), and $(f, n)$-DHE problem that is used to prove the property of maximum number accountability (Game 4). We notice that the intractability of these three hard problems can be analysed in the generic group model by following the proof in [5,12] for the original GDDHE problem. For completeness, we analyse these problems based on the Theorem 2 in [12] in the full version of this paper.

Let $g_0, h_0, w$ be random generators from $\mathbb{G}$ and $a, \gamma$ be random integers from $\mathbb{Z}_p$. The three hard problems are defined as follows.

$(f, n)$-GDDHE Problem:

Instance: Any $(2n+1)$-degree polynomial function $f(x) \in \mathbb{Z}_p[x]$.

$$\begin{array}{llllllll} g_0, & g_0^{a}, & g_0^{a^2}, & \cdots, & g_0^{a^{2n}}, & g_0^{af(a)} & g_0^{af(a)\theta}. \\ h_0, & h_0^{a}, & h_0^{a^2}, & \cdots, & h_0^{a^{2n}} & & \\ w, & w^{a}, & w^{a^2}, & \cdots, & w^{a^{2n}}, & w^{\theta} & w^{a\theta} \end{array}$$

$T \in \mathbb{G}_T$, which is either at random or equal to $e(g_0, h_0)^{f(a)\theta}$

Target: Return $b = 1$ if $T = e(g_0, h_0)^{f(a)\theta}$; otherwise, $b = 0$.

**Definition 8.** *The $(f, n)$-GDDHE problem holds with $(t, \epsilon)$ if given an instance generated from a security parameter $1^\lambda$ and any $(2n+1)$-degree polynomial function $f(x) \in \mathbb{Z}_p[x]$, the advantage of solving this problem in $t$ polynomial time is $\epsilon$ at most which is a negligible function of $\lambda$.*

$(f, g, n)$-aMSE-DDH Problem:

Instance: Any $(2n+1)$-degree polynomial function $f(x) \in \mathbb{Z}_p[x]$, and any degree $\leq 2n$ polynomial function $g(x) \in \mathbb{Z}_p[x]$, such that $gcd(f(x), g(x)) = 1$ (or any nonzero number).

$$\begin{array}{llllll} g_0, & g_0^{a}, & g_0^{a^2}, & \cdots, & g_0^{a^{2n}}, & g_0^{\theta a f(a)} \\ g_0^{\gamma}, & g_0^{\gamma a}, & g_0^{\gamma a^2}, & \cdots, & g_0^{\gamma a^{2n+2}} & \\ h_0, & h_0^{a}, & h_0^{a^2}, & \cdots, & h_0^{a^{2n}}, & h_0^{\theta g(a)} \\ h_0^{\gamma}, & h_0^{\gamma a}, & h_0^{\gamma a^2}, & \cdots, & h_0^{\gamma a^{2n}} & \end{array}$$

$T \in \mathbb{G}_T$, which is either at random or equal to $e(g_0, h_0)^{f(a)\theta}$

Target: Return $b = 1$ if $T = e(g_0, h_0)^{f(a)\theta}$; otherwise, $b = 0$.

**Definition 9.** *The $(f, g, n)$-aMSE-DDH problem holds with $(t, \epsilon)$ if given an instance generated from a security parameter $1^\lambda$ and any two co-prime polynomial functions $f(x), g(x)$ in $\mathbb{Z}_p[x]$ with $(2n+1)$ degrees and $\leq 2n$ degrees respectively, the advantage of solving this problem in $t$ polynomial time is $\epsilon$ at most which is a negligible function of $\lambda$.*

$(f, n)$-DHE Problem:

Instance: $g_0, \ g_0^{a}, \ g_0^{a^2}, \ \cdots, \ g_0^{a^n}$.

Output: Return $(f(x), g_0^{f(a)})$, where $f(x) \in \mathbb{Z}_p[x]$ is an $n'$-degree polynomial function $n' > n$.

**Definition 10.** *The $(f, n)$-DHE problem holds with $(t, \epsilon)$ if given an instance generated from a security parameter $1^\lambda$, the advantage of solving this problem in $t$ polynomial time is $\epsilon$ at most which is a negligible function of $\lambda$.*

## 4.2 Security Proof

**Theorem 1 (Indistinguishability against Secret Key).** *Suppose the $(f, n)$-GDDHE problem is $(t, \epsilon)$-hard, we can construct $(t', q_k, \epsilon')$-secure membership encryption against secret key. Here, $t' = t - O(q_k n t_e)$ and $\epsilon' = \frac{\epsilon}{q_k}$, where $t_e$ denotes the average time of an exponentiation in $\mathbb{G}$.*

The proof is given in the full version of this paper.

**Theorem 2 (Indistinguishability against Membership).** *Suppose the $(f, g, n)$-aMSE-DDH problem is $(t, \epsilon)$-hard, we can construct $(t', q_k, \epsilon')$ selectively secure membership encryption against membership. Here, $t' = t - O(n^2 t_e)$ and $\epsilon' = \epsilon$, where $t_e$ denotes the average time of an exponentiation in $\mathbb{G}$.*

We prove the security of membership in the selective security model in which the adversary must output $A^*$ and $\mathbf{G}^*$ before the setup of system parameter. The security proof can be transformed into full security by correctly guessing the challenge target, but it is only suitable for small $n$ and $\mathbf{G}^*$.

*Proof.* Suppose there exists an adversary who can break the membership encryption against membership under selective security model. We construct an algorithm $\mathcal{B}$ that solves the $(f, g, n)$-aMSE-DDH problem. $\mathcal{B}$ interacts with the adversary as the follows.

**Initialization.** Let $\mathbb{PG} = (\mathbb{G}, \mathbb{G}_T, e, p, g')$ be the pairing group and $\mathbb{A} = \{A_1, A_2, \cdots, A_n\}$ be all attributes. The adversary outputs $(A^*, \mathbf{G}^*)$ for challenge where $A^* \notin \mathbf{G}^*$.

**Setup.** The algorithm $\mathcal{B}$ works as follows to simulate the system parameter.

– Let $\mathbf{G}^* = \{A_1^*, A_2^*, \cdots, A_k^*\}$ be the attributes in $\mathbf{G}^*$. Define the set $\mathbf{G}_1$ as follows
$$\mathbf{G}_1 = \{A_1, A_2, \cdots, A_n\}/\{A_1^*, A_2^*, \cdots, A_k^*, A^*\}.$$
– Randomly choose $\beta_0, \beta_1$ from $\mathbb{Z}_p$. Let $f(x)$ be a $(2n+1)$-degree polynomial function and $g(x)$ be a $(k+1)$-degree polynomial function defined as
$$(x + A^*) \prod_{A_i \in \mathbf{G}_1} (x + A_i) \cdot \prod_{A_i \in \mathbb{A}} (\beta_0 x + \beta_1 + A_i) \Big| f(x)$$
$$g(x) = (\beta_0 x + \beta_1 + A^*) \prod_{A_i \in \mathbf{G}^*} (x + A_i),$$
such that $gcd\big(f(x), g(x)\big) = 1$ (or any nonzero number).
– Send $f(x), g(x)$ to the $(f, g, n)$-aMSE-DDH problem generator. Let be challenge instance be
$$\begin{array}{lllll} g_0, & g_0^a, & g_0^{a^2}, \cdots, & g_0^{a^{2n}}, & g_0^{\theta a f(a)} \\ g_0^{\gamma}, & g_0^{\gamma a}, & g_0^{\gamma a^2}, \cdots, & g_0^{\gamma a^{2n+2}} & \\ h_0, & h_0^a, & h_0^{a^2}, \cdots, & h_0^{a^{2n}}, & h_0^{\theta g(a)} \\ h_0^{\gamma}, & h_0^{\gamma a}, & h_0^{\gamma a^2}, \cdots, & h_0^{\gamma a^{2n}} & \\ T \in \mathbb{G}_T & & & & \end{array}$$
– Set $\alpha, \beta, \gamma, g, h$ as
$$\alpha = a, \quad \beta = \beta_0 a + \beta_1, \quad \gamma = \gamma, \quad g = g_0^{f(a)}, \quad h = h_0,$$
where $a, \gamma$ are the randomness in the challenge instance.

– Compute $e(g^\gamma, h), g^{\gamma\alpha}, u_i, v_i$, as

$$e(g^\gamma, h) = e(g_0, h_0)^{\gamma f(a)}, \quad g^{\gamma\alpha} = g_0^{af(a)\gamma}, \quad u_i = h^{\gamma\alpha^i} = h_0^{\gamma a^i}$$

$$v_i = h^{\gamma\beta\alpha^i} = h_0^{\beta_0\gamma a^{i+1}+\beta_1\gamma a^i}.$$

– Compute $d_{A_i}$ as follows.

- Randomly choose $s_i'$ from $\mathbb{Z}_p$ and set

$$s_i = (s_i'\gamma a + 1)f_{A_i}(a),$$

where $f_{A_i}(x)$ is defined as follows

$$f_{A_i}(x) = \begin{cases} \frac{1}{\beta_1+A^*}(\beta_0 x + \beta_1 + A^*) & \text{if } A_i = A^*, \\ \frac{1}{A_i}(x + A_i) & \text{else if } A_i \in \mathbf{G}^*, \\ 1 & \text{otherwise } A_i \in \mathbf{G}_1. \end{cases}$$

We have

$$\frac{f_{A_i}(a) - 1}{a} = \begin{cases} \frac{\beta_0}{\beta_1+A^*} & \text{if } A_i = A^*, \\ \frac{1}{A_i} & \text{else if } A_i \in \mathbf{G}^*, \\ 0 & \text{otherwise } A_i \in \mathbf{G}_1. \end{cases}$$

such that

$$\begin{aligned}\frac{s_i - 1}{\alpha} = \frac{(s_i'\gamma a + 1)f_{A_i}(a) - 1}{a} &= f'_{A_i}(a) \\ = \gamma s_i' f_{A_i}(a) + \frac{f_{A_i}(a) - 1}{a} &= a_2\gamma a + a_1\gamma + a_0,\end{aligned}$$

where $a_2, a_1, a_0$ are coefficients. Let $f''_{A_i}(x)$ be defined as

$$f''_{A_i}(x) = \frac{f(x)f_{A_i}(x)}{(x + A_i)(\beta_0 x + \beta_1 + A_i)}.$$

We have $f''_{A_i}(x)$ is a polynomial function with $2n$ degrees at most.

- Compute $d_{A_i}$ as

$$\left(g_0^{s_i'\gamma a f''_{A_i}(a)+f''_{A_i}(a)},\ h_0^{f'_{A_i}(a)},\ h_0^{(s_i'\gamma a+1)f_{A_i}(a)}, \cdots, h_0^{(s_i'\gamma a+1)f_{A_i}(a)a^{n-2}}\right).$$

According to the setting of the randomness $s_i = (s_i'\gamma a + 1)f_{A_i}(a)$, we have

$$\begin{aligned} d_{A_i} &= \left(g_0^{s_i'\gamma a f''_{A_i}(a)+f''_{A_i}(a)},\ h_0^{f'_{A_i}(a)},\ h_0^{(s_i'\gamma a+1)f_{A_i}(a)}, \cdots, h_0^{(s_i'\gamma a+1)f_{A_i}(a)a^{n-2}}\right) \\ &= \left(g^{\frac{s_i}{(\alpha+A_i)(\beta+A_i)}},\ h^{\frac{s_i-1}{\alpha}},\ h^{s_i},\ \cdots,\ h^{s_i\alpha^{n-2}}\right).\end{aligned}$$

All elements are computable from the challenge instance and setting. $\mathcal{B}$ generates the system parameter and sends it to the adversary.

**Challenge.** The adversary returns $(A^*, \mathcal{P}(\mathbf{G})^*, \mathbf{G}^*, \mathbf{S}^*, M_0, M_1)$ for challenge. Let $\mathbf{S} = \tau^*$, $\mathcal{P}(\mathbf{G}^*) = (w_1, w_2, w_3)$ and $\mathbf{G}^* = \{A_1^*, A_2^*, \cdots, A_k^*\}$. The algorithm $\mathcal{B}$ randomly chooses a coin $c \in \{0,1\}$, and simulates the challenge ciphertext as follows

$$C = (c_1^*, c_2^*, c_3^*) = \Big( \big(h_0^{\theta g(a)}\big)^{\tau^*},\ \ g_0^{\theta a f(a)},\ \ T \cdot M_c \Big).$$

Let $r = \frac{\theta}{\gamma}$. If $T = e(g_0, h_0)^{\theta f(a)}$, we have

$$\begin{aligned}
(w_2 w_1^{A^*})^r &= (h^{\tau^* \gamma \prod_{i=1}^{k}(\alpha + A_i^*)(\beta + A^*)})^r = (h_0^{\tau^* \gamma g(a)})^{\frac{\theta}{\gamma}} = \big(h_0^{\theta g(a)}\big)^{\tau^*} \\
(g^{\gamma\alpha})^r &= (g_0^{\gamma a f(a)})^{\frac{\theta}{\gamma}} = g_0^{\theta a f(a)} \\
e(g^\gamma, h)^r &= e(g_0^{\gamma f(a)}, h_0)^{\frac{\theta}{\gamma}} = e(g_0, h_0)^{\theta f(a)} = T.
\end{aligned}$$

Therefore, $C = (c_1^*, c_2^*, c_3^*)$ is a valid ciphertext on $M_c$ for $(A^*, \mathcal{P}(\mathbf{G}^*))$. $\mathcal{B}$ sends it to the adversary.

**Win:** The adversary outputs $c' \in \{0,1\}$, and the algorithm $\mathcal{B}$ outputs $c'$ as the guess of $T$.

This completes the description of our simulation. If $T = e(g_0, h_0)^{\theta f(a)}$, the challenge ciphertext is valid and the adversary will output $c' = c$ with advantage $1/2 + \epsilon$; otherwise, $T$ is universally random and the adversary's advantage is $1/2$. The simulation time is mainly dominated by the $d_{\mathbb{A}}$ simulation, and each $d_{A_i}$ costs $O(n)$ exponentiations. No abortion occurs during our simulation. We therefore obtain the Theorem 2. □

**Theorem 3 (Privacy).** *$\mathcal{P}(\mathbf{G})$ unconditionally preserves the privacy of all attributes in $\mathbf{G}$.*

*Proof.* Let $\mathcal{P}(\mathbf{G})$ be a group token generated from $\mathbf{G} = \{A_1, A_2, \cdots, A_{k_1}\}$ and $\mathbf{S} = \tau$. We have

$$\mathcal{P}(\mathbf{G}) = (w_1, w_2, w_3) = \Big( h^{\tau\gamma \prod_{i=1}^{k_1}(\alpha + A_i)},\ \ h^{\tau\gamma\beta \prod_{i=1}^{k_1}(\alpha + A_i)},\ \ h^{\tau\beta\alpha^{n-k_1} \prod_{i=1}^{k_1}(\alpha + A_i)} \Big).$$

Since there exists $\mathbf{G}' = \{A_1', A_2', \cdots, A_{k_2}'\}$ and $\tau' \in \mathbb{Z}_p$ satisfying

$$\tau \prod_{i=1}^{k_1}(\alpha + A_i) = \tau' \prod_{i=1}^{k_2}(\alpha + A_i'),$$

we have $\mathcal{P}(\mathbf{G})$ can be also seen as a group token generated for $\mathbf{G}' = \{A_1', A_2', \cdots, A_{k_2}'\}$ and $\tau'$. Thus, the privacy of all attributes in $\mathcal{P}(\mathbf{G})$ is unconditionally preserved. This completes the proof and we obtain the Theorem 3. □

**Theorem 4 (Maximum Number Accountability).** *Suppose the $(f, n)$-DHE problem is hard, the group token $\mathcal{P}(\mathbf{G})$ is secure with maximum number accountability.*

The proof is given in the full version of this paper.

## 5 Oblivious Transfer from Membership Encryption

In this section, we show how to construct an efficient K-out-of-N oblivious transfer protocol ($\mathsf{OT}_\mathsf{N}^\mathsf{K}$) from membership encryption. Our $\mathsf{OT}_\mathsf{N}^\mathsf{K}$ protocol only requires two rounds between receiver and sender. Using our construction, the $\mathsf{OT}_\mathsf{N}^\mathsf{K}$ protocol exhibits the nice property of constant communication cost, where the receiver sends constant-size messages to the sender independent of $K$ and $N$.

Suppose the sender has messages $M_1, M_2, M_3, \cdots, M_N$ for any $N \leq n$, and the receiver wants to receive messages $M_{i_1}, M_{i_2}, \cdots, M_{i_K}$ for any $\{i_1, i_2, \cdots, i_K\} \subseteq \{1, 2, \cdots, N\}$. Let $SP$ be the system parameter of membership encryption, where all attributes are the indices, i.e. $\mathbb{A} = \{1, 2, \cdots, n\}$. Our OT protocol from membership encryption depicted in **Fig. 1** is described as follows.

$$
\begin{array}{lcl}
\textbf{Receiver} & & \textbf{Sender} \\
\text{Choices: } i_1, i_2, \cdots, i_K & & \text{Messages: } M_1, M_2, \cdots, M_N \\
 & & \\
\mathbf{G} = \{i_1, i_2, \cdots, i_K\} & & \\
\mathbf{S} = \tau & & \\
\mathcal{P}(\mathbf{G}) \leftarrow \mathsf{GroupGen}[\mathbf{S}, \mathbf{G}, SP] & & \\
 & \xrightarrow{\quad\mathcal{P}(\mathbf{G})\quad} & \\
 & & \text{Verify that } |\mathcal{P}(\mathbf{G})| \leq K \\
 & & C_1 \;= \mathsf{ME}[1,\ \mathcal{P}(\mathbf{G}), M_1] \\
 & & C_2 \;= \mathsf{ME}[2,\ \mathcal{P}(\mathbf{G}), M_2] \\
 & & \quad \cdots \\
 & & C_N = \mathsf{ME}[N, \mathcal{P}(\mathbf{G}), M_N] \\
 & & C = (C_1, C_2, \cdots, C_N). \\
 & \xleftarrow{\qquad C \qquad} & \\
M_{i_1} = \mathsf{MD}[C_{i_1},\ \mathbf{G}, \mathbf{S}] & & \\
M_{i_2} = \mathsf{MD}[C_{i_2},\ \mathbf{G}, \mathbf{S}] & & \\
\quad \cdots & & \\
M_{i_K} = \mathsf{MD}[C_{i_K}, \mathbf{G}, \mathbf{S}] & &
\end{array}
$$

**Fig. 1.** $K$-Out-of-$N$ Oblivious Transfer

- The receiver runs the $\mathsf{GroupGen}$ algorithm to generate $\mathcal{P}(\mathbf{G})$ on $\mathbf{G} = \{i_1, i_2, \cdots, i_K\}$, and sends $\mathcal{P}(\mathbf{G})$ to the sender.
- Upon receiving $\mathcal{P}(\mathbf{G})$ from the receiver, the sender verifies that $|\mathcal{P}(\mathbf{G})| \leq K$. If it is false, reject. Otherwise, runs the encryption algorithm $C_i = \mathsf{ME}[i, \mathcal{P}(\mathbf{G}), M_i]$ for all $i = 1, 2, \cdots, N$ and sends all ciphertexts $C = (C_1, C_2, \cdots, C_N)$ to the receiver.
- Upon receiving all ciphertexts from the sender, the receiver runs the decryption algorithm $\mathsf{MD}[C_i, \mathbf{G}, \mathbf{S}]$ to get the message $M_i$ for all $i = i_1, i_2, \cdots, i_K$.

Our $\mathsf{OT}_\mathsf{N}^\mathsf{K}$ scheme preserves receiver's privacy and protects sender's messages against malicious receivers under the security model definition in [10]. According to the Theorem 3, given $\mathcal{P}(\mathbf{G})$, the sender cannot distinguish $\mathcal{P}(\mathbf{G})$ which is generated from either $\mathbf{G} = \{i_1, i_2, \cdots, i_K\}$ or $\mathbf{G} = \{i'_1, i'_2, \cdots, i'_K\}$. According to the Theorems 2 and 4, the receiver can only obtain chosen messages $M_j$ for all $j \in \{i_1, i_2, \cdots, i_K\}$.

The system parameter $SP$ in our OT protocol can be generated by the sender or the trust third party for the universal application. In our $\mathsf{OT}_\mathsf{N}^\mathsf{K}$ scheme, the receiver sends the group token $\mathcal{P}(\mathbf{G})$ aggregated for all $K$ choices and the sender responds with $N$ ciphertexts. The $\mathsf{OT}_\mathsf{N}^\mathsf{K}$ protocol is composed of two rounds only. Using our membership encryption, our group token is constant-size and is independent of the number of choice $K$, and our ciphertext is also constant-size and is dependent on the security parameter only. In **Table 1**, we compare the communication cost of all two-round $K$-out-of-$N$ oblivious transfer protocols in the literature. It shows that $\mathsf{OT}_\mathsf{N}^\mathsf{K}$ protocol from our membership encryption requires the smallest communication cost.

**Table 1.** Communication cost of two-round $K$-out-of-$N$ oblivious transfer

| | [16] | [18,10,9] | **Ours** |
|---|---|---|---|
| Messages from Receiver to Sender | $O(N)$ | $O(K)$ | $O(1)$ |
| Messages from Sender to Receiver | $O(N)$ | $O(N)$ | $O(N)$ |

## 6 Conclusion

Protecting membership privacy is essential in many applications. Existing solutions were based on membership proof for $\mathcal{P}(A) \in \mathbf{G}$ and $A \in \mathcal{P}(\mathbf{G})$. In this work, we extended the membership proof to membership encryption. We introduced the notion of membership encryption, where if the encryption takes as input an attribute $A$ and a privacy-preserving group token $\mathcal{P}(\mathbf{G})$, successful decryption must satisfy $A \in \mathbf{G}$. We constructed a provably secure membership encryption where the group token $\mathcal{P}(\mathbf{G})$ is constant-size and the maximum attribute number is accountable. The ciphertext is also constant and is dependent on security parameter only. We showed how to apply our encryption scheme to the construction of two-round $K$-out-of-$N$ oblivious transfer protocols $\mathsf{OT}_\mathsf{N}^\mathsf{K}$. Using our membership encryption, the $\mathsf{OT}_\mathsf{N}^\mathsf{K}$ protocol only requires $O(1)$ communication cost from receiver to sender, against the other existing two-round oblivious transfer protocols.

**Acknowledgement.** We would like to thank the anonymous reviewers for their helpful comments and suggestions.

## References

1. Au, M.H., Tsang, P.P., Susilo, W., Mu, Y.: Dynamic universal accumulators for DDH groups and their application to attribute-based anonymous credential systems. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 295–308. Springer, Heidelberg (2009)

2. Barić, N., Pfitzmann, B.: Collision-free accumulators and fail-stop signature schemes without trees. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 480–494. Springer, Heidelberg (1997)
3. Benaloh, J.C., de Mare, M.: One-way accumulators: A decentralized alternative to digital signatures (extended abstract). In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 274–285. Springer, Heidelberg (1994)
4. Boneh, D., Boyen, X.: Efficient selective-id secure identity-based encryption without random oracles. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)
5. 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)
6. Camenisch, J., Chaabouni, R., Shelat, A.: Efficient protocols for set membership and range proofs. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 234–252. Springer, Heidelberg (2008)
7. Camenisch, J., Kohlweiss, M., Soriente, C.: An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 481–500. Springer, Heidelberg (2009)
8. Camenisch, J., Lysyanskaya, A.: Dynamic accumulators and application to efficient revocation of anonymous credentials. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 61–76. Springer, Heidelberg (2002)
9. Chen, Y., Chou, J.S., Hou, X.W.: A novel k-out-of-n oblivious transfer protocols based on bilinear pairings. IACR Cryptology ePrint Archive 2010, 27 (2010)
10. Chu, C.-K., Tzeng, W.-G.: Efficient $k$-out-of-$n$ oblivious transfer schemes with adaptive and non-adaptive queries. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 172–183. Springer, Heidelberg (2005)
11. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs 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)
12. 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)
13. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 537–554. Springer, Heidelberg (1999)
14. Guo, F., Mu, Y., Chen, Z.: Mutative identity-based signatures or dynamic credentials without random oracles. In: Bao, F., Ling, S., Okamoto, T., Wang, H., Xing, C. (eds.) CANS 2007. LNCS, vol. 4856, pp. 1–14. Springer, Heidelberg (2007)
15. 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)
16. Mu, Y., Zhang, J., Varadharajan, V.: $m$ out of $n$ oblivious transfer. In: Batten, L.M., Seberry, J. (eds.) ACISP 2002. LNCS, vol. 2384, pp. 395–405. Springer, Heidelberg (2002)
17. Nguyen, L.: Accumulators from bilinear pairings and applications. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 275–292. Springer, Heidelberg (2005)
18. Ogata, W., Kurosawa, K.: Oblivious keyword search. J. Complexity 20(2-3), 356–371 (2004)
19. Rabin, M.O.: How to exchange secrets with oblivious transfer. IACR Cryptology ePrint Archive 2005, 187 (2005)

# Security Proofs for Hash Tree Time-Stamping Using Hash Functions with Small Output Size

Ahto Buldas[1,2,3] and Risto Laanoja[1,2,⋆]

[1] GuardTime AS, Tammsaare tee 60, 11316 Tallinn, Estonia
[2] Tallinn University of Technology, Raja 15, 12618 Tallinn, Estonia
[3] Cybernetica AS, Mealuse 2/1, 12618 Tallinn, Estonia

**Abstract.** The known security proofs for hash tree time-stamping assume collision-resistance (CR). An asymptotically optimally tight proof has the security loss formula $\frac{t'}{\delta'} \approx 14\sqrt{C}\left(\frac{t}{\delta}\right)^{1.5}$, where $\frac{t'}{\delta'}$ is the time-success ratio of a collision-finder, $\frac{t}{\delta}$ is the ratio of a back-dating adversary and $C$ is the size of the hash tree created in every time unit. Practical schemes use 256-bit hash functions that are just $2^{128}$-secure because of the birthday bound. For having a $2^{80}$-secure time-stamping scheme, we have $C < 10^3$ that is insufficient for global scale solutions. Due to tightness bounds for CR, practically relevant security proofs must use assumptions stronger than CR. We show that under the random oracle (RO) assumption, the security loss is independent of $C$. We establish a linear-preserving security reduction under the Pre-Image Awareness (PrA) assumption. We present a new slightly stronger assumption SPrA that leads to much tighter proofs. We also show that bounds on $C$ are necessary—based on any PrA/SPrA function, we construct a PrA/SPrA function that is insecure for unbounded time-stamping.

## 1 Introduction

Hash tree (keyless) time-stamping was first introduced by Haber and Stornetta [11] in order to eliminate secret-based cryptography and trusted third parties. In their scheme, a collection of $N$ documents is hashed down to a single digest of few dozen bytes that is then published in widely available media such as newspapers. Merkle hash trees [14] enable to create compact certificates (of size $\log N$) for each of the $N$ documents. Such certificates just consist of all sibling hash values in the path from a document (a leaf of the tree) to the root of the tree. The certificate is sufficient to re-compute the root hash value from the document and can be used as a *proof of membership*. Haber and Stornetta drafted a large-scale time-stamping scheme [1] where at every unit of time a large hash tree is created co-operatively by numerous servers all over the world and the root value is published in newspapers.

⋆ This research was supported by the European Regional Development Fund through the Estonian Center of Excellence in Computer Science, EXCS, and by Estonian Research Council's personal research grant PUT-2.

It might seem obvious that the security of a hash-then-publish time-stamping scheme can be reduced to collision-resistance of the hash function. However, the first correct security proof of such a scheme was published as late as in 2004 [6]. It turned out that the potential number $N$ of time-stamps explicitly affects the tightness of security proofs. The proof in [6] shows that if there is an adversary with running time $t$ that backdates a document with probability $\delta$, then there is a collision-findiner with running time $t' \approx 2t$ and success probability $\delta' \approx \frac{\delta^2}{N}$. When measuring security in terms of time-success ratio [12] we have to use $2N \cdot \frac{t}{\delta^2}$-collision resistant hash functions to have a $\frac{t}{\delta}$-secure time-stamping scheme, i.e. the hash function must be roughly $\frac{2N}{\delta}$ times more secure than the time-stamping system constructed from it. As $N$ could be very large in considering global-scale time-stamping, the security requirements for the hash function may grow unreasonably large. Indeed, it is said in [6] that such a security proof is practical only for hash functions with about 400-bit output.

The tightest known security proof [4] for hash tree time-stamping has the security loss formula $\frac{t'}{\delta'} \approx 14\sqrt{N}\frac{t}{\delta^{1.5}}$, which is insufficient because the security level it implies for systems that use 256-bit hash functions is not that one expects today. In order to have a $2^{80}$-secure system with the total capacity of $N$ timestamps, we need $n = \log_2 N + 248$ output bits. If $n = 256$, then $N < 2^8 = 256$, which is clearly too small. Moreover, the proof of [4] is asymptotically optimally tight, if the collision-resistance property is used as the security assumption. So, the only way out is to use stronger (or incomparable) security assumptions for hash functions. In this paper, we first show that if the hash function is assumed to be a random oracle, the security loss does not depend on $N$. Next, we establish a linear-preserving reduction that assumes the hash functions to be pre-image aware (PrA) constructions from ideal components. We also present a new slightly stronger than PrA security condition (SPrA) under which the security proof is much tighter than all previous ones and is very close to the tightness in the random oracle model. Finally, we show that the bounded capacity $N$ is a necessary assumption to prove the security of hash-tree time-stamping schemes under the PrA/SPrA assumption. Based on arbitrary PrA/SPrA hash function, we construct another PrA/SPrA hash function that is totally insecure for unbounded time-stamping. This negativity result is a generalization of a somewhat weaker oracle-separation based result [6] presented in Asiacrypt 2004 about the proofs that use collision-resistance.

## 2 Preliminaries

### 2.1 Security Proofs and Their Tightness

The security of a cryptographic protocol (or a primitive) is measured by the amount of resources (such as running time) needed for an adversary to break the primitive. A protocol is said to be $S$-secure, if it can be broken by no adversaries with less than $S$ units of resources available. Considering that the running time $t$ and the success probability $\delta$ of the known practical attacks against the protocol may vary, Luby [12] proposed the *time-success ratio* $\frac{t}{\delta}$ as a universal security measure. This means that a protocol is $S$-secure, if the success probability of any $t$-time adversary does not exceed $\frac{t}{S}$.

In a typical security proof for a protocol $\mathcal{P}$ built from a primitive $\mathcal{Q}$, it is shown that if $\mathcal{Q}$ is $S_q$-secure, then $\mathcal{P}$ is $S_p$-secure. Bellare and Rogaway [2,3] first emphasized the importance of studying the *tightness* of security proofs in practical applications. Informally, tightness shows how much security of the primitive is transformed to the protocol. Numerically, we may express tightness as the ratio $S_p/S_q$. The notion opposite to tightness is *security loss*.

Security proofs are often presented as *reductions*, i.e. we assume that we have an adversary for the protocol $\mathcal{P}$ with running time $t$ and success probability $\delta$ and then construct an adversary for the primitive $\mathcal{Q}$ with running time $t'$ and success probability $\delta'$. This means that for having a protocol that is secure against adversaries with running time $t$ and with success probability $\delta$, we have to use a $\frac{t'}{\delta'}$-secure primitive. The ratio $\frac{t'}{\delta'}$ is a function of $t$ and $\delta$, but not always the function of $\frac{t}{\delta}$. For example, if $\frac{t'}{\delta'} = c \cdot \frac{t}{\delta^2}$, then there is no dependence of type $\frac{t'}{\delta'} = F\left(\frac{t}{\delta}\right)$. However, due to $t \geq 1$, we have an inequality $\frac{t'}{\delta'} \leq c \cdot \left(\frac{t}{\delta}\right)^2$.

## 2.2 Security Properties of Hash Functions

In this paper, we study the security properties of hash functions $H^P$ that use some kind of ideal functionality $P$ (random permutations, random functions, ideal ciphers, etc.) as an oracle. For example, in case of the Merkle-Damgård hash functions, the compression function and the output transform are often assumed to be ideal objects. In this section, we describe some of the properties of hash functions, starting from the strongest ones.

**Random- and Pseudorandom Oracles.** By a *fixed length (FIL-) random oracle* $\mathcal{R}$, we mean a function that is chosen randomly from the set of all functions of type $\{0,1\}^m \to \{0,1\}^n$. There are $(2^n)^{2^m} = 2^{n2^m}$ possible choices of $\mathcal{R}$. By a *variable length (VIL-) random oracle* $\mathcal{V}$, we mean a function that is chosen randomly from the set $\Omega$ of all functions of type $\{0,1\}^* \to \{0,1\}^n$. The probability distribution of $\mathcal{V}$ is defined so that for any fixed input length $m$, the restriction of $\mathcal{V}_m$ to $\{0,1\}^m$ is distributed like a FIL-random oracle.

By the *random oracle heuristic* we mean a security argument when an application of a hash function (e.g. a time-stamping scheme, a signature scheme) is proved to be secure in the so-called *random oracle model*, where the hash function is replaced with a VIL-random oracle. The random oracle heuristic was first introduced by Bellare and Rogaway [2]. Although it was proved later by Canetti et al [7] that the random oracle heuristic fails in certain theoretical cases, proofs in the random oracle model are still considered as valuable security arguments, especially if no better security proofs are known.

**Definition 1 (PRO).** *We say that $H^P$ is a* pseudo-random oracle (PRO) *if there is an efficient simulator $S^{\mathcal{V}}$, such that for every efficient distinguisher $D$, the following difference is negligible:*

$$\left| \mathsf{Pr}\left[1 \leftarrow D^{H^P,P}\right] - \mathsf{Pr}\left[1 \leftarrow D^{\mathcal{V},S^{\mathcal{V}}}\right] \right| \quad .$$

This notion is first studied by Maurer et al [13] and was adapted to hash functions by Coron et al [9]. The most important practical implication of the pseudo-random oracle property of $H^P$ is that any application (e.g. a time-stamping scheme) that uses $H^P$ as a hash function is almost as secure as if a variable length random oracle $\mathcal{V}$ is used instead of $H^P$. This means that the *random oracle heuristics* applies in case of the particular application, i.e. we can prove the security of the application in the random oracle model and then replace the oracle by a more realistic (but still ideal!) model $H^P$ of the hash function. Note that the PRO-property is a very strong assumption and often we would like to know if some lighter assumptions would also be sufficient for the security of the application. For example, it was shown [9] that the commonly used Merkle-Damgård style hash functions do not satisfy the PRO property.

**Pre-image Awareness.** Informally, pre-image awareness of a (hash) function $H$ means, that if we first commit an output $y$ and later come up with an input $x$, such that $y = H(x)$, then it is safe to conclude that we knew $x$ before committing $y$. This notion was first formalized by Dodis et al. [10] for hash functions $H^P$ that are built using an ideal primitive $P$ in a black box way. For $H^P$ being pre-image aware, there has to be an efficient deterministic algorithm $\mathcal{E}$ (the so-called *extractor*) which when given $y$ and the list $\alpha$ of all previously made $P$-calls (by the adversary), outputs $x$, such that $H^P(x) = y$, or $\perp$ if $\mathcal{E}$ was unable to find such an $x$. The adversary tries to find $x$ and $y$ so that $\mathcal{E}(\alpha, y) \neq x$ and $y = H^P(x)$.

To define pre-image awareness of $H^P$ in a precise way, we set up an experiment **Exp** (see Figure 1), specified as a game which an attacker $B$ is trying to win. $B$ is constrained to oracle access to $P$, via a wrapper oracle $\mathsf{P}$, which records all $P$-calls made by $B$ as an advise string $\alpha$. Likely, the extractor $\mathcal{E}$ is also accessible through another wrapper oracle $\mathsf{Ex}$, which uses global arrays $\mathtt{Q}$ (initially $\perp$ everywhere) and $\mathtt{V}$ (initially blank). $\mathtt{Q}$ is used to record all input parameters to $\mathcal{E}$; $\mathtt{V}$ is used to store all successfully extracted values corresponding to $\mathcal{E}$'s inputs. The adversary $B$ tries to output a value $x$ such that $H^P(x) = y$, $\mathtt{Q}[y] = 1$ and $\mathtt{V}[y] \neq x$, i.e. $\mathcal{E}$ tried to invert $y$, but was either unsuccessful ($\mathtt{V}[y] = \perp$) or found a different pre-image $x' \neq x$ (a *collision* for $H^P$). As $\mathsf{P}$- and $\mathsf{Ex}$-calls are unit cost, the running time of $B$ does not depend on the running time of $\mathcal{E}$.

**Definition 2 (Pre-image Awareness).** *A function $H^P$ is $S$-secure* pre-image aware *(PrA) if there is an efficient extractor $\mathcal{E}$, so that for every $t$-time $B$:*

$$\mathbf{Adv}^{\mathrm{pra}}_{H,P,\mathcal{E}}(B) = \mathsf{Pr}\left[1 \leftarrow \mathbf{Exp}^{\mathrm{pra}}_{H,P,\mathcal{E},B}\right] \leq \frac{t}{S} \quad . \tag{1}$$

It is easy to see that pre-image awareness of $H^P$ implies collision-resistance of $H^P$. Hence, as there is the so-called birthday bound for the collision-resistance which says that no function with $n$-bit output can be more than $2^{n/2}$-secure, we conclude that no function with $n$-bit output can be more than $2^{n/2}$-secure pre-image aware. The Pre-image awareness property alone is not sufficient for proving the PRO property. However, together with some additional assumptions this has been proved to be possible [10,8].

```
$\mathbf{Exp}^{\mathrm{pra}}_{H,P,\mathcal{E},B}$:
    $x \leftarrow B^{\mathsf{P},\mathsf{Ex}}$
    $y \leftarrow H^P(x)$
    If $\mathtt{Q}[y] = 1$ and $\mathtt{V}[y] \neq x$ return 1,
    else return 0
```

```
oracle $\mathsf{P}(m)$:
    $c \leftarrow P(m)$
    $\alpha \leftarrow \alpha || (m, c)$
    return $c$
```

```
oracle $\mathsf{Ex}(y)$:
    $\mathtt{Q}[y] \leftarrow 1$
    $\mathtt{V}[y] \leftarrow \mathcal{E}(y, \alpha)$
    return $\mathtt{V}[y]$
```

**Fig. 1.** Preimage awareness experiment and the wrapper oracles

**Collision Resistance.** Informally, the collision resistance of a hash function $H^P$ means that it is infeasible for adversaries to find two different inputs $x$ and $x'$ that have the same hash value, i.e. $H^P(x) = H^P(x')$. This definition makes sense only if the ideal primitive $P$ contains some randomness, because for fixed functions, there always exist collisions that can be "wired" into the adversary.

**Definition 3 (Collision Resistance).** *A function $H^P$ is $S$-secure* collision resistant *(CR) if for every adversary $B$ with running time $t$:*

$$\mathbf{Adv}^{\mathrm{cr}}_{H,P}(B) = \mathsf{Pr}\left[x, x' \leftarrow B^P\colon x \neq x',\ H^P(x) = H^P(x')\right] \leq \frac{t}{S} \ . \qquad (2)$$

Note that due to the so-called Birthday attack, functions with $n$-bit output can only be $2^{\frac{n}{2}}$-secure collision resistant.

## 2.3 Hash-Tree Time-Stamping Schemes and Their Security

Hash trees were introduced by Merkle [14]. Let $h\colon \{0,1\}^{2n} \to \{0,1\}^n$ be a hash function. By a *hash-tree* we mean a tree-shaped data structure that consists of nodes each of which contains an $n$-bit hash value. Each node is either a *leaf* which means it has no child nodes, or an *internal node* that has two child nodes (the left- and the right child) whereas the hash value $y$ of an internal node is computed as a hash $y = h(y_0, y_1)$, where $y_0$ and $y_1$ are the hash values of the left- and the right child, respectively. There is one *root node* that is not a child of any other node. If $\mathcal{T}$ is a hash tree with $m$ leaves with hash values $x_1, \ldots, x_m$ and $r$ is the hash value of the root node, then we use the notation $r = \mathcal{T}(x_1, \ldots, x_m)$.

**Encoding the Leaves of a Hash Tree.** Each node in a hash tree can be *naturally named* by using finite bit-strings in the following way. The root node is named by the empty string $\|$. If a node is named by a bit-string $\ell$, then its left- and right child nodes are named by $\ell 0$ and $\ell 1$, respectively. The name $\ell$ of a node resembles an *address* of the node, considering that one starts the searching process from the root node, and then step by step, chooses one of the child nodes depending on the corresponding bit in $\ell$, i.e. 0 means "left" and 1 means "right".

**Shape of a Hash Tree.** Hash tree has a particular *shape* by which we mean the set of all names of the leaf-nodes. For example a balanced complete tree with four nodes (Fig. 2, left) has shape $\{00, 01, 10, 11\}$. If the root hash value is denoted by $r$ (instead of $r_{[]}$) and $r_\ell$ denotes the hash value of a node with name $\ell$, then in this example, the relations between the nodes are the following: $r = h(r_0, r_1)$, $r_0 = h(r_{00}, r_{01})$, and $r_1 = h(r_{10}, r_{11})$. The shape $\{000, 001, 01, 1\}$ represents a totally unbalanced tree with four leaves (Fig. 2, right), with the hash values being in the following relations: $r = h(r_0, r_1)$, $r_0 = h(r_{00}, r_{01})$, and $r_{00} = h(r_{000}, r_{001})$. Note also that the shape is always a *prefix-free code.*

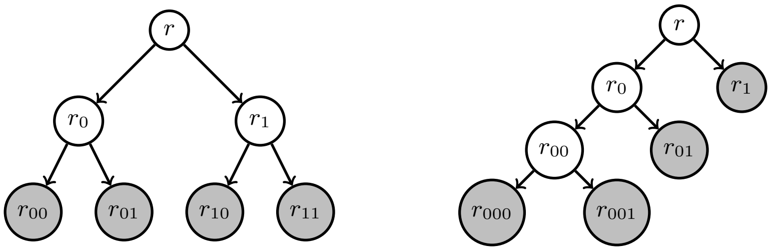


**Fig. 2.** A balanced tree (left) and an unbalanced tree (right)

**Hash Chains.** In order to prove that a hash value $r_\ell$ (where $\ell_1\ell_2 \ldots \ell_m$ is the binary representation of $\ell$) participated in the computation of the root hash value $r$, it is sufficient to present all the sibling hash values of the nodes on the unique path from $r_\ell$ to the root $r$. For example, in the balanced tree with four leaves (Fig. 2, left), to prove that $r_{01}$ belongs to the tree, we have to show the sibling hash values $r_{00}$ and $r_1$, which enable a verifier to compute $r_0 = h(\underline{r_{00}}, r_{01})$ and $r = h(r_0, \underline{r_1})$. In general, we define a hash chain as follows:

**Definition 4 (Hash-chain).** *A* hash-link *from $x$ to $r$ (where $x, r \in \{0,1\}^n$) is a pair $(s, b)$, where $s \in \{0,1\}^n$ and $b \in \{0,1\}$, such that either $b = 0$ and $r = h(x\|s)$, or $b = 1$ and $r = h(s\|x)$. A* hash-chain *from $x$ to $r$ is a (possibly empty) list $c = ((s_1, b_1), \ldots, (s_m, b_m))$, such that either $c = ()$ and $x = r$; or there is a sequence $x_0, x_1, \ldots, x_m$ of hash values, such that $x = x_0$, $r = x_m$, and $(s_i, b_i)$ is an hash-link from $x_{i-1}$ to $x_i$ for every $i \in \{1, \ldots, m\}$. We denote by $x \overset{c}{\rightsquigarrow} r$ the proposition that $c$ is a hash chain from $x$ to $r$. Note that $x \overset{()}{\rightsquigarrow} x$ for every $x \in \{0,1\}^n$. By the* shape *$\ell(c)$ of $c$ we mean the $m$-bit string $b_1 b_2 \ldots b_m$.*

**Hash-Tree Time-Stamping Schemes.** The time-stamping procedure runs as follows. During every time unit $t$ (e.g. one second) the time-stamping server receives a list $\mathcal{X}_t = (x_1, \ldots, x_m)$ of requests ($n$-bit hash values) from clients, computes the root hash value $r_{(t)} = \mathcal{T}(x_1, \ldots, x_m)$ of a hash tree $\mathcal{T}$ and publishes $r_{(t)}$ in a public repository $\mathcal{R} = (r_{(1)}, r_{(2)}, \ldots, r_{(t)})$ organized as an append-only list. Each request $x_i$ is then provided with a hash chain $c_i$ (the *time stamp* for $x_i$) that proves the participance of $x_i$ in the computation of the root hash value

$r_{(t)}$. A request $x \in \mathcal{X}_t$ is said to precede another request $x' \in \mathcal{X}_{t'}$ if $t < t'$. The requests of the same batch are considered simultaneous. In order to verify the time stamp $c_i$ of a request $x_i$, one computes the output hash value of $c_i$ (the last hash value $x_m$ in the sequence) and checks whether $x_m = r$.

**Bounded, Unbounded and Shape-Compact Time-Stamping Schemes.** It was first mentioned by Buldas et al [6] that for proving the security of time-stamping schemes, there must be restrictions on the shape $\ell(c)$ of the hash chains that are acceptable in time stamps. In general, we denote the set of allowed shapes by $\mathcal{S}$. If $\mathcal{S}$ is finite then there is always a binary tree $\mathcal{T}(\mathcal{S})$ such that for any $\ell \in \mathcal{S}$ it contains a node with name $\ell$. For example, if $\mathcal{S} = \{0, 000, 01, 1, 11, 100\}$, the smallest such binary tree (assuming that every non-leaf node has two children) is depicted in Fig. 3. We say that a time-stamping scheme is *unbounded*,

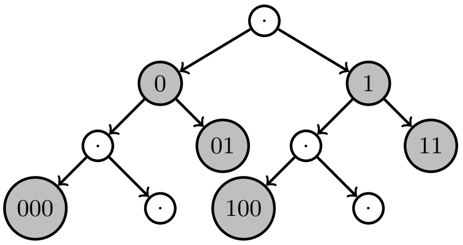


**Fig. 3.** The smallest binary tree induced by $\mathcal{S} = \{0, 000, 01, 1, 11, 100\}$

if $\mathcal{S} = \{0,1\}^*$. We say that a hash-tree time-stamping scheme is *C-bounded* if $|\mathcal{S}| \leq C$. A $C$-bounded hash-tree time-stamping scheme is said to be *shape-compact* if the tree $\mathcal{T}(\mathcal{S})$ induced by allowed shapes has no more than $2C$ vertices and $|\ell| \leq 2\log_2 C$ for every $\ell \in \mathcal{S}$, where $|\ell|$ denotes the bit length of $\ell$.

**Security of Time-Stamping.** Informally, we want that no efficient adversary can *back-date* any request $x$, i.e. first publishing a hash value $r$, and only after that generating a new "fresh" $x$ (not pre-computed by the adversary), and a hash chain $c$, so that $x \overset{c}{\rightsquigarrow} r$. To formalize the security condition, we use the so-called *entropy-based security* condition [5] inspired by the following attack-scenario with a two-stage adversary $A = (A_1, A_2)$ cooperating with the server. At the first stage $A_1$ the adversary forces server to create a public repository $\mathcal{R}$ of commitments. Note that there is no guarantee that the hash values in $\mathcal{R}$ are created in the proper way, i.e. by using the hash tree. After that, the second stage $A_2$ of the adversary presents a high-entropy (unpredictable) $x$ and a hash chain $c$ so that $x \overset{c}{\rightsquigarrow} r$ for an $r \in \mathcal{R}$. The unpredictability of $x$ is crucial because otherwise $x$ could have been pre-computed (guessed) by $A$ before $r$ is published and hence $x$ could be in fact older than $r$ and thereby not back-dated by $A$.

Hence, for defining the security of time-stamping schemes, the class of possible adversaries is restricted. Only *unpredictable* adversaries that produce unpredictable $x$ are considered, i.e. the output component $x$ is assumed to be unpredictable even if the contents of $\mathcal{R}$ and all the internal computations of the adversary (while computing $\mathcal{R}$) are known. The original security definition from [5] is somewhat inconvenient to use for exact security estimations, because it extensively uses the polynomial security model. In this paper, we slightly weaken the adversary by assuming the so-called *strong unpredictability.* Intuitively, the strong unpredictability means that $x \in \{0,1\}^n$ is almost identically distributed, i.e. its conditional min-entropy $H_\infty[x \mid \mathcal{R}, a]$ must be at least $n-1$ bits, i.e. for every input of $A_2$ and for any possible value $x_0$ of $x$, the probability of $x = x_0$ is upper bounded by $\frac{1}{2^{n-1}}$. For practical justification of such an assumption, note that in practical applications, $x$ is mostly a cryptographic hash of a (much) longer document $X$ that contains a considerable amount of new (fresh) information. Cryptographic hash functions are assumed to be good entropy-extractors, and hence the assumption of strong unpredictability is practically reasonable.

**Definition 5 (Security against back-dating).** *A time-stamping scheme is $S$-secure against back-dating if for every $t$-time strongly unpredictable $(A_1, A_2)$:*

$$\mathsf{Pr}\left[(\mathcal{R}, a) \leftarrow A_1, (x,c) \leftarrow A_2(\mathcal{R}, a)\colon\ x \overset{c}{\rightsquigarrow} \mathcal{R},\ \ell(c) \in \mathcal{S}\right] \le \frac{t}{S}\ , \tag{3}$$

*where by $x \overset{c}{\rightsquigarrow} \mathcal{R}$ we mean that $x \overset{c}{\rightsquigarrow} r$ for some $r \in \mathcal{R}$, and $a$ is an advice string that contains possibly useful information that $A_1$ stores for $A_2$.*

**Existing Security Proofs and Their Tightness.** We present the tightness parameters of two known security proofs for hash-tree time-stamping schemes: the first correct proof [6] that was presented in Asiacrypt 2004, and a tighter proof [4] from ACISP 2010 that was also proved to be asymptotically optimally tight. Both proofs assume the collision-resistance property of the hash function. Both proofs apply only to $N$-bounded time-stamping schemes and their tightness depends on the capacity $N$ of the system. It was proved in [6] by using oracle separation that collision-resistance is insufficient for proving the security of *unbounded* time-stamping schemes. Both proofs are in the form of a reduction: a $t$-time backdating adversary with success probability $\delta$ is converted to a $t'$-time collision-finding adversary with success probability $\delta'$.

In Tab. 1 we present the parameters of these two security proofs. Closer analysis shows that the parameter $N$ used in these security proofs [6,4] can be expressed by $N = C \cdot |\mathcal{R}|$ in terms of this paper, where $\mathcal{R}$ is the hash repository created by the first stage $A_1$ of the back-dating adversary. As $|\mathcal{R}|$ is always upper bounded by the running time $t$ of the adversary, we have $N \le C \cdot t$. The third column of Tab. 1 presents the converted tightness formulae assuming $N \approx Ct$. The fourth column presents a formula for the required output size $n$ of the hash function, assuming that we want the time-stamping scheme to be $S$-secure and that the hash function is secure near to the birthday barrier, i.e. $n$-bit hash function is assumed to be $2^{n/2}$-secure. The last column presents the output size in a particular case, where $S = 2^{80}$ (standard requirement for "commercial" security), and $C = 2^{64}$, i.e. acceptable hash chains are assumed to be no longer than 64 steps. Note that this does not mean that we assume a hash-tree with $2^{64}$ nodes is built in every second! The exponent 64 is only a theoretical bound for the length of a hash chain, which is quite realistic in a system of global scale with a multi-level structure of servers all over the world that compute hash trees.

The required output size 312 is too large, because one would like to use smaller hash functions such as SHA2-256 in time-stamping schemes. As the proof of ACISP 2010 is optimally tight, we have no hope to construct tighter proofs under the collision-resistance assumption. Schemes with smaller hash functions can only be proved secure under assumptions stronger than collision-resistance.

**Table 1.** Tightness parameters of two security proofs

| Proof | Formula | Output Size Formula $n = n(C, S)$ | $n(2^{64}, 2^{80})$ |
|---|---|---|---|
| Asiacrypt 2004 | $\frac{t'}{\delta'} \approx 2C\left(\frac{t}{\delta}\right)^2$ | $n = 2\log_2 C + 4\log_2 S + 2$ | 448 |
| ACISP 2010 | $\frac{t'}{\delta'} \approx 14\sqrt{C}\left(\frac{t}{\delta}\right)^{1.5}$ | $n = \log_2 C + 3\log_2 S + 8$ | 312 |

## 3 Security under RO and PrA Assumptions

**Theorem 1.** *If $h : \{0,1\}^{2n} \to \{0,1\}^n$ is a random oracle, then the corresponding (bounded or unbounded) hash-tree time-stamping schemes are $2^{\frac{n-1}{2}}$-secure.*

*Proof.* Let $A = (A_1, A_2)$ be a strongly unpredictable adversary with running time $t$, as described in (3). Let $t_1, t_2$ denote the running times of $A_1$ and $A_2$, respectively. Considering that $(\mathcal{R}, a) \leftarrow A_1$, let $R_1 \subseteq \{0,1\}^n$ be the set of all $x$-s so that the $h$-calls performed by $A_1$ induce a hash-chain from $x$ to an $r \in \mathcal{R}$. Note that $\mathcal{R} \subseteq R_1$, as an empty hash-chain is always induced by any set of $h$-calls. We assume without loss of generality that the advice string $a$ contains $R_1$. Because of strong unpredictability of $A$, we have $\mathsf{Pr}\left[x \in R_1\right] \leq \frac{|R_1|}{2^{n-1}}$.

In case of $x \notin R_1$, in order to be successful, $A_2$ has to make additional $h$-calls so that a chain from $x$ to $r \in \mathcal{R}$ is induced. A necessary condition that $A_2$ has to satisfy is that it has to find $x' = x'_1 \| x'_2$ so that $x'_1 \notin R_1$ or $x'_2 \notin R_1$ (this means that $A_1$ did not make $h$-calls with input $x'$), but $h(x') \in R_1$. The probability of this condition does not exceed $t_2 \frac{|R_1|}{2^n}$, hence, considering that $|R_1| \leq 2t_1$ and $t_1, t_2 \geq 1$, the overall success probability $\delta$ of $A$ can be estimated as follows:

$$\delta \leq \frac{|R_1|}{2^{n-1}} + \left(1 - \frac{|R_1|}{2^{n-1}}\right) t_2 \frac{|R_1|}{2^n} \leq \frac{2t_1}{2^{n-1}} + \frac{t_1 t_2}{2^{n-1}} \leq \frac{3t_1 t_2}{2^{n-1}} \leq \frac{(t_1 + t_2)^2}{2^{n-1}} = \frac{t^2}{2^{n-1}} .$$

Hence, as $\delta^2 \leq \delta \leq \frac{t^2}{2^{n-1}}$, we have $\frac{t}{\delta} \geq 2^{\frac{n-1}{2}}$. □

**Corollary 1.** *Hash-tree time-stamping schemes (bounded/non-bounded) are $S$-secure in the RO model if they use hash functions with $2\log_2 S + 1$ output bits.*

**Theorem 2.** *If $H^P\colon \{0,1\}^{2n} \to \{0,1\}^n$ is a hash-function built from an ideal primitive $P$ that is $S$-secure PrA, then the corresponding $C$-bounded shape-compact hash-tree time-stamping schemes are $\frac{S}{2C}$-secure against strongly unpredictable adversaries with running time $t \ll \frac{2^n}{C}$.*

*Proof.* Due to the PrA assumption there exists an efficient extractor $\mathcal{E}$. Let $A^P = (A_1^P, A_2^P)$ be a strongly unpredictable back-dating adversary with running time $t \ll 2^n/C$ and success probability $\delta$.

We construct a PrA-adversary $B^{\mathsf{P},\mathsf{Ex}}$ that first simulates $(\mathcal{R}, a) \leftarrow A_1^P$ so that all $P$-calls of are executed through the $\mathsf{P}$-oracle. After that, for every $r \in \mathcal{R}$, the adversary builds a hash-tree by using the $\mathsf{Ex}$-oracle in the following way. It calls $z \leftarrow \mathsf{Ex}(r)$ and if $z = \perp$, then no more extractions are performed. If $z \neq \perp$ (this means $r = H^P(z)$), then $B$ assigns $r_0 = z_{1\ldots n}$ and $r_1 = z_{n+1\ldots 2n}$, i.e. $r_0$ equals to the first $n$ bits of $z$ and $r_1$ equals to the last $n$ bits of $z$. The same procedure is then applied to $r_0$ and $r_1$, etc. until the whole hash-tree $\mathcal{T}(\mathcal{S})$ (induced by the set $\mathcal{S}$ of allowed shapes) is filled with hash values. If the extractor fails in some branches of the tree, the extraction procedure is stopped in this branch but is continued in other branches. For example, in case $\mathcal{S} = \{00, 01, 10, 11\}$ the adversary tries to extract $(r_0, r_1) \leftarrow \mathsf{Ex}(r)$, $(r_{00}, r_{01}) \leftarrow \mathsf{Ex}(r_0)$, $(r_{10}, r_{11}) \leftarrow \mathsf{Ex}(r_1)$. Due to the shape-compactness, the $\mathsf{Ex}$-oracle is called no more than $C \cdot |\mathcal{R}| \leq Ct$ times.

Finally, $B$ simulates $A_2^P$ so that all its $P$ calls are executed through the $\mathsf{P}$-oracle. With probability $\delta$ we obtain a hash value $x$ and a hash chain $c$ such that $\ell(c) \in \mathcal{S}$ and $x \overset{c}{\rightsquigarrow} r$ for some $r \in \mathcal{R}$. Due to the strong unpredictability of $A$ and $Ct \ll 2^n$, the probability that $x$ coincides with some of the extracted hash values $r_\ell$ is upper bounded by $\frac{2Ct}{2^{n-1}}$ which is negligible. Hence, with probability almost $\delta$, we have a hash value $x$ that is not in the extracted tree, and still there is a hash chain $c = \{(c_1, b_1), (c_2, b_2), \ldots, (c_m, b_m)\}$ with output hash value $r$ that certainly is in the extracted tree. Let $x_0, x_1, \ldots, x_m$ be the intermediate hash values (outputs of hash links) as described in Def. 4. Let $k$ be the smallest index such that $x_{k-1}$ is not in the extracted tree but $x_k$ is. For such $k$,

$$\begin{cases} \mathsf{H}^{\mathsf{P}}(c_k \| x_{k-1}) = x_k \text{ and } \mathsf{Ex}(x_k) \neq (c_k \| x_{k-1}) & \text{if } b_k = 0 \ ; \\ \mathsf{H}^{\mathsf{P}}(x_{k-1} \| c_k) = x_k \text{ and } \mathsf{Ex}(x_k) \neq (x_{k-1} \| c_k) & \text{if } b_k = 1 \ . \end{cases}$$

The adversary $B$ outputs $(c_k \| x_{k-1})$ if $b_k = 0$ or $(x_{k-1} \| c_k)$ if $b_k = 1$. Hence, an adversary $B$ with approximate running time $t' = Ct + t \leq 2Ct$ wins the PrA experiment with probability $\delta' \approx \delta$. Hence, $\frac{t}{\delta} \geq \frac{t'}{2C\delta'} \geq \frac{S}{2C}$. □

**Analysis.** Tightness of security proofs under the CR, PrA, and RO assumptions is compared in Tab. 2. In case of PrA, we assume that the hash function is about $2^{n/2}$-secure, which is a limit because PrA implies CR and the birthday barrier applies. We see that even though PrA seems to be much stronger than CR, the required output length is not much smaller. This is because the security loss is linear in $C$ and not in $\sqrt{C}$ as in the case of the CR assumption.

**Table 2.** Tightness parameters of security proofs under different assumptions

| Assumption | Formula | Output Size Formula $n = n(C, S)$ | $n(2^{64}, 2^{80})$ |
|---|---|---|---|
| CR | $\frac{t'}{\delta'} \approx 14\sqrt{C}\left(\frac{t}{\delta}\right)^{1.5}$ | $n = \log_2 C + 3\log_2 S + 8$ | 312 |
| PrA | $\frac{t'}{\delta'} \approx 2C\frac{t}{\delta}$ | $n = 2(\log_2 C + \log_2 S + 1)$ | **290** |
| RO | $\frac{t}{\delta} \geq 2^{\frac{n-1}{2}}$ | $n = 2\log_2 S + 1$ | 161 |

## 4 Strong Pre-image Awareness

By studying the proof of Theorem 2, we may see the following way to improve the tightness of the proof. Why should we extract the whole tree $T(\mathcal{S})$? Maybe we just extract the *hash chain* $x \overset{c}{\rightsquigarrow} r$ instead. This would mean that we call the $\mathsf{Ex}$-oracle only $2\log_2 C$ times instead of $C\cdot|\mathcal{R}|$ times. However, this is not allowed. First, we do not know $x$ before simulating $A_2$ and second, after executing $A_2$, we have changed the string $\alpha$ in the wrapper oracle. If we extract the tree $T(\mathcal{S})$ after executing $A_2$, then the contents of the tree (the hash values) depend on $x$ and we cannot apply the strong unpredictability argument. The probability that $x$ is in the tree may not be negligible any more. But, what if we still extract the tree after the execution of $A_2$ but use the same "old" $\alpha$ that was created after the simulation of $A_1$? The problem is that though we are able to execute the extractor $\mathcal{E}$ directly with an old $\alpha$, the results will not be saved in the arrays $\mathtt{V}$ and $\mathtt{Q}$ and the adversary is unable to win the game for "technical reasons".

This inspires a new stronger notion of PrA in which "old" advise strings $\alpha$ can be used in those queries $\mathsf{Ex}(y)$ where $y$ is created not later than $\alpha$ was formed. So, we define a new $\mathsf{Ext}$-oracle that always uses the "oldest" possible $\alpha$. For example, if we obtain $x \leftarrow \mathsf{Ext}(y)$ (where $x = x_1x_2$ and $x_1, x_2 \in \{0,1\}^n$) for which the oracle uses $\alpha$, and later we call $\mathsf{Ext}(x_1)$, the same $\alpha$ is used for extraction, because the oracle remembers that $x_1$ was created by just "parsing" $x$ and it is thereby as old as $x$ and the use of $\alpha$ is "legal". If the ordinary $\mathsf{Ex}$ oracle is replaced with the $\mathsf{Ext}$-oracle, then we call the corresponding security property *strong pre-image awareness (SPrA)*. For the SPrA condition to make sense for functions with variable input length, we use the notion of a *parser*. We also assume a quite natural additional property of the extractor algorithm. We use the notation $\alpha \subseteq \alpha'$ to mean that $\alpha$ is an initial segment of $\alpha'$, i.e. $\alpha'$ is created from $\alpha$ by adding some extra pairs to the top of it.

**Definition 6 (Natural extractor).** *An extractor $\mathcal{E}$ is* natural*, if $\alpha \subseteq \alpha'$ and $\mathcal{E}(\alpha, x) \neq \mathcal{E}(\alpha', x)$ implies $\mathcal{E}(\alpha, x) = \bot$.*

**Definition 7 ($n$-parser).** *By an $n$-parser $\Pi$, we mean any efficiently computable deterministic (not necessarily invertible) function that converts a finite bit-string $x$ to a finite set $\mathcal{X}_x$ of $n$-bit strings.*

```
$\mathbf{Exp}^{\mathrm{spra}}_{H,P,\mathcal{E},\Pi,A}$:
$x \leftarrow A^{\mathsf{P},\mathsf{Ext}_\Pi}$
$y \leftarrow H^P(x)$
If $\mathtt{Q}[y] = 1$ and $\mathtt{V}[y] \neq x$
return 1,
else return 0
```

```
oracle $\mathsf{P}(m)$:
$c \leftarrow P(m)$
$\alpha \leftarrow \alpha || (m, c)$
return $c$
```

```
oracle $\mathsf{Ext}_\Pi(y)$:
if $\mathtt{A}[y] = \perp$ then $\mathtt{A}[y] \leftarrow \alpha$
$\mathtt{Q}[y] \leftarrow 1$
$\mathtt{V}[y] \leftarrow \mathcal{E}(y, \mathtt{A}[y])$
if $\mathtt{V}[y] \neq \perp$ and $\Pi(V[y]) = \{y_1, \ldots, y_m\}$ then for every $i = 1 \ldots m$:
    if $\mathtt{A}[y_i] = \perp$ then $\mathtt{A}[y_i] \leftarrow \mathtt{A}[y]$
return $\mathtt{V}[y]$
```

**Fig. 4.** Strong pre-image awareness experiment and the wrapper oracles

**Definition 8 (Strong Pre-image Awareness).** *A function* $H^P : \{0,1\}^* \to \{0,1\}^n$ *is S-secure* strongly pre-image aware *(SPrA) if there is an efficient natural extractor* $\mathcal{E}$*, so that for every n-parser* $\Pi$ *and for every t-time A:*

$$\mathbf{Adv}^{\mathrm{spra}}_{H,P,\mathcal{E},\Pi}(A) = \mathsf{Pr}\left[1 \leftarrow \mathbf{Exp}^{\mathrm{spra}}_{H,P,\mathcal{E},\Pi,A}\right] \leq \frac{t}{S} \ . \tag{4}$$

It can be shown that random oracles are SPrA and the next theorem shows that SPrA is a property the strength of which lies between RO and PrA.

**Theorem 3.** *SPrA implies PrA.*

*Proof.* Let $H^P$ be strongly pre-image aware with the corresponding natural extractor $\mathcal{E}$ and let $B$ be an PrA-adversary. We define an SPrA-adversary $A$ so that it simulates $B$ and its $\mathsf{P}$- and $\mathsf{Ex}$ oracle calls, ret aining their results in arrays $\alpha'$, $\mathtt{Q}'$ and $\mathtt{V}'$ that $A$ maintains itself. In parallel, $A$ calls the real oracles $\mathsf{P}$ and $\mathsf{Ext}$ that maintain the arrays $\mathtt{Q}$ and $\mathtt{V}$. For $B$, the simulated oracles behave exactly like $\mathsf{P}$ and $\mathsf{Ex}$ (Fig. 5). However, for some of the outputs $y$ it might be

```
$\mathbf{Exp}^{\mathrm{spra}}_{H,P,\mathcal{E},\Pi,A}$:
$x \leftarrow A^{\mathsf{P},\mathsf{Ext}_\Pi} \equiv \begin{cases} \text{Initialize } \alpha', \mathtt{V}' \text{ and } \mathtt{Q}' \\ x \leftarrow B^{\mathsf{P}',\mathsf{Ex}'} \end{cases}$
$y \leftarrow H^P(x)$
If $\mathtt{Q}[y] = 1$ and $\mathtt{V}[y] \neq x$ return 1,
else return 0
```

```
simulated $\mathsf{P}'(m)$:
call $c \leftarrow \mathsf{P}(m)$
$\alpha' \leftarrow \alpha' || (m, c)$
return $c$
```

```
simulated $\mathsf{Ex}'(y)$:
$\mathtt{Q}'[y] \leftarrow 1$
$\mathtt{V}'[y] \leftarrow \mathcal{E}(y, \alpha')$
call $v \leftarrow \mathsf{Ext}_\Pi(y)$
return $\mathtt{V}'[y]$
```

**Fig. 5.** Simulation of oracles by $\mathbf{Exp}^{\mathrm{spra}}_{H,P,\mathcal{E},\Pi,A}$

that $\mathcal{E}(\alpha, y) = \mathtt{V}[y] \neq \mathtt{V}'[y] = \mathcal{E}(\alpha', y)$ because the extractions are made using different advise strings. But our construction guarantees that $\alpha \subseteq \alpha'$ and thus

$$\mathtt{V}[y] \neq \mathtt{V}'[y] \quad \Longrightarrow \quad \mathtt{V}[y] = \perp \tag{5}$$

because $\mathcal{E}$ is natural. If $B$ is successful, i.e. finds $x$ with output $y = H^P(x)$, such that $\mathtt{Q}'[y] = 1$ and $\mathtt{V}'[y] \neq x$, then also $\mathtt{Q}[y] = 1$ and $\mathtt{V}[y] \neq x$, because $\mathtt{V}[y] = x$ implies $\mathtt{V}[y] \neq \mathtt{V}'[y]$ which by (5) gives $x = \mathtt{V}[y] = \perp$, a contradiction. Hence, $\mathbf{Adv}^{\mathrm{pra}}_{H,P,\mathcal{E}}(B) \leq \mathbf{Adv}^{\mathrm{spra}}_{H,P,\mathcal{E},\Pi}(A)$. □

**Theorem 4.** *If $H^P\colon \{0,1\}^{2n} \to \{0,1\}^n$ is a hash function with ideal primitive $P$ that is $S$-secure SPrA, then the corresponding $C$-bounded shape-compact hash-tree time-stamping schemes are $\frac{S}{4\log_2 C}$-secure against strongly unpredictable adversaries with running time $t \ll \frac{2^n}{C}$.*

*Proof.* Let $\mathcal{E}$ be an SPrA-extractor and $A^P = (A_1^P, A_2^P)$ be a strongly unpredictable back-dating adversary with running time $t \ll 2^n/C$ and success $\delta$. We construct an SPrA-adversary $B^{\mathsf{P},\mathsf{Ext}}$ which first simulates $(\mathcal{R}, a) \leftarrow A_1$ and then calls $\mathsf{Ext}(r)$ for every $r \in \mathcal{R}$. This step fixes the advice string $\alpha$, thus the following tree extraction will be independent of the computations of $A_2$. After that, $B$ simulates $(x, c) \leftarrow A_2^P(\mathcal{R}, a)$ so that all $P$-calls of $H$ are executed through the $\mathsf{P}$-oracle. Note that as $Ct \ll 2^n$, the probability that $x$ is in the extraction tree $\mathcal{T}(\mathcal{S})$ is negligible. Finally, $B$ uses the $\mathsf{Ext}$-oracle to extract the hash values along the hash chain $c$ (there are at most $2\log_2 C$ of them). The proof is exactly like in the case of the PrA assumption, except we have a much smaller tree (i.e. a chain instead of a tree). Hence, an adversary $B$ with approximate running time

$$t' \approx t + |\mathcal{R}| + 2\log_2 C \le 2t + 2\log_2 C \le 4t\log_2 C$$

wins the SPrA game with probability $\delta' \approx \delta$. Hence, $\frac{t}{\delta} \ge \frac{t'}{4\log_2 C \cdot \delta'} \ge \frac{S}{4\log_2 C}$. □

Tab. 3 summarizes the results. As we see, a seemingly minor and natural strengthening of PrA leads to much tighter security proofs for time-stamping schemes.

**Table 3.** Tightness parameters of security proofs under different assumptions

| Assumption | Formula | Output Size Formula $n = n(C, S)$ | $n(2^{64}, 2^{80})$ |
|---|---|---|---|
| CR | $\frac{t'}{\delta'} \approx 14\sqrt{C}\left(\frac{t}{\delta}\right)^{1.5}$ | $n = \log_2 N + 3\log_2 S + 8$ | 344 |
| PrA | $\frac{t'}{\delta'} \approx 2C\frac{t}{\delta}$ | $n = 2(\log_2 C + \log_2 S + 1)$ | 290 |
| **SPrA** | $\frac{t'}{\delta'} \approx 4\log_2 C\frac{t}{\delta}$ | $n = 2(\log_2\log_2 C + \log_2 S + 2)$ | **176** |
| RO | $\frac{t}{\delta} \ge 2^{\frac{n-1}{2}}$ | $n = 2\log_2 S + 1$ | 161 |

## 5 Necessity of Boundedness

The security proof in the pure random oracle model does not depend on the capacity $C$ of the time-stamping scheme. In all other cases, the assumption about the boundedness is necessary and the tightness of the reduction depends on $C$. We show in this section that even under assumptions so strong as PrA, the boundedness is a necessary assumption. Namely, under the assumption that PrA hash functions exist, we construct another PrA hash function that is totally insecure for unbounded time-stamping schemes. For every $n \ge 2$ let $\mathcal{I}_n$ be a subset of $\{0,1\}^{2n}$ defined as follows:

$$\mathcal{I}_n = \{(\mathtt{0}x\mathtt{0}, \mathtt{0}x\mathtt{1})\colon x \in \{0,1\}^{n-2},\, x \ne \mathtt{0}^{n-2}\} \ .$$

Define $\iota_n\colon \mathcal{I}_n \to \{0,1\}^{n-1}$ so that $\iota_n(\mathtt{0}x\mathtt{0}, \mathtt{0}x\mathtt{1}) = \mathtt{0}x$ for any $x \in \{0,1\}^{n-2} \backslash \{\mathtt{0}^{n-2}\}$. Clearly, $\iota_n$ is injective and easily invertible. The computations with $\iota_n$ form a tree (with nodes of the form $x = 0x'$, where $x' \in \{0,1\}^{n-1} \backslash \{0^{n-1}\}$). The main property of this tree is that for every $x = 0x'$, where $x' \neq 0^{n-1}$ there is an easily computable hash chain from $x$ to the root node $r = 0^{n-1}1$. For every $F^P\colon \{0,1\}^* \to \{0,1\}^{n-1}$ and for every extractor $\mathcal{E}\colon \{0,1\}^{n-1} \times \{0,1\}^* \to \{0,1\}^*$ define $F_0^P\colon \{0,1\}^* \to \{0,1\}^n$ and $\mathcal{E}_0\colon \{0,1\}^n \times \{0,1\}^* \to \{0,1\}^*$ so that:

$$F_0^P(x) = \begin{cases} \mathtt{0}\|\iota_n(x) & \text{if } x \in \mathcal{I}_n \\ \mathtt{1}\|F^P(x) & \text{if } x \notin \mathcal{I}_n \end{cases} \qquad \mathcal{E}_0(y, \alpha) = \begin{cases} \iota_n^{-1}(y') & \text{if } 0^n \neq y = 0\|y' \in \{0,1\}^n \\ \mathcal{E}(y', \alpha) & \text{if } y = 1\|y' \\ \perp & \text{otherwise.} \end{cases}$$

**Theorem 5.** *$F_0^P$ is insecure for unbounded time-stamping, i.e. there is an efficient strongly unpredictable adversary $(A_1, A_2)$ with success probability $\approx \frac{1}{2} - \frac{1}{2^n}$.*

*Proof.* The first stage $A_1$ outputs the root value $r = 0^{n-1}1$ of the computational tree for $\iota_n$. The second stage choses randomly and uniformly $x \leftarrow \{0,1\}^n$, which with probability $\approx \frac{1}{2} - \frac{1}{2^n}$ belongs to the computational tree and hence there is an easily computable hash chain $x \overset{c}{\rightsquigarrow} r$. □

**Theorem 6.** *If $F^P$ is PrA then so is $F_0^P$.*

*Proof.* Let $\mathcal{E}$ be the extractor for $F^P$ and let $\mathcal{E}_0$ be the corresponding extractor for $F_0^P$. Let $B_0$ be a PrA adversary for $F_0^P$. Let $\mathsf{Ex}_0$ be the oracle that corresponds to $\mathcal{E}_0$. We modify the adversary $B_0$ to an adversary $B$ that uses the $\mathsf{Ex}$-oracle (that corresponds to $\mathcal{E}$) and simulates $B$'s $\mathsf{Ex}_0$-calls with input $y$ as follows:

- if $y = 0\|y' \in \{0,1\}^n \backslash \{0^n\}$, the $\mathsf{Ex}_0$-call is answered with $\iota_n^{-1}(y')$;
- if $y = 1\|y'$, it makes an oracle call $x \leftarrow \mathsf{Ex}(y')$ and answers the call with $x$;
- otherwise, the call is answered with $\perp$.

If $1 \leftarrow \mathbf{Exp}^{\mathrm{pra}}_{F_0,P,\mathcal{E}_0,B_0}$, then during the execution $x \leftarrow B_0^{\mathsf{P},\mathsf{Ex}_0}$ an $\mathsf{Ex}_0$-call was made with an input $y = F_0^P(x)$ such that $\mathcal{E}_0(F_0^P(x), \alpha) \neq x$. Note that $y \neq 0\|y'$, because otherwise (due to the definition of $F_0^P$) we would have $y' \in \mathcal{I}_n$, $y' = \iota_n(x)$, and $\mathcal{E}_0(F_0^P(x), \alpha) = \mathcal{E}_0(0\|y', \alpha) = \iota^{-1}(y') = \iota_n^{-1}(\iota_n(x)) = x$.

Therefore, $y = F_0^P(x) = 1\|y'$, which means that $y' = F^P(x)$ and $\mathcal{E}_0(y, \alpha) = \mathcal{E}(y', \alpha)$. This also means that $B$ makes an $\mathsf{Ex}$-call with parameter $y'$, such that

$$\mathcal{E}(F^P(x), \alpha) = \mathcal{E}(y', \alpha) = \mathcal{E}_0(y, \alpha) = \mathcal{E}_0(F_0^P(x), \alpha) \neq x \ ,$$

and we have $1 \leftarrow \mathbf{Exp}^{\mathrm{pra}}_{F,P,\mathcal{E},B}$. Hence, $\mathbf{Adv}^{\mathrm{pra}}_{F_0,P,\mathcal{E}_0}(B_0) \leq \mathbf{Adv}^{\mathrm{pra}}_{F,P,\mathcal{E}}(B)$. □

**Theorem 7.** *If $F^P$ is SPrA then so is $F_0^P$.*

*Proof.* Let $\mathcal{E}$ be the extractor for $F^P$ and let $\mathcal{E}_0$ be the corresponding extractor for $F_0^P$. Let $\Pi_0$ be an arbitrary $n$-parser and $B_0$ be an SPrA adversary for $F_0^P$. Let $\mathsf{Ext}_0$ be the oracle that corresponds to $\mathcal{E}_0$ and $\Pi_0$. We define an $(n-1)$-parser $\Pi$ in the following way: if $\Pi_0(x) = \{y_1, \ldots, y_m\}$ and $y_1 = b_1\|y_1' \ldots y_m = b_m\|y_m'$

(where $b_1, \ldots, b_m \in \{0,1\}$), then $\Pi(x) = \{y'_1, \ldots, y'_m\}$. Let $\mathsf{Ext}$ be the oracle that corresponds to $\mathcal{E}$ and $\Pi$. We modify the adversary $B_0$ to an adversary $B$ that uses the $\mathsf{Ext}$-oracle (that corresponds to $\mathcal{E}$) and simulates $B$'s $\mathsf{Ext}_0$-calls with input $y$ as follows:

- if $y = 0\|y' \in \{0,1\}^n \backslash \{0^n\}$, then the $\mathsf{Ext}_0$-call is answered with $\iota_n^{-1}(y')$;
- if $y = 1\|y'$, it makes an oracle call $x \leftarrow \mathsf{Ext}(y')$ and answers the call with $x$;
- otherwise, the call is answered with $\perp$.

If $1 \leftarrow \mathbf{Exp}^{\text{spra}}_{F_0,P,\mathcal{E}_0,\Pi_0,B_0}$, then during the execution $x \leftarrow B_0^{\mathsf{P},\mathsf{Ext}_0}$ an $\mathsf{Ext}_0$-call was made with an input $y = F_0^P(x)$ such that $\mathcal{E}_0(F_0^P(x), \alpha) \neq x$. Note that $y \neq 0\|y'$, because otherwise (due to the definition of $F_0^P$) we would have $y' \in \mathcal{I}_n$, $y' = \iota_n(x)$, and $\mathcal{E}_0(F_0^P(x), \alpha) = \mathcal{E}_0(0\|y', \alpha) = \iota^{-1}(y') = \iota_n^{-1}(\iota_n(x)) = x$.

Therefore, $y = F_0^P(x) = 1\|y'$, which means that $y' = F^P(x)$ and $\mathcal{E}_0(y, \alpha) = \mathcal{E}(y', \alpha)$. This also means that $B$ makes an $\mathsf{Ext}$-call with parameter $y'$, such that

$$\mathcal{E}(F^P(x), \alpha) = \mathcal{E}(y', \alpha) = \mathcal{E}_0(y, \alpha) = \mathcal{E}_0(F_0^P(x), \alpha) \neq x \ .$$

Note that due to the definition of the parser $\Pi$, the $\mathsf{Ext}$-oracle uses the same $\alpha$ for extracting $y'$ then the $\mathsf{EXT}_0$-oracle would use for extracting $y = 1\|y'$. Thereby $1 \leftarrow \mathbf{Exp}^{\text{spra}}_{F,P,\mathcal{E},\Pi,B}$, and hence, $\mathbf{Adv}^{\text{pra}}_{F_0,P,\mathcal{E}_0,\Pi_0}(B_0) \leq \mathbf{Adv}^{\text{pra}}_{F,P,\mathcal{E},\Pi}(B)$. □

**Generalization of an Oracle Separation Result from Asiacrypt 2004.** The construction of $F_0^P$ from $F^P$ allows to generalize and simplify the result in [6] which says that there are no black-box reductions that prove the security of unbounded time-stamping scheme based on the collision-freeness of the underlying hash function. Using the construction $F \mapsto F_0$ enables to extend this result form black-box reductions to arbitrary proofs. This is due to the next theorem:

**Theorem 8.** *If $F^P$ is CR then so is $F_0^P$ (even if $F^P$ does not use $P$ at all).*

The proof relies on the fact that the function $F_0^P$ is injective on $\mathcal{I}_n$ and hence finding a collision for $F_0^P$ is equivalent to finding collisions for $F_0$.

*Corollary:* If there exist collision-free hash functions, then there also exist collision free hash functions that are insecure for unbounded hash-then-publish time-stamping schemes. Note that this corollary is a much stronger statement than an oracle separation because it rules out any proving attempts, not only the black-box ones. Therefore, the implication "$F$ is CR $\Rightarrow$ $F$ is secure for unbounded time-stamping" is true only if there exist no collision-free hash functions.

## 6 Open Questions and Further Research

It would be interesting to know whether the SPrA assumption can be used in the indifferentiability framework in a way analogous to the PrA assumption, i.e. is it weak enough for being preserved by the Merkle-Damgård transform, and can the conventional compression functions (like Davies-Meyer) to be proven SPrA.

## References

1. Bayer, D., Haber, S., Stornetta, W.-S.: Improving the efficiency and reliability of digital timestamping. In: Sequences II: Methods in Communication, Security, and Computer Sci., pp. 329–334. Springer, Heidelberg (1993)
2. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: The 1st ACM Conference on Computer and Communications Security: CCS 1993, pp. 62–73. ACM (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. Buldas, A., Niitsoo, M.: Optimally tight security proofs for hash-then-publish time-stamping. In: Steinfeld, R., Hawkes, P. (eds.) ACISP 2010. LNCS, vol. 6168, pp. 318–335. Springer, Heidelberg (2010)
5. Buldas, A., Laur, S.: Do broken hash functions affect the security of time-stamping schemes? In: Zhou, J., Yung, M., Bao, F. (eds.) ACNS 2006. LNCS, vol. 3989, pp. 50–65. Springer, Heidelberg (2006)
6. Buldas, A., Saarepera, M.: On provably secure time-stamping schemes. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 500–514. Springer, Heidelberg (2004)
7. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. JACM 51(4), 557–594 (2004)
8. Chang, D., Yung, M.: Adaptive preimage resistance analysis revisited: requirements, subtleties and implications. In: IACR Cryptology ePrint Archive 209 (2012)
9. Coron, J.-S., Dodis, Y., Malinaud, C., Puniya, P.: Merkle-Damgård revisited: How to construct a hash function. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 430–448. Springer, Heidelberg (2005)
10. Dodis, Y., Ristenpart, T., Shrimpton, T.: Salvaging Merkle-Damgård for practical applications. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 371–388. Springer, Heidelberg (2009)
11. Haber, S., Stornetta, W.-S.: How to time-stamp a digital document. Journal of Cryptology 3(2), 99–111 (1991)
12. Luby, M.: Pseudorandomness and Cryptographic Applications. Princeton University Press, Princeton (1996)
13. Maurer, U., Renner, R., Holenstein, C.: Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 21–39. Springer, Heidelberg (2004)
14. Merkle, R.C.: Protocols for public-key cryptosystems. In: Proceedings of the 1980 IEEE Symposium on Security and Privacy, pp. 122–134 (1980)

# Improved Boomerang Attacks on SM3*

Dongxia Bai[1], Hongbo Yu[1,**], Gaoli Wang[2], and Xiaoyun Wang[3,4,5]

[1] Department of Computer Science and Technology,
Tsinghua University, Beijing 100084, China
baidx10@mails.tsinghua.edu.cn, yuhongbo@mail.tsinghua.edu.cn

[2] School of Computer Science and Technology,
Donghua University, Shanghai 201620, China
wanggaoli@dhu.edu.cn

[3] Institute for Advanced Study, Tsinghua University, Beijing 100084, China

[4] Key Laboratory of Cryptologic Technology and Information Security,
Ministry of Education, Shandong University, Jinan 250100, China

[5] School of Mathematics, Shandong University, Jinan 250100, China
xiaoyunwang@mail.tsinghua.edu.cn

**Abstract.** The cryptographic hash function SM3 is designed by X. Wang *et al.* and published by Chinese Commercial Cryptography Administration Office for the use of electronic certification service system in China. It is based on the Merkle-Damgård design and is very similar to SHA-2 but includes some additional strengthening features. In this paper, we apply the boomerang attack to SM3 compression function, and present such distinguishers on up to 34/35/36/37 steps out of 64 steps, with time complexities $2^{31.4}$, $2^{33.6}$, $2^{73.4}$ and $2^{93}$ compression function calls respectively. Especially, we are able to obtain the examples of the distinguishers on 34-step and 35-step on a PC due to their practical complexities. In addition, incompatible problems in the recent boomerang attack are pointed out.

**Keywords:** SM3, hash function, boomerang attack, cryptanalysis.

## 1 Introduction

Cryptographic hash functions play a significant role in the modern cryptology. The perfect hash function is required to satisfy three certain properties: preimage resistance, 2nd preimage resistance and collision resistance. In recent years, the cryptanalysis of hash functions has become an important topic within the cryptographic community, and the significant advances of hash function research have a formative influence on the field of hash functions. The breakthrough results that many well-known hash functions including MD5 and SHA-1 were broken by

* Supported by 973 program (No. 2013CB834205), the National Natural Science Foundation of China (No. 61133013 and No. 61103238), the Tsinghua University Initiative Scientific Research Program (No. 20111080970), and Tsinghua National Laboratory for Information Science and Technology.

** Corresponding author.

X. Wang *et al.* in 2005 [18,19] convinced lots of cryptographers that these most widely used hash functions can no longer be considered secure. As a consequence, NIST proposed the transition from SHA-1 to SHA-2 family, and many companies and organizations are also migrating to SHA-2. Furthermore, in 2007 NIST started a hash function competition to develop a new hash standard (SHA-3), and the competition ended in 2012 when NIST announced that Keccak would be the new SHA-3 hash algorithm. Meanwhile, people are evaluating these hash functions, not only consider the three classical security requirements, but also regard near-collision, rebound distinguisher, differential distinguisher, boomerang distinguisher, etc. Whenever a hash function behaves differently from the one expected of a random function, it is regarded as the weaknesses of the hash function, whose security is considered to be suspect. Therefore, many attack results in such framework are proposed recently.

Especially, the idea of boomerang attack has been applied to hash functions as part of the new and useful hash function results. A. Biryukov *et al.* [5] and M. Lamberger *et al.* [8] independently applied boomerang attack to hash function and mount distinguishers on compression functions of BLAKE-32 and SHA-2 respectively. In [4] boomerang attack on SHA-2 was extended to 47 steps with practical complexity. In [10], F. Mendel and T. Nad presented boomerang distinguishers for SIMD-512 compression function. Y. Sasaki [12] proposed a boomerang distinguisher for the full compression function of 5-pass HAVAL. Y. Sasaki *et al.* also gave a 2-dimension sums attack on 48-step RIPEMD-128 and 42-step RIPEMD-160 in [13] and boomerang distinguishers for full HAS-160 compression function in [14]. Boomerang distinguishers have also been applied to Skein. In [1], J.-P. Aumasson *et al.* proposed a related-key boomerang distinguisher on 34-step and a known-related-key boomerang distinguisher on 35-step for the core algorithm of Skein (the block cipher Threefish-512). In [9], G. Leurent and A. Roy pointed out the incompatibility between the differential characteristics used in [1] and gave a boomerang distinguisher for 32-step Skein-256. Then H. Yu *et al.* [20] presented the first valid boomerang attacks on 32/33/34-step Skein-512.

SM3 [15] is the Chinese cryptographic hash function standard which is designed by X. Wang *et al.*, and is used in applications such as the electronic certification service system. The design of SM3 builds on the Merkle-Damgård design, and is very similar to the MD4 family of hash functions and in particular to SHA-2, but introducing some additional strengthening features, such as a more complex step function and stronger message dependency than SHA-256. In this work, we study the security of SM3, and present boomerang attacks on step-reduced SM3.

**Related Work.** In the last few years, the amount of cryptanalytic results on SM3 is much lower than other hash function standards. In [21], J. Zou *et al.* present the first preimage attacks on SM3 reduced to 30 steps out of 64 steps with time complexity $2^{249}$, starting from step 6, and 28 steps with time complexity $2^{241.5}$, starting from step 0. At SAC 2012 A. Kircanski *et al.* [7] apply the boomerang attack to SM3 compression function for 32/33/34/35 steps, and give examples of zero-sum quartets for 32-step and 33-step distinguishers. Meanwhile, they also expose a side-rotational property of SM3-XOR function and give a slide-rotational pair for SM3-XOR compression function. Moreover, G. Wang *et al.* [17] propose preimage attacks on SM3 reduced to 29/30 steps, and pseudo-preimage attacks reduced to 31/32 steps, with lower complexities than [21] and all from the first step (step 0), and they also convert those (pseudo) preimage attacks into pseudo-collision attacks on 29/30/31/32-step on SM3 for the first time. Then F. Mendel *et al.* [11] provide the first security analysis of step-reduced SM3 regarding its collision resistance, and they present a collision attack for 20 steps and a free-start collision attack for 24 steps of SM3, both with practical complexity. The above are all the previous results that we are aware of on the analysis of SM3.

**Our Contribution.** In this work, we study the security of the Chinese hash function standard SM3, and show the application of boomerang attack to step-reduced SM3 compression function. Our analysis is based on the two high probability differential characteristics of SM3. Then we use these differential characteristics to build boomerang distinguishers for step-reduced SM3 compression function on up to 34 and 35 steps with practical complexity, and the examples of boomerang quartets are also given. Moreover, the distinguishers can be extended to attacks on 36-step and 37-step SM3. Finally, we show some incompatible problems existed in the differential characteristics used in the previous work [7]. With respect to the attack in the framework of boomerang distinguisher, our results of SM3 are the best as far as we know. The summary of previous results and ours are given in Table 1.

**Outline.** The structure of the paper is as follows. In Section 2, we provide a short description of SM3 hash function. Section 3 briefly overviews the boomerang attack. In Section 4, we present the differential characteristics, and build boomerang distinguishers for step-reduced SM3 compression function on up to 34/35/36/37 steps out of 64 steps. Finally, we conclude our paper in Section 5.

## 2 Description of SM3

SM3 is an iterated hash function that processes 512-bit input message blocks and produces a 256-bit hash value. It basically consists of two parts: the message expansion and the state update transformation. A detailed description of SM3 hash function is given in [15].

**Message Expansion.** The message expansion of SM3 splits the 512-bit message block into 16 words $m_i$ ($0 \leq i \leq 15$), and expands them into 68 expanded

**Table 1.** Summary of the attacks on SM3

| attack type | target | rounds | time | source |
|---|---|---|---|---|
| preimage attack | HF | 28 | $2^{241.5}$ | [21] |
| preimage attack | HF | 30† | $2^{249}$ | |
| preimage attack | HF | 29 | $2^{245}$ | [17] |
| preimage attack | HF | 30 | $2^{251.1}$ | |
| pseudo-preimage attack | HF | 31 | $2^{245}$ | |
| pseudo-preimage attack | HF | 32 | $2^{251.1}$ | |
| pseudo-collision attack | HF | 29 | $2^{122}$ | |
| pseudo-collision attack | HF | 30 | $2^{125.1}$ | |
| pseudo-collision attack | HF | 31 | $2^{122}$ | |
| pseudo-collision attack | HF | 32 | $2^{125.1}$ | |
| collision attack | HF | 20 | practical | [11] |
| free-start collision attack | CF | 24 | practical | |
| boomerang attack | CF | 32 | $2^{14.4}$ | [7] |
| boomerang attack | CF | 33∗ | $2^{32.4}$ | |
| boomerang attack | CF | 34 | $2^{53.1}$ | |
| boomerang attack | CF | 35 | $2^{117.1}$ | |
| boomerang attack | CF | 34 | $2^{31.4}$ | Sect.4 |
| boomerang attack | CF | 35 | $2^{33.6}$ | |
| boomerang attack | CF | 36 | $2^{73.4}$ | |
| boomerang attack | CF | 37 | $2^{93}$ | |

†: the attack starts from step 6;
∗: the attack has some incompatible problems shown later.

message words $w_i$ $(0 \le i \le 67)$ and 64 expanded message words $w'_i$ $(0 \le i \le 63)$ as follows:

$$w_i = \begin{cases} m_i, & 0 \le i \le 15 \\ P_1(w_{i-16} \oplus w_{i-9} \oplus (w_{i-3} \lll 15)) \oplus (w_{i-13} \lll 7) \oplus w_{i-6}, & 16 \le i \le 67 \end{cases}$$
$$w'_i = w_i \oplus w_{i+4}, 0 \le i \le 63.$$

The function $P_1(X)$ is given by

$$P_1(X) = X \oplus (X \lll 15) \oplus (X \lll 23).$$

**State Update Transformation.** The state update transformation starts from an (fixed) initial value $IV = (A_0, B_0, C_0, D_0, E_0, F_0, G_0, H_0)$ of eight 32-bit words and updates them in 64 steps. In each step the two 32-bit words $w_i$ and $w'_i$ are used to update the state variables $A_i, B_i, C_i, D_i, E_i, F_i, G_i, H_i$ as follows:

$$\begin{aligned}
SS1_i &= ((A_i \lll 12) + E_i + (T_i \lll i)) \lll 7,\\
SS2_i &= SS1_i \oplus (A_i \lll 12),\\
TT1_i &= FF_i(A_i, B_i, C_i) + D_i + SS2_i + w'_i,\\
TT2_i &= GG_i(E_i, F_i, G_i) + H_i + SS1_i + w_i,\\
A_{i+1} &= TT1_i,\\
B_{i+1} &= A_i,\\
C_{i+1} &= B_i \lll 9,\\
D_{i+1} &= C_i,\\
E_{i+1} &= P_0(TT2_i),\\
F_{i+1} &= E_i,\\
G_{i+1} &= F_i \lll 19,\\
H_{i+1} &= G_i.
\end{aligned}$$

The step constants are $T_i = 0x79cc4519$ for $i \in \{0, \ldots, 15\}$ and $T_i = 0x7a879d8a$ for $i \in \{16, \ldots, 63\}$. The bitwise Boolean functions $FF(X, Y, Z)$ and $GG(X, Y, Z)$ used in each step are defined as follows:

$$FF_i(X,Y,Z) = \begin{cases} X \oplus Y \oplus Z, & 0 \le i \le 15\\ (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z), & 16 \le i \le 63 \end{cases}$$

$$GG_i(X,Y,Z) = \begin{cases} X \oplus Y \oplus Z, & 0 \le i \le 15\\ (X \wedge Y) \vee (\neg X \wedge Z), & 16 \le i \le 63. \end{cases}$$

The linear function $P_0(X)$ is defined as follows:

$$P_0(X) = X \oplus (X \lll 9) \oplus (X \lll 17).$$

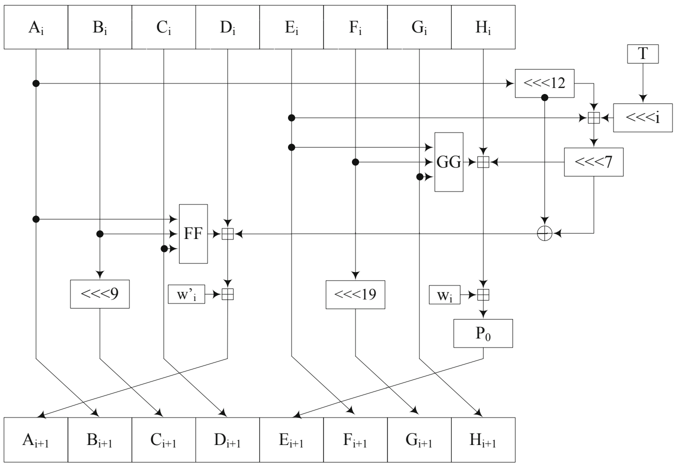


**Fig. 1.** The $i$-th step function of SM3

After the last step of the state update transformation, the initial values are added to the output values of the last step. The result is the final hash value or the input of the next message block. One step of the SM3 compression function is illustrated in Fig. 1.

## 3 The Boomerang Attack

The boomerang attack was introduced by D. Wagner in 1999 [16] as a tool for the cryptanalysis of block cipher. It is an adaptive chosen plaintext and ciphertext attack utilizing differential cryptanalysis. The cipher is treated as a cascade of two sub-ciphers, where a short differential is used in each of these sub-ciphers. These differentials are combined to exploit an adaptive chosen plaintext and ciphertext property of the cipher that has high probability. Then J. Kelsey *et al.* [6] further developed it into a chosen plaintext attack called the amplified boomerang attack, and later it was developed by E. Biham *et al.* [3] into the rectangle attack. Then E. Biham *et al.* [2] combined the boomerang (and the rectangle) attack with related-key differentials and proposed the related-key boomerang and rectangle attacks, which use the related-key differentials instead of the single-key differentials.

We mainly review the known-related-key boomerang attack [4] which can be used to distinguish a given permutation from a random oracle. Applying the known-related-key boomerang attack to the compression function in the MMO mode, i.e, $CF(M, K) = E(M, K) + M$, it is possible to start from the middle steps since the message $M$ and the key $K$ can be chosen randomly (refer to [4,9]). Then we have a backward differential characteristic $(\beta, \beta_k) \rightarrow \alpha$ with probability $p$ for $CF_0^{-1}$, and another forward differential characteristic $(\gamma, \gamma_k) \rightarrow \delta$ with probability $q$ for $CF_1$. Next the known-related-key boomerang attack can be constructed using these two differentials as follows:

- Choose a random value $X_1$ and $K_1$, compute $X_2 = X_1 \oplus \beta$, $X_3 = X_1 \oplus \gamma$, $X_4 = X_3 \oplus \beta$, and $K_2 = K_1 \oplus \beta_k$, $K_3 = K_1 \oplus \gamma_k$, $K_4 = K_3 \oplus \beta_k$.
- Compute backward from $(X_i, K_i)$ using $CF_0^{-1}$ to obtain $P_i$ (i=1,2,3,4).
- Compute forward from $(X_i, K_i)$ using $CF_1$ to obtain $C_i$ (i=1,2,3,4).
- Check whether $P_1 \oplus P_2 = P_3 \oplus P_4 = \alpha$ and $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$.

We can deduce that $P_1 \oplus P_2 = P_3 \oplus P_4 = \alpha$ and $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$ hold with probability at least $p^2$ in the backward direction and $q^2$ in the forward direction. Hence, the attack succeeds with probability $p^2q^2$ when assuming that the differentials are independent.

## 4 The Boomerang Attacks on SM3

In this section, we present the boomerang attacks on the SM3 compression function reduced to 34 and 35 steps with practical examples of boomerang quartets, and then extend the attacks to 36 and 37 steps. Firstly, we have to find the differential characteristics used in the attack to distinguish the target compression function from random functions. Secondly, we derive the sufficient conditions in the intermediate steps, and correct these conditions by using message modification technique. Finally, we evaluate the complexities of our attacks and search right quartet examples.

## 4.1 Step-Reduced Differential Trails

We give the two differential characteristics which are used to attack 34-step SM3 compression function and find boomerang distinguishers, where the top differential characteristic is from step 15 to 0, and the bottom one is from step 16 to 33. Note that we all use the XOR difference $\Delta a = a \oplus a'$, and let $\Delta a$: $i$ for $1 \leq i \leq 32$ denotes that the $i$-th bit of $a$ is different from the $i$-th bit of $a'$, and all the other bits of $a$ and $a'$ are the same.

We start from the middle states of the distinguisher quartet ($V_1$, $V_2$, $V_3$, $V_4$), and for the top differential characteristic, the differences of the message word $w_i$ and the chaining values $A_{16}$ to $H_{16}$ are chosen as follows:

– $\Delta w_2 : 32$ (the MSB difference), $\Delta w_i = 0$ ($0 \leq i \leq 15$, $i \neq 2$), if we choose the message word with this difference, we will find that 13 steps (step 1 to 13) are passed for free according to the message expansion of SM3. This is significant for us to get the high probability differential characteristic.

– $\Delta A_{16} : 2, 3, 10, 12, 15, 19, 23, 27, 32$, $\Delta B_{16} : 15, 23, 32$, $\Delta E_{16} : 2, 4, 10, 11, 19, 27, 28$, these differences are decided by the difference of the message word $\Delta w_2$ above. We can easily get the differences of the message words $\Delta w_0$–$\Delta w_{15}$, $\Delta w'_0$–$\Delta w'_{15}$ in the top differential characteristic from above: $\Delta w_2 : 32$, $\Delta w'_2 : 32$, $\Delta w'_{14} : 15, 23, 32$, and all the other ones are zero. Then we directly derive the differences of the chaining values of step 14 and 15 with some sufficient conditions.

For the bottom differential characteristic, we select the differences as follows:

– $\Delta w_{20} : 20$ (the 20-th bit difference), $\Delta w_i = 0$ ($21 \leq i \leq 35$), so we can pass 11 steps (step 21 to 31) for free similarly.

– $\Delta C_{16} : 9, 16, 18, 23, 25, 26, 30, 31$, $\Delta D_{16} : 11, 20$, $\Delta G_{16} : 9, 16, 18, 24, 25, 26, 30, 32$, $\Delta H_{16} : 1, 3, 4, 10, 12, 19, 20, 28$, according to the differences of the message words above, and also considering the compatibility with the top differential characteristic in the middle steps which cannot contain any contradiction, the differences of chaining values in bottom differential characteristic are derived with some sufficient conditions. For example, to cancel the 9-th and 10-th bit differences of $w'_{17}$, we choose the difference in $D_{17}$ only on bit 9 but not on bits 9 and 10, because if we has difference in $D_{17}$ on bit 10, then in step 16 the condition $A_{16,10} = B_{16,10}$ (note that $C_{16} = D_{17}$) cannot be satisfied in the other side ($V_2, V_4$).

In Table 2 and Table 3 the differential characteristics for both forward and backward directions are shown. Furthermore, the conditions and probabilities for each step of the differential characteristics are given.

**Table 2.** Differential characteristic for steps 0-15 using XOR differences (34 steps)

| $i$ | chaining value | message | conditions | prob |
|---|---|---|---|---|
| 0 | $B_0 : 23$<br>$C_0 : 32$<br>$D_0 : 23, 32$<br>$F_0 : 13$<br>$G_0 : 32$<br>$H_0 : 13, 32$ | | $(A_0 \oplus B_0 \oplus C_0)_{23} \neq D_{0,23}$,<br>$(E_0 \oplus F_0 \oplus G_0)_{13} \neq H_{0,13}$ | $2^{-2}$ |
| 1 | $C_1 : 32$<br>$D_1 : 32$<br>$G_1 : 32$<br>$H_1 : 32$ | | | 1 |
| 2 | $D_2 : 32$<br>$H_2 : 32$ | $w_2 : 32$<br>$w'_2 : 32$ | | 1 |
| 3 | | | | 1 |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| 14 | | $w'_{14} : 15, 23, 32$ | $TT1_{14,i} = w'_{14,i} (i = 15, 23)$ | $2^{-2}$ |
| 15 | $A_{15} : 15, 23, 32$ | | $SS1_{15,(2,10,19)} = A_{15,(15,23,32)}$,<br>$TT1_{15,i} = (A_{15} \oplus B_{15} \oplus C_{15})_i$<br>$(i = 15, 23)$,<br>$TT1_{15,i} = SS2_{15,i}$<br>$(i = 2, 3, 10, 12, 19, 27)$,<br>$TT2_{15,i} = SS1_{15,i} (i = 2, 10, 19)$ | $2^{-14}$ |
| 16 | $A_{16} : 2, 3, 10, 12, 15, 19, 23, 27, 32$<br>$B_{16} : 15, 23, 32$<br>$E_{16} : 2, 4, 10, 11, 19, 27, 28$ | | | — |

**Table 3.** Differential characteristic for steps 16-33 using XOR differences (34 steps)

| $i$ | chaining value | message | conditions | prob |
|---|---|---|---|---|
| 16 | $C_{16}: 9, 16, 18, 23, 25, 26, 30, 31$ <br> $D_{16}: 11, 20$ <br> $G_{16}: 9, 16, 18, 24, 25, 26, 30, 32$ <br> $H_{16}: 1, 3, 4, 10, 12, 19, 20, 28$ | $w'_{16}: 20$ | $A_{16,i} = B_{16,i}$ $(i = 9, 16, 18, 23, 25, 26, 30, 31)$, <br> $D_{16,20} \neq w'_{16,20}$, <br> $TT1_{16,11} = D_{16,11}$, <br> $E_{16,i} = 1 (i = 9, 16, 24, 25, 26, 30, 32)$, <br> $E_{16,18} = 0$, <br> $TT2_{16,i} = H_{16,i}$ $(i = 1, 3, 4, 10, 12, 19, 20, 28)$, <br> $TT2_{16,18} = G_{16,18}$ | $2^{-27}$ |
| 17 | $A_{17}: 11$ <br> $D_{17}: 9, 16, 18, 23, 25, 26, 30, 31$ <br> $E_{17}: 1$ <br> $H_{17}: 9, 16, 18, 24, 25, 26, 30, 32$ | $w_{17}: 8, 9, 10, 16, 18, 24, 25, 27, 32$ <br> $w'_{17}: 8, 9, 10, 16, 18, 24, 25, 27, 32$ | $SS1_{17,30} = A_{17,11}$, $SS1_{17,8} = E_{17,1}$, <br> $B_{17,11} = C_{17,11}, w'_{17,8} \neq SS2_{17,8}$, <br> $D_{17,9} = w'_{17,9} \neq w'_{17,10}$, <br> $D_{17,i} \neq w'_{17,i} (i = 16, 18)$, <br> $D_{17,23} = SS2_{17,23} \neq w'_{17,24}$, <br> $D_{17,25} = w'_{17,25} = D_{17,26} \neq w'_{17,27}$, <br> $D_{17,30} = SS2_{17,30} = D_{17,31}$, <br> $F_{17,1} = G_{17,1}, w_{17,8} \neq SS1_{17,8}$, <br> $H_{17,9} = w_{17,9} \neq w_{17,10}$, <br> $H_{17,i} \neq w_{17,i} (i = 16, 18, 24)$, <br> $H_{17,25} = w_{17,25} = H_{17,26} \neq w_{17,27}$, <br> $H_{17,30} \neq SS1_{17,30}$ | $2^{-26}$ |
| 18 | $B_{18}: 11$ <br> $F_{18}: 1$ | | $A_{18,11} = C_{18,11}$, <br> $E_{18,1} = 0$ | $2^{-2}$ |
| 19 | $C_{19}: 20$ <br> $G_{19}: 20$ | | $A_{19,20} = B_{19,20}$, <br> $E_{19,20} = 1$ | $2^{-2}$ |
| 20 | $D_{20}: 20$ <br> $H_{20}: 20$ | $w_{20}: 20$ <br> $w'_{20}: 20$ | $D_{20,20} \neq w'_{20,20}$, <br> $H_{20,20} \neq w_{20,20}$ | $2^{-2}$ |
| 21 | | | | 1 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| 32 | | $w'_{32}: 3, 11, 20$ | $TT1_{32,i} = w'_{32,i} (i = 3, 11, 20)$ | $2^{-3}$ |
| 33 | $A_{33}: 3, 11, 20$ | | $SS1_{33,(22,30,7)} = A_{33,(3,11,20)}$, <br> $B_{33,i} = C_{33,i} (i = 3, 11, 20)$, <br> $TT1_{33,i} = SS2_{33,i}$ $(i = 7, 15, 22, 23, 30)$, <br> $TT2_{33,i} = SS1_{33,i} (i = 7, 22, 30)$ | $2^{-14}$ |
| 34 | $A_{34}: 7, 15, 22, 23, 30, 32$ <br> $B_{34}: 3, 11, 20$ <br> $E_{34}: 7, 15, 16, 22, 24, 30, 31$ | | | — |

### 4.2 Message Modification for the Middle Steps

Here we use the message modification technique to modify the chaining values and message words to satisfy the conditions of the middle steps to improve the complexity of our attack.

In the top differential characteristic, there are 16 sufficient conditions from step 15 to 14, which can be satisfied both in two sides $(V_1, V_2)$ and $(V_3, V_4)$ by modifying $A_{16}$, $B_{16}$ and $F_{16}$. Therefore, the conditions of this part (step 15 to 14) can hold with probability 1.

Similarly, 59 conditions in total from step 16 to 20 in the bottom differential characteristic need to be corrected in each side. We can make all these conditions hold in one side $(V_1, V_3)$ by the message modification. Furthermore, part of the conditions in the other side $(V_2, V_4)$ can be corrected, and 14 conditions including $SS1_{17,30} = A_{17,11}$, $SS1_{17,8} = E_{17,1}$, $w'_{17,8} \neq SS2_{17,8}$, $D_{17,23} = SS2_{17,23} \neq w'_{17,24}$, $D_{17,30} = SS2_{17,30} = D_{17,31}$, $w_{17,8} \neq SS1_{17,8}$, $H_{17,30} \neq SS1_{17,30}$, $A_{18,11} = C_{18,11}$, $E_{18,1} = 0$, $A_{19,20} = B_{19,20}$, $E_{19,20} = 1$, and $H_{20,20} \neq w_{20,20}$ are not corrected. As a result, all the conditions of step 16 to 20 in the bottom differential characteristic hold with probability at least $2^{-14}$, rather than the much lower average probability $2^{-2\times 59} = 2^{-118}$.

### 4.3 Complexity of the Attack

After the message modification, the boomerang distinguisher in the middle steps (14 to 20) hold with a much higher probability $2^{-14}$. Meanwhile, the probability of step 0 to 13 in the top differential characteristic is $2^{-2}$, and for step 21 to 33 in the bottom differential characteristic is $2^{-17}$. So the complexity of the 34-step boomerang distinguisher is $2^{14} + 2^{2\times(2+17)} \approx 2^{38}$. Note that it can be further reduced to $2^{14} + 2^{2\times 3} \times 3^{2+14} \approx 2^{14} + 2^{31.4} \approx 2^{31.4}$ if we only obtain a zero-sum distinguisher.

Due to the low complexity, our distinguisher on up to 34-step compression function of SM3 is practical, and we are able to find boomerang quartets on a PC quickly. We give an example of 34-step boomerang distinguisher in Table 4.

### 4.4 Attacks on 35/36/37-Step SM3 Compression Function

**35-step attack(steps 0-34).** Using the same top differential characteristic shown in Table 2, we add one more step as the new 16-th step in the bottom differential characteristic as illustrated in Table 5 to mount a 35-step attack. So the step where the single bit difference in the message word $w_i$ in the bottom differential characteristic has been set should slip to step 21. Now we look at the choice of differences in bottom differential characteristic, if we still use the same bit difference on bit 20 in $w_{21}$, some contradictions will emerge, and through theoretical derivation and program tests we find that only the 24-th bit difference in $w_{21}$ is applicable and compatible between the two differential characteristics. We only correct all conditions in the side $(V_1, V_2)$ of the top

**Table 4.** Example of a boomerang quartet for 34-step CF of SM3. $P_i$, $C_i$ and $M_i$ respectively denote the chaining values of step 0, 33 and message words.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $P_1$ | 8e328bf1 | 540ba9e5 | 026995ca | d1271808 | 8afc4d19 | 95bddaa7 | a56d9207 | a2c44d1c |
| $P_2$ | 8e328bf1 | 544ba9e5 | 826995ca | 51671808 | 8afc4d19 | 95bdcaa7 | 256d9207 | 22c43d1c |
| $P_3$ | 11ee1c76 | ee57de46 | 54838689 | 0665bf71 | df61a977 | 5f4c46e9 | d42981b4 | c15ec4f8 |
| $P_4$ | 11ee1c76 | ee17de46 | d4838689 | 8625bf71 | df61a977 | 5f4c56e9 | 542981b4 | 415eb4f8 |
| $M_1$ | d7a6bd34 | 66fa6efa | 78ce08a1 | 9a585055 | 94c8bc0b | 3b679ebd | 3910da41 | f0e82d8a |
| | d5f41b80 | 64f0041d | 947bccb4 | 4344d2ed | bcc94a67 | 6b5f97ff | 79000306 | 16233872 |
| $M_2$ | d7a6bd34 | 66fa6efa | f8ce08a1 | 9a585055 | 94c8bc0b | 3b679ebd | 3910da41 | f0e82d8a |
| | d5f41b80 | 64f0041d | 947bccb4 | 4344d2ed | bcc94a67 | 6b5f97ff | 79000306 | 16233872 |
| $M_3$ | d7acbd36 | 26fbaefa | 50ce00a1 | 1fdad3d5 | 94c2b90e | 333fb685 | 3918da41 | 70e8ad8c |
| | d5f49b80 | 64f0041d | 9570c9b3 | c3c4d3ed | bcc94a67 | 6b5797ff | f9008304 | 16233872 |
| $M_4$ | d7acbd36 | 26fbaefa | d0ce00a1 | 1fdad3d5 | 94c2b90e | 333fb685 | 3918da41 | 70e8ad8c |
| | d5f49b80 | 64f0041d | 9570c9b3 | c3c4d3ed | bcc94a67 | 6b5797ff | f9008304 | 16233872 |
| $C_1$ | 5cc18f78 | adf682b8 | 837bc39c | 1550ef7d | 5e6d092c | b95a7f10 | 0fdde16d | 3dc6bf65 |
| $C_2$ | 35437883 | a37697ca | 94fa71b5 | 169e842d | 07d1f375 | e5e58686 | e97b5e86 | 72b07d54 |
| $C_3$ | bcd1cfbd | adee86bc | 837bc39c | 1550ef7d | fecc48ec | b95a7f10 | 0fdde16d | 3dc6bf65 |
| $C_4$ | d5533846 | a36e93ce | 94fa71b5 | 169e842d | a770b2b5 | e5e58686 | e97b5e86 | 72b07d54 |

differential characteristic and the side $(V_1, V_3)$ of the bottom differential characteristic, and part of conditions (12 conditions) in the other side $(V_3, V_4)$ of the top differential characteristic. The remaining conditions in middle steps have not been dealt with. So in theory the boomerang distinguisher in the middle steps (14 to 21) holds with probability $2^{-46}$. However, according to our experiments, on average, only about 32 conditions in the middle steps have not been corrected. As a result, the complexity of 35-step boomerang distinguisher is $2^{32} + 2^{2\times 3} \times 3^{2+15} \approx 2^{32} + 2^{33} \approx 2^{33.6}$, and the practical example of 35-step boomerang distinguisher quartet can be found on a PC, see Table 6.

**36-step attack(steps 0-35).** The 36-step attack is obtained with the same differential characteristics as 35-step attack by adding one step in the top differential characteristic as the new first step, where the top differential characteristic is from step 0 to 16 and the bottom differential characteristic is from step 17 to 35. In order to keep the conditions of connection part between the top and bottom differential characteristics mostly unchanged, we change the differences of the top differential characteristic slightly: $\Delta w_0 : 4, 5, 7, 12, 20, 21, 22, 28, 30$, $\Delta w_3 : 32$, $\Delta w_i = 0$ $(0 \leq i \leq 15, i \neq 0, 3)$, $\Delta A_{17} : 2, 3, 10, 12, 19, 27$, $\Delta B_{17} : 15, 23, 32$, $\Delta E_{17} : 2, 4, 10, 11, 19, 27, 28$, see Table 7. The complexity of the 36-step attack is $2^{32} + 2^{2\times(2+3)} \times 3^{25+15} \approx 2^{32} + 2^{73.4} \approx 2^{73.4}$.

**37-step attack(steps 0-36).** Extending the 36-step boomerang distinguisher for one more step at the end of the top differential characteristic, by using the message modification technique, we get a 37-step attack with a complexity of $2^{93} + 2^{2\times(2+3)} \times 3^{25+15} \approx 2^{93} + 2^{73.4} \approx 2^{93}$.

**Table 5.** Differential characteristic for steps 16-34 using XOR differences (35 steps)

| $i$ | chaining value | message | conditions | prob |
|---|---|---|---|---|
| 16 | $B_{16}: 4, 5, 11, 13, 18, 20, 21, 25, 26$<br>$C_{16}: 15, 22, 23$<br>$F_{16}: 1, 3, 9, 12, 15, 17, 26$<br>$G_{16}: 5, 7, 8, 14, 16, 23, 24, 32$ | | $A_{16,i} = B_{16,i} (i = 15, 22, 23)$,<br>$A_{16,i} = C_{16,i}$<br>$(i = 4, 5, 11, 13, 18, 20, 21, 25, 26)$,<br>$E_{16,i} = 0$<br>$(i = 1, 3, 9, 12, 15, 17, 26)$,<br>$E_{16,i} = 1$<br>$(i = 5, 7, 8, 14, 16, 23, 24, 32)$ | $2^{-27}$ |
| 17 | $C_{17}: 2, 3, 13, 14, 20, 22, 27, 29, 30$<br>$D_{17}: 15, 22, 23$<br>$G_{17}: 2, 4, 13, 20, 22, 28, 31$<br>$H_{17}: 5, 7, 8, 14, 16, 23, 24, 32$ | $w'_{17}: 24$ | $A_{17,i} = B_{17,i}$<br>$(i = 2, 3, 13, 14, 20, 27, 29, 30)$,<br>$A_{17,22} \neq B_{17,22}, TT1_{17,15} = D_{17,15}$,<br>$C_{17,22} = D_{17,22} = D_{17,23} \neq w'_{17,24}$,<br>$E_{17,i} = 1 (i = 2, 4, 13, 20, 28, 31)$,<br>$E_{17,22} = 0$,<br>$TT2_{17,i} = H_{17,i}$<br>$(i = 5, 7, 8, 14, 16, 23, 24)$,<br>$TT2_{17,22} = G_{17,22}$ | $2^{-28}$ |
| 18 | $A_{18}: 15$<br>$D_{18}: 2, 3, 13, 14, 20, 22, 27, 29, 30$<br>$E_{18}: 5$<br>$H_{18}: 2, 4, 13, 20, 22, 28, 31$ | $w_{18}: 4, 12, 13, 14, 20, 22, 28, 29, 31$<br>$w'_{18}: 4, 12, 13, 14, 20, 22, 28, 29, 31$ | $SS1_{18,2} = A_{18,15}$, $SS1_{18,12} = E_{18,5}$,<br>$D_{18,2} = SS2_{18,2} = D_{18,3} \neq w'_{18,4}$,<br>$SS2_{18,12} \neq w'_{18,12}$,<br>$D_{18,i} \neq w'_{18,i} (i = 13, 14, 20, 22)$,<br>$D_{18,27} = SS2_{18,27} \neq w'_{18,28}$,<br>$D_{18,29} = w'_{18,29} = D_{18,30} \neq w'_{18,31}$,<br>$B_{18,15} = C_{18,15}, SS1_{18,12} = w_{18,12}$,<br>$H_{18,i} \neq w_{18,i} (i = 4, 20, 22, 31)$,<br>$H_{18,i} = w_{18,i} \neq w_{18,i+1} (i = 13, 28)$,<br>$H_{18,2} \neq SS1_{18,2}, F_{18,5} = G_{18,5}$ | $2^{-27}$ |
| 19 | $B_{19}: 15$<br>$F_{19}: 5$ | | $A_{19,15} = C_{19,15}$,<br>$E_{19,5} = 0$ | $2^{-2}$ |
| 20 | $C_{20}: 24$<br>$G_{20}: 24$ | | $A_{20,24} = B_{20,24}$,<br>$E_{20,24} = 1$ | $2^{-2}$ |
| 21 | $D_{21}: 24$<br>$H_{21}: 24$ | $w_{21}: 24$<br>$w'_{21}: 24$ | $D_{21,24} \neq w'_{21,24}$,<br>$H_{21,24} \neq w_{21,24}$ | $2^{-2}$ |
| 22 | | | | 1 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| 33 | | $w'_{33}: 7, 15, 24$ | $TT1_{33,i} = w'_{33,i} (i = 7, 15, 24)$ | $2^{-3}$ |
| 34 | $A_{34}: 7, 15, 24$ | | $SS1_{34,(26,2,11)} = A_{34,(7,15,24)}$,<br>$B_{34,i} = C_{34,i} (i = 7, 15, 24)$,<br>$TT1_{34,i} = SS2_{34,i}$<br>$(i = 2, 4, 11, 19, 26, 27)$,<br>$TT2_{34,i} = SS1_{34,i} (i = 2, 11, 26)$ | $2^{-15}$ |
| 35 | $A_{35}: 2, 10, 15, 19, 23, 32$<br>$B_{35}: 15, 23, 32$<br>$E_{35}: 2, 4, 10, 11, 19, 27, 28$ | | | — |

**Table 6.** Example of a boomerang quartet for 35-step CF of SM3

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $P_1$ | 7f57e38d | 801906df | caf2cf8c | 42c58fba | 9feec59b | ef5ab3fc | d261869c | 892ca15c |
| $P_2$ | 7f57e38d | 805906df | 4af2cf8c | c2858fba | 9feec59b | ef5aa3fc | 5261869c | 092cb15c |
| $P_3$ | 0188f80d | 5d3b7666 | 9f941688 | fc411326 | 3a674355 | 2c6075fb | 85a38600 | 892e081b |
| $P_4$ | 0188f80d | 5d7b7666 | 1f941688 | 7c011326 | 3a674355 | 2c6065fb | 05a38600 | 092e181b |
| $M_1$ | f5bc88b9 | af543ad9 | f5068596 | beaebbf0 | 9984c067 | ed6e551a | 7973166d | cef6b36f |
| | c6978096 | fdba14b7 | 2872ffba | 2cf314e6 | 750499b3 | 4ceb9f22 | bd2d99db | 71cc928b |
| $M_2$ | f5bc88b9 | af543ad9 | 75068596 | beaebbf0 | 9984c067 | ed6e551a | 7973166d | cef6b36f |
| | c6978096 | fdba14b7 | 2872ffba | 2cf314e6 | 750499b3 | 4ceb9f22 | bd2d99db | 71cc928b |
| $M_3$ | 75bc89b9 | aff43af9 | f51a8592 | 3eae3bf2 | c1acf86f | edce054a | fcf195ed | ce76b36f |
| | c69f80fe | fdb214b7 | 2872ffba | 3c434496 | 7d0489bb | 4ceb9f22 | bdad99db | 71c492a3 |
| $M_4$ | 75bc89b9 | aff43af9 | 751a8592 | 3eae3bf2 | c1acf86f | edce054a | fcf195ed | ce76b36f |
| | c69f80fe | fdb214b7 | 2872ffba | 3c434496 | 7d0489bb | 4ceb9f22 | bdad99db | 71c492a3 |
| $C_1$ | ecda4c19 | 39e58fb5 | 8fbc81e3 | 75eec099 | 655e3f8b | f4273d52 | 94532c77 | 6967f472 |
| $C_2$ | 93ffb93f | e7e2ffb3 | 447c0e9f | b8ff8f6c | 37a12b0a | ca38d92c | 7eb36c56 | 899e0baf |
| $C_3$ | f2de485f | 3965cff5 | 8fbc81e3 | 75eec099 | 6f523b8d | f4273d52 | 94532c77 | 6967f472 |
| $C_4$ | 8dfbbd79 | e762bff3 | 447c0e9f | b8ff8f6c | 3dad2f0c | ca38d92c | 7eb36c56 | 899e0baf |

## 4.5 The Incompatibility in Previous Boomerang Attacks on SM3

In [7], boomerang distinguishers for SM3 compression function reduced to 33 steps and the corresponding example of zero-sum quartet for 33 steps are given. However, we find that the proposed example of quartet is not consistent with the differential characteristics shown in table 4 and 5 in that paper. According to the differences of the given example, it is supposed to be generated from the another boomerang distinguisher which is the same with their 32-step attack by adding one step at the end of the bottom differential characteristic with the same single bit difference on bit 15 in message word $w_{19}$. Then we study the given boomerang distinguisher in [7] and find some contradictions between the two differential characteristics.

For the differences in step 20 in the bottom differential characteristic, it is easy to deduce that $D_{20,28} = C_{19,28} = B_{18,19} = A_{17,19} = TT1_{16,19} = D_{16,19}$, so the condition $D_{20,28} \neq w'_{20,28}$ in step 20 can be rewritten as $D_{16,19} \neq w'_{20,28}$. From the top differential characteristic, we get that $\Delta D_{16} = 0$ (so $\Delta D_{16,19} = 0$), $\Delta w'_{20,28} = 1$ (according to the message expansion), so the condition $D_{20,28} \neq w'_{20,28}$ in step 20 cannot be satisfied in the other side $(V_2, V_4)$ for the bottom differential characteristic. We can correct the bottom differential characteristic by simply changing the single bit difference of message word $w_{20}$ from bit 28 to 20.

**Table 7.** Differential characteristic for steps 0-16 using XOR differences (36 steps)

| $i$ | chaining value | message | conditions | prob |
|---|---|---|---|---|
| 0 | $A_0 : 23$ <br> $B_0 : 23$ <br> $C_0 : 23, 32$ <br> $D_0 : 3, 4, 5, 7, 10, 12, 21, 22, 23, 28, 30, 32$ <br> $E_0 : 13$ <br> $F_0 : 13$ <br> $G_0 : 13, 32$ <br> $H_0 : 4, 5, 7, 10, 12, 13, 21, 22, 28, 30, 32$ | $w_0 : 4, 5, 7, 12, 20, 21, 22, 28, 30$ <br> $w'_0 : 4, 5, 7, 12, 20, 21, 22, 28, 30$ | $SS1_{0,10} = A_{0,23}$, <br> $SS1_{0,20} = E_{0,13}$, <br> $D_{0,23} \neq (A_0 \oplus B_0 \oplus C_0)_{23}$, <br> $D_{0,i} \neq SS2_{0,i} (i = 3, 10)$, <br> $D_{0,i} \neq w'_{0,i}$ <br> $(i = 4, 5, 7, 12, 21, 22, 28, 30)$, <br> $SS2_{0,20} \neq w'_{0,20}$, <br> $H_{0,13} \neq (E_0 \oplus F_0 \oplus G_0)_{13}$, <br> $H_{0,i} \neq w_{0,i}$ <br> $(i = 4, 5, 7, 12, 21, 22, 28, 30)$, <br> $H_{0,10} \neq SS1_{0,10}$, <br> $SS1_{0,20} \neq w_{0,20}$ | $2^{-25}$ |
| 1 | $B_1 : 23$ <br> $C_1 : 32$ <br> $D_1 : 23, 32$ <br> $F_1 : 13$ <br> $G_1 : 32$ <br> $H_1 : 13, 32$ | | $(A_1 \oplus B_1 \oplus C_1)_{23} \neq D_{1,23}$, <br> $(E_1 \oplus F_1 \oplus G_1)_{13} \neq H_{1,13}$ | $2^{-2}$ |
| 2 | $C_2 : 32$ <br> $D_2 : 32$ <br> $G_2 : 32$ <br> $H_2 : 32$ | | | 1 |
| 3 | $D_3 : 32$ <br> $H_3 : 32$ | $w_3 : 32$ <br> $w'_3 : 32$ | | 1 |
| 4 | | | | 1 |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| 15 | | $w'_{15} : 15, 23, 32$ | $TT1_{15,i} = w'_{15,i} (i = 15, 23)$ | $2^{-2}$ |
| 16 | $A_{16} : 15, 23, 32$ | | $SS1_{16,(2,10,19)} = A_{16,(15,23,32)}$, <br> $B_{15,i} = C_{15,i} (i = 15, 23, 32)$, <br> $TT1_{15,i} = SS2_{15,i}$ <br> $(i = 2, 3, 10, 12, 19, 27)$, <br> $TT2_{16,i} = SS1_{16,i} (i = 2, 10, 19)$ | $2^{-15}$ |
| 17 | $A_{17} : 2, 3, 10, 12, 19, 27$ <br> $B_{17} : 15, 23, 32$ <br> $E_{17} : 2, 4, 10, 11, 19, 27, 28$ | | | — |

## 5 Conclusion

In this paper, we present step-reduced differential characteristics of SM3 hash function with high probabilities and construct boomerang distinguishers for the compression function of SM3 reduced to 34/35/36/37 steps out of 64 steps, and give the examples of quartet of attacks on up to 34-step and 35-step. These are the best attack results to date. Our attacks do not contradict the security claims of SM3.

**Acknowledgment.** We are grateful to the anonymous reviewers for their valuable comments on this paper.

## References

1. Aumasson, J.-P., Çalık, Ç., Meier, W., Özen, O., Phan, R.C.-W., Varıcı, K.: Improved Cryptanalysis of Skein. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 542–559. Springer, Heidelberg (2009)
2. Biham, E., Dunkelman, O., Keller, N.: Related-Key Boomerang and Rectangle Attacks. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 507–525. Springer, Heidelberg (2005)
3. Biham, E., Dunkelman, O., Keller, N.: The Rectangle Attack - Rectangling the Serpent. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 340–357. Springer, Heidelberg (2001)
4. Biryukov, A., Lamberger, M., Mendel, F., Nikolić, I.: Second-Order Differential Collisions for Reduced SHA-256. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 270–287. Springer, Heidelberg (2011)
5. Biryukov, A., Nikolić, I., Roy, A.: Boomerang Attacks on BLAKE-32. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 218–237. Springer, Heidelberg (2011)
6. Kelsey, J., Kohno, T., Schneier, B.: Amplified Boomerang Attacks Against Reduced-Round MARS and Serpent. In: Schneier, B. (ed.) FSE 2000. LNCS, vol. 1978, pp. 75–93. Springer, Heidelberg (2001)
7. Kircanski, A., Shen, Y., Wang, G., Youssef, A.M.: Boomerang and Slide-Rotational Analysis of the SM3 Hash Function. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 304–320. Springer, Heidelberg (2013)
8. Lamberger, M., Mendel, F.: Higher-Order Differential Attack on Reduced SHA-256, `http://eprint.iacr.org/2011/037.pdf`
9. Leurent, G., Roy, A.: Boomerang Attacks on Hash Function Using Auxiliary Differentials. In: Dunkelman, O. (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 215–230. Springer, Heidelberg (2012)
10. Mendel, F., Nad, T.: Boomerang Distinguisher for the SIMD-512 Compression Function. In: Bernstein, D.J., Chatterjee, S. (eds.) INDOCRYPT 2011. LNCS, vol. 7107, pp. 255–269. Springer, Heidelberg (2011)
11. Mendel, F., Nad, T., Schläffer, M.: Finding Collisions for Round-Reduced SM3. In: Dawson, E. (ed.) CT-RSA 2013. LNCS, vol. 7779, pp. 174–188. Springer, Heidelberg (2013)
12. Sasaki, Y.: Boomerang Distinguishers on MD4-Family: First Practical Results on Full 5-Pass HAVAL. In: Miri, A., Vaudenay, S. (eds.) SAC 2011. LNCS, vol. 7118, pp. 1–18. Springer, Heidelberg (2012)

13. Sasaki, Y., Wang, L.: 2-Dimension Sums: Distinguishers Beyond Three Rounds of RIPEMD-128 and RIPEMD-160, `http://eprint.iacr.org/2012/049.pdf`
14. Sasaki, Y., Wang, L., Takasaki, Y., Sakiyama, K., Ohta, K.: Boomerang Distinguishers for Full HAS-160 Compression Function. In: Hanaoka, G., Yamauchi, T. (eds.) IWSEC 2012. LNCS, vol. 7631, pp. 156–169. Springer, Heidelberg (2012)
15. Specification of SM3 cryptographic hash function, `http://www.oscca.gov.cn/UpFile/20101222141857786.pdf` (in Chinese)
16. Wagner, D.: The Boomerang Attack. In: Knudsen, L.R. (ed.) FSE 1999. LNCS, vol. 1636, pp. 156–170. Springer, Heidelberg (1999)
17. Wang, G., Shen, Y.: Preimage and Pseudo-Collision Attacks on Step-Reduced SM3 Hash Function. Information Processing Letters 113(8), 301–306
18. 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)
19. 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)
20. Yu, H., Chen, J., Wang, X.: The Boomerang Attacks on the Round-Reduced Skein-512. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 287–303. Springer, Heidelberg (2013)
21. Zou, J., Wu, W., Wu, S., Su, B., Dong, L.: Preimage Attacks on Step-Reduced SM3 Hash Function. In: Kim, H. (ed.) ICISC 2011. LNCS, vol. 7259, pp. 375–390. Springer, Heidelberg (2012)

# Expressive Cryptography: Lattice Perspectives

Xavier Boyen

Queensland University of Technology

## Abstract of the Invited Lecture

The invention of asymmetric encryption back in the seventies was a conceptual leap that vastly increased the expressive power of encryption of the times. For the first time, it allowed the sender of a message to designate the intended recipient in an cryptographic way, expressed as a "public key" that was related to but distinct from the "private key" that, alone, embodied the ability to decrypt. This made large-scale encryption a practical and scalable endeavour, and more than anything else—save the internet itself—led to the advent of electronic commerce as we know and practice it today.

An equally significant transition is currently underway. Just as public keys gave us the first workable if unpronounceable cryptographic vocabulary, a flurry of advances in the past twelve years have expanded this metaphor in a number of ways. Ranging from intuitive "identity-based" orthography, to modular "attribute-based" grammars, all the way to programmatic "functional" inference semantics, much progress has been made, driven by the common goal to express, in increasingly powerful and flexible ways, the exact beneficiaries of the right to decrypt a given ciphertext.

Under the hood, and even more so than vanilla public-key encryption itself, such expressive achievements have tended to rely on mathematical principles that would, quite unfortunately, render them utterly insecure against quantum computers, should those ever become a working reality. And being that the consensus opinion has shifted from viewing this event as a scientific uncertainty to an engineering challenge, it behooves us to acknowledge and prepare for its eventuality.

In this context, Euclidean lattices are enjoying great interest on the part of the cryptographic research community, owing to a rather unique combination of (conjectured) "post-quantum" resistance and a fertile mathematical structure propitious to the design of rich and expressive cryptosystems.

In this lecture, we shall give an overview of a number of recent results in this area, and discuss the underlying principles that (currently) drive this new and exciting branch of cryptographic research, with an emphasis on the similarities and differences between lattice-based and non-lattice constructions.

# Related-Key Boomerang Attacks on KATAN32/48/64

Takanori Isobe[1], Yu Sasaki[2], and Jiageng Chen[3]

[1] Kobe University
Takanori.Isobe@jp.sony.com
[2] NTT Secure Platform Laboratories
sasaki.yu@lab.ntt.co.jp
[3] Japan Advanced Institute of Science Technology
jg-chen@jaist.ac.jp

**Abstract.** KATAN/KTANTAN is a family of hardware oriented block ciphers proposed at CHES 2009. Although the KTANTAN family have been broken by a meet-in-the-middle approach, the KATAN family are secure at present. In this paper, we investigate the KATAN family in the related-key boomerang framework with several techniques. By using an efficient differential characteristics search method, long boomerang distinguishers can be built. Furthermore, the key recovery phase is optimized by exploiting several properties of the round function such as the high linearity of the round function and the slow key diffusion. As a result, we can attack 174, 145 and 130 rounds of KATAN32, KATAN48 and KATAN64, which substantially improve the known best results whose attacked rounds are 120, 103, 94 rounds, respectively. Our attacks are confirmed by various experimental verifications, especially, we give concrete right quartets for KATAN32.

**Keywords:** KATAN32/48/64, ultra lightweight block cipher, related-key attack, boomerang attack, differential.

## 1 Introduction

KATAN/KTANTAN is a family of lightweight block ciphers designed for extremely resource-constrained devices such as RFID and sensor nodes [1]. After its publication in CHES 2009, the full-round KTANTAN family was theoretically broken by using a meet-in-the-middle approach [2]. The attack takes advantage of the simple key scheduling algorithm for the KTANTAN family. The complexity of the attack was later improved by using the splice-and-cut technique [3]. Armed with related-key model, KTANTAN family can even be broken in practical time [4]. For the KATAN family where the key is loaded into a register and updated in each round, the meet-in-the-middle approach is not likely to work well as the cases of KTANTAN. In the single-key setting, a conditional differential attack is applied to 78, 70 and 68 rounds of KATAN32, KATAN48 and KATAN64, respectively [5]. These results were further improved by using a variant of the meet-in-the-middle approach to 110, 100 and 94 rounds [6]. Also, a differential-style attack broke the 115-round KATAN32 [7]. Even in the related-key setting, only 120, 103 and 90 rounds for the respective three versions were broken by the conditional differential attack [8]. Given the full 254 rounds, the KATAN family seem to have enough security margin at present. Note that the accelerating key searches for the full KATAN32/48/64 were presented in [9].

In this paper, we further investigate the security of the KATAN family in the related-key boomerang framework. In order to build a long and efficient boomerang distinguisher, we use an efficient differential characteristics search strategy. Generally speaking, this strategy is inspired by observing that there exists 39 consecutive rounds where the related key difference is zero. We call it blank step, and by fixing the starting round of the blank step, we can go backwards and forwards to compute the input and output differences for both $E_0$ and $E_1$. Since the key scheduling algorithm is linear for the KATAN family, key difference fixed in $E_1$ can still be propagated in backwards deterministically. Although a similar strategy was used for conditional differential attacks [5,8], we optimize it for boomerang-type attacks. In particular, we carefully choose sets of input differences which are likely to produce differential characteristics with very high probability, and then exhaustively search for differential characteristics of each input set. The probability for $E_0$ can be further controlled by adding conditions in the plaintexts. By taking multiple output differences for $E_0$ and multiple input differences for $E_1$ into consideration, we are able to build 140, 119 and 113 rounds related-key boomerang distinguisher for the corresponding three versions. Based on the boomerang distinguisher, we further optimize the key recovery phase by exploiting the property of the round function in order to reduce the complexity as well as increasing the number of attacked rounds. The comparison of the attacks against the KATAN family is summarized in Table 1. Our attacks substantially improve previous attacks for all variants, and are confirmed by various experimental verifications, especially, we give the concrete right quartets for KATAN32 which supports the feasibility of the attack.

**Outline of the Paper** This paper is organized as follows. A description of KATAN and related-key boomerang attack are given in Section 2. The related-key boomerang distinguisher on KATAN32 is shown in Section 3. In Section 4, we present a key recovery attack using the boomerang distinguisher on KATAN32. The analysis of KATAN48/64 is given in Section 5. Finally, we present conclusions in Section 6 with various experimental results showed in Appendix.

## 2 Preliminaries

### 2.1 KATAN Block Cipher

The KATAN family [1] is a feedback shift register-based block cipher consisting of three variants : KATAN32, KATAN48, KATAN64, whose block sizes are 32 bits, 48 bits and 64 bits, respectively. All variants use the same LFSR(Linear Feedback Shift Register)-type key scheduling function accepting an 80-bit key.

**Table 1.** Comparison of attacks against KATAN family

| Cipher | Attacking Technique | #Rounds | Time | Data | Mem. | Reference |
|---|---|---|---|---|---|---|
| KATAN32 | Differential (SK) | 78 | $2^{76}$ | $2^{16}$ CP | Not given | [5] |
| | MitM (SK) | 110 | $2^{77}$ | 138 KP | $2^{75.1}$ | [6] |
| | Differential (SK) | 115 | $2^{79}$ | 138 KP | $2^{75.1}$ | [7] |
| | Differential (RK) | 120 | $2^{31}$ | Practical (**CP**) | Practical | [8] |
| | **Boomerang (RK)** | **172** | $\mathbf{2^{76.2}}$ | $\mathbf{2^{27.6}}$ **CP** | $\mathbf{2^{26.6}}$ | **Ours** |
| | **Boomerang (RK)** | **173** | $\mathbf{2^{77.5}}$ | $\mathbf{2^{27.6}}$ **CP** | $\mathbf{2^{26.6}}$ | **Ours** |
| | **Boomerang (RK)** | **174** | $\mathbf{2^{78.8}}$ | $\mathbf{2^{27.6}}$ **CP** | $\mathbf{2^{26.6}}$ | **Ours** |
| KATAN48 | Differential (SK) | 70 | $2^{78}$ | $2^{31}$ CP | Not given | [5] |
| | MitM (SK) | 100 | $2^{78}$ | 128 KP | $2^{78}$ | [6] |
| | Differential (RK) | 103 | $2^{25}$ | Practical (**CP**) | Practical | [8] |
| | **Boomerang (RK)** | **145** | $\mathbf{2^{78.5}}$ | $\mathbf{2^{38.4}}$ **CP** | $\mathbf{2^{37.4}}$ | **Ours** |
| KATAN64 | Differential (SK) | 68 | $2^{78}$ | $2^{32}$ CP | Not given | [5] |
| | MitM (SK) | 94 | $2^{77.68}$ | 116 KP | $2^{77.68}$ | [6] |
| | Differential (RK) | 90 | $2^{27}$ | Practical (**CP**) | Practical | [8] |
| | **Boomerang (RK)** | **130** | $\mathbf{2^{78.1}}$ | $\mathbf{2^{53.1}}$ **CP** | $\mathbf{2^{52.1}}$ | **Ours** |

SK: Single Key, RK: Related Key, KP: Know Plaintext, CP:Chosen Plaintext.

The key scheduling function expands an 80-bit user-provided key $k_i$ ($0 \leq i < 80$) into a 508-bit subkey $sk_i$ ($0 \leq i < 508$) by the following linear operations,

$$sk_i = \begin{cases} k_i \ (0 \leq i < 80), \\ k_{i-80} \oplus k_{i-61} \oplus k_{i-50} \oplus k_{i-13} \ (80 \leq i < 508). \end{cases}$$

These operations are expressed as an 80-bit LFSR whose polynomial is $x_{80} + x_{61} + x_{50} + x_{13} + 1$ as shown in Fig 1.

In the round function, each bit of a plaintext is loaded into registers $L_1$ and $L_2$. Then, these are updated as follows:

$$\begin{aligned} f_a(L_1) &= L_1[x_1] \oplus L_1[x_2] \oplus (L_1[x_3] \cdot L_1[x_4]) \oplus (L_1[x_5] \cdot IR) \oplus k_a, \\ f_b(L_2) &= L_2[y_1] \oplus L_2[y_2] \oplus (L_2[y_3] \cdot L_2[y_4]) \oplus (L_2[y_5] \cdot L_2[y_6]) \oplus k_b, \\ L_1[i] &= L_1[i-1] \ (1 \leq i < |L_1|), \quad L_1[0] = f_b(L_2), \\ L_2[i] &= L_2[i-1] \ (1 \leq i < |L_2|), \quad L_2[0] = f_a(L_1), \end{aligned}$$

where $\oplus$ and $\cdot$ are bitwise XOR and AND operations, respectively, and $L[x]$ denotes the $x$-th bit of $L$, $IR$ is the round constant value defined in the specification, and $k_a$ and $k_b$ are two subkey bits. Table 2 shows the detailed parameters of KATAN32/48/64. For round $i$, $k_a$ and $k_b$ correspond to $sk_{2(i-1)}$ and $sk_{2(i-1)+1}$, respectively. After 254 rounds (from 1 to 254) values of registers are output as a ciphertext. Fig. 2 illustrates the round function of KATAN32.

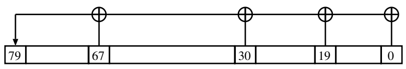


**Fig. 1.** Key scheduling function of KATAN32/48/64

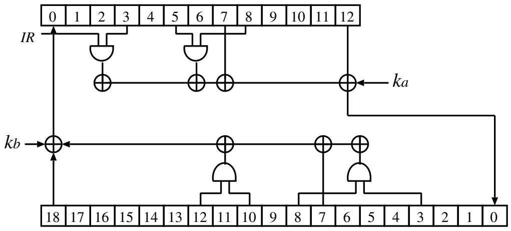


**Fig. 2.** Round function of KATAN32

**Table 2.** Parameters of KATAN family

| Algorithm | $\|L_1\|$ | $\|L_2\|$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $y_1$ | $y_2$ | $y_3$ | $y_4$ | $y_5$ | $y_6$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| KATAN32 | 13 | 19 | 12 | 7 | 8 | 5 | 3 | 18 | 7 | 12 | 10 | 8 | 3 |
| KATAN48 | 19 | 29 | 18 | 12 | 15 | 7 | 6 | 28 | 19 | 21 | 13 | 15 | 6 |
| KATAN64 | 25 | 39 | 24 | 15 | 20 | 11 | 9 | 38 | 25 | 33 | 21 | 14 | 9 |

## 2.2 Related-Key Boomerang Attack

The related-key boomerang attack [10,11,12] is a combination of the boomerang attack [13], and the related-key differential attack [14,15,16].

**Boomerang-Type Attack.** The main idea behind the boomerang attack [13] is to use two short differentials with high probability instead of one long differential with low probability. Suppose that a block cipher with $n$-bit block and $k$-bit key, $E : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n$, is expressed as a cascade cipher $E = E_1 \circ E_0$, where $E_0$ has a differential $\alpha \to \beta$ with probability $p$, and $E_1$ has a differential $\gamma \to \delta$ with probability $q$. Then, the distinguisher is mounted as follows:

**1** : Ask for the ciphertexts $C_1 = E(P_1)$ and $C_2 = E(P_2)$, where $P_2 = P_1 \oplus \alpha$.
**2** : Ask for the plaintexts $P_3 = E^{-1}(C_3)$ and $P_4 = E^{-1}(C_4)$, where $C_3 = C_1 \oplus \delta$ and $C_4 = C_2 \oplus \delta$.
**3** : Check whether $P_3 \oplus P_4 = \alpha$.

Here, $E$ satisfies the condition of $P_3 \oplus P_4 = \alpha$ with probability of $p^2q^2$, while that of a random permutation is $2^{-n}$. Note that the attack can be mounted for all possible $\beta$'s and $\gamma$'s simultaneously. Therefore, the probability is improved to $\hat{p}^2\hat{q}^2$ from $p^2q^2$, where $\hat{p} = \sqrt{\sum_\beta Pr^2[\alpha \to \beta]}$ and $\hat{q} = \sqrt{\sum_\gamma Pr^2[\gamma \to \delta]}$.

The amplified boomerang attack converts the adaptive setting into the non-adaptive one [17]. It exploits the birthday paradox in the middle round. An attacker encrypts many plaintext pairs with a difference $\alpha$, and collects plaintext/ciphertext quartets. Then, she searches for right quartets in the form of $P_1 \oplus P_2 = P_3 \oplus P_4 = \alpha$ and $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$. For $E$, this event occurs if the following three conditions are satisfied:

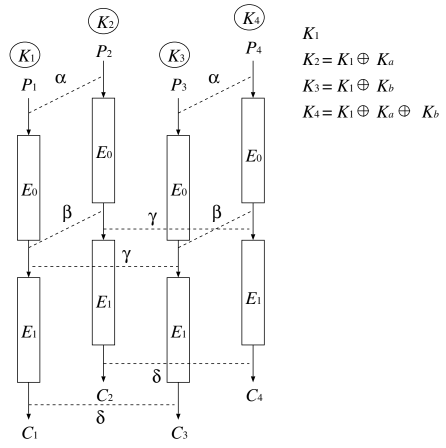


**Fig. 3.** Related-key boomerang quartet

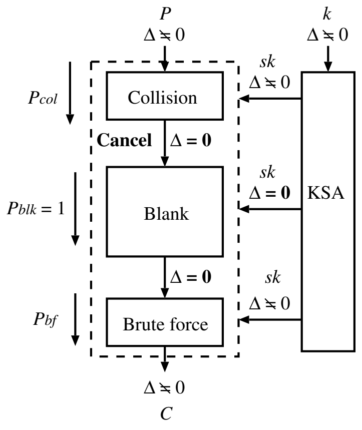


**Fig. 4.** Strategy for finding differential characteristics

**Condition 1** : $E_0(P_1) \oplus E_0(P_2) = E_0(P_3) \oplus E_0(P_4) = \beta$,
**Condition 2** : $E_0(P_1) \oplus E_0(P_3)(\text{or } E_0(P_2) \oplus E_0(P_4)) = \gamma$,
**Condition 3** : $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$.

The probability that a quartet is the right one is $2^{-n}p^2q^2$. For a random permutation, this event occurs with probability of $2^{-2n}$. Thus, if $pq > 2^{-n/2}$, we can distinguish $E$ from a random permutation. Given $N$ plaintext pairs having $\alpha$ difference, there are $\binom{N}{2} \times 2 \approx N^2$ quartets. Thus, the expected number of right quartets in $N$ pairs is $N^2 \cdot 2^{-n}p^2q^2$. The rectangle attack [18] exploits all $\beta$ and $\gamma$ to improve the amplified boomerang attack. If $\hat{p}^2\hat{q}^2 > 2^{-n/2}$, this distinguisher works. Though the rectangle attack requires a large amount of data, it can perform a key recovery phase in the non-adaptive setting.

In this paper, we refer boomerang-type attack using amplified and rectangle techniques to boomerang attack for sake of simplicity.

**Related-Key Boomerang Attack.** The related-key boomerang attack [10,11,12] additionally uses key differences. See Fig. 3 for its illustration. Assume that $E_0$ has a differential $\alpha \to \beta$ under a key difference $\Delta K_a$ with probability $\hat{p}$, and $E_1$ has a differential $\gamma \to \delta$ under a key difference $\Delta K_b$ with probability $\hat{q}$. A related-key distinguisher is constructed by using four different unknown keys, $K_1$, $K_2 = K_1 \oplus K_a$, $K_3 = K_1 \oplus K_b$, $K_4 = K_1 \oplus K_a \oplus K_b$, as follows:

**1** : Ask $N$ ciphertext pairs $(C_1, C_2)$, where $C_1 = E_{K_1}(P_1)$, $C_2 = E_{K_2}(P_2)$ and $P_1 \oplus P_2 = \alpha$. Define the set of these pairs as $S$.
**2** : Ask $N$ ciphertext pairs $(C_3, C_4)$, where $C_3 = E_{K_3}(P_3)$, $C_4 = E_{K_4}(P_4)$ and $P_3 \oplus P_4 = \alpha$. Define the set of these pairs as $T$.
**3** : Find right quartets satisfying the following conditions from $S$ and $T$:
$P_1 \oplus P_2 = P_3 \oplus P_4 = \alpha$ and $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$,

**Table 3.** Output differences of $z$ for each input value $(x, y)$ and its difference

| Value of | Difference of $(x, y)$ | | |
|---|---|---|---|
| $(x, y)$ | (0,1) | (1,0) | (1,1) |
| (0,0) | 0 | 0 | 1 |
| (0,1) | 0 | 1 | 0 |
| (1,0) | 1 | 0 | 0 |
| (1,1) | 1 | 1 | 1 |

**Table 4.** Sets of input diff. with Hamming weight 2

| set | key difference | no-difference subkeys | plaintext differences | $P_{col}$ |
|---|---|---|---|---|
| 0 | 0, 19 | 20 – 98 | $L_2[9], L_1[12]$ | $2^{-2}$ |
| 1 | 1, 20 | 21 – 99 | $L_2[18], L_1[2, 7, 12]$ | $2^{-3}$ |
| 2 | 2, 21 | 22 – 100 | $L_2[8], L_1[11]$ | $2^{-3}$ |
| 3 | 3, 22 | 23 – 101 | $L_2[17], L_1[1, 6, 11]$ | $2^{-3}$ |
| 4 | 4, 23 | 24 – 102 | $L_2[7, 18], L_1[10]$ | $2^{-3}$ |
| 5 | 5, 24 | 25 – 103 | $L_2[16], L_1[0, 5, 10]$ | $2^{-4}$ |
| 6 | 6, 25 | 26 – 104 | $L_2[6, 17], L_1[9]$ | $2^{-3}$ |
| 7 | 7, 26 | 27 – 105 | $L_2[15, 18], L_1[4, 9]$ | $2^{-3}$ |
| 8 | 8, 27 | 28 – 106 | $L_2[5, 16], L_1[8]$ | $2^{-4}$ |
| 9 | 9, 28 | 29 – 107 | $L_2[14, 17], L_1[3, 8]$ | $2^{-5}$ |
| 10 | 10, 29 | 30 – 108 | $L_2[4, 15], L_1[7, 12]$ | $2^{-4}$ |

## 3 Related-Key Boomerang Distinguisher on KATAN32

In this section, we introduce an effective search strategy for finding good related-key differential characteristics. This technique exploits the linearity of the key scheduling and the low dependency of subkey bits. Although a similar search strategy was used in [5,8], we optimize it for a boomerang-type attack.

### 3.1 Differential Properties of KATAN

**Round Function.** Let us consider an XOR differential property of the round function of KATAN in which there are four nonlinear components, *i.e.*, AND operations. Table 3 shows the differential property of the AND operation whose inputs are $x, y$ and the output is $z$, namely $z = x \cdot y$. For example, for the value $(1, 0)$ and the difference $(1, 0)$, the difference of $z$ is obtained as $(x \cdot y) \oplus ((x \oplus 1) \cdot (y \oplus 0)) = (1 \cdot 0) \oplus (0 \cdot 1) = 0$. From Table 3, when input values have any differences, the output also has a difference with probability $2^{-1}(= 6/12)$.

Besides, one AND operation takes $IR$ as one of the input bits. If one of input bits is public and constant, the corresponding output difference is deterministic. Thus, we can focus on only *three* AND operations as nonlinear operations.

**Key Scheduling Function.** The key scheduling function employs only linear operations based on the LFSR. Then, we obtain the following observation.

**Observation.** *Choosing input key differences properly, 79 consecutive subkey bits have no differences after the key scheduling function.*

Since the key scheduling function is the 80-bit LFSR-type construction, any 80 consecutive subkeys surely contain some differences if the key has differences. However, if only one bit of $k_i$ $(0 \leq i \leq 18)$ has a difference, there is no differences in $k_{i+1} - k_{i+79}$, because $k_i$ is not used until $k_{i+80}$. As for the other case, Table 4 shows all possible sets of 2-bit input key differences producing such 79-bit no differences subkeys. For example, assuming that $k_0$ and $k_{19}$ have differences (set 0), $k_{20} - k_{98}$ do not have differences because the difference $k_0$ is canceled by $k_{19}$ when it is used for computing $k_{80}$. The same event occurs in other sets 1-10.

Note that, for all 79 consecutive subkey bits, we can generate the subkey difference which does not make any difference for the target 79 subkey bits. This can be done by the kernel computing approach in [19]. However these sets do not give advantage compared to the sets 1-10, and thus we omit the details.

## 3.2 Strategy for Finding Differential Characteristics

We introduce an effective search strategy for finding good related-key differential characteristics. It is well-suited for boomerang-type attacks in terms of short differential characteristics with very high probability. In general, it is difficult to find good differential characteristics for a bit-oriented cipher due to the large search space. Besides, the related-key setting where key differences are additionally inserted makes it more difficult. In order to get rid of this problem, our strategy is focusing on particular input differential sets which are expected to give good characteristics for boomerang-type attacks.

The differential characteristic search strategy consists of a collision step, a blank step and a brute force step as shown in Fig.4.

**Collision Step** : Plaintext difference and key difference cancel each other.
**Blank Step** : No difference exists in registers and inserted subkeys.
**Brute Force Step** : Subkey differences propagate to the registers.

The key idea of this strategy is to construct the rounds having no difference called *blank round.* Since the blank round does not reduce the differential probability, *i.e.*, differential probability of such rounds is one, we expect to obtain differential characteristics with high probability. For constructing a long blank round, we utilize the observation 1: *we can set 79 consecutive subkey bits having no difference.* If there is no difference in registers where these 79-bit subkeys are used, the blank round can be easily constructed. In other words, we properly choose plaintext difference for canceling out subkey differences just before the blank round. Table 4 shows the plaintext differences for canceling the corresponded input key differences before the blank round and its probability. After the blank round, we search for all differential characteristics. As mentioned before, we regard three AND operations as nonlinear components. Let $P_{col}$, $P_{blk}$ and $P_{bf}$ be the differential probability of each step, respectively. The whole differential characteristic probability is calculated as $P_{col} \cdot P_{blk} \cdot P_{bf}$, where $P_{blk} = 1$.

Input key differences are restricted to the set satisfying the property of the observation 1. Then, plaintext differences are also determined from the set of input key differences for constructing the blank round (see Table 4).

## 3.3 Related-Key Boomerang Distinguisher on 140-Round KATAN32

Using the efficient differential characteristics search, we obtain the maximum probability of differential characteristics of each input set in $E_0$ starting from round 1 (see Table 5).

**Table 5.** Maximum probability of differential characteristics of each set in $E_0$

| Round | **set0** | **set1** | **set2** | **set3** | **set4** | **set5** | **set6** | **set7** | **set8** | **set9** | **set10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 65 | $2^{-9}$ | $2^{-9}$ | $2^{-7}$ | $2^{-8}$ | $2^{-7}$ | $2^{-8}$ | $2^{-7}$ | $2^{-7}$ | $2^{-7}$ | $2^{-7}$ | $2^{-7}$ |
| 66 | $2^{-10}$ | $2^{-10}$ | $2^{-7}$ | $2^{-9}$ | $2^{-7}$ | $2^{-9}$ | $2^{-7}$ | $2^{-8}$ | $2^{-8}$ | $2^{-8}$ | $2^{-8}$ |
| 67 | $2^{-12}$ | $2^{-10}$ | $2^{-8}$ | $2^{-10}$ | $2^{-7}$ | $2^{-10}$ | $2^{-7}$ | $2^{-9}$ | $2^{-8}$ | $2^{-9}$ | $2^{-9}$ |
| 68 | $2^{-13}$ | $2^{-11}$ | $2^{-9}$ | $2^{-10}$ | $2^{-8}$ | $2^{-11}$ | $2^{-7}$ | $2^{-11}$ | $2^{-8}$ | $2^{-10}$ | $2^{-10}$ |
| 69 | $2^{-14}$ | $2^{-12}$ | $2^{-10}$ | $2^{-12}$ | $2^{-9}$ | $2^{-11}$ | $2^{-8}$ | $2^{-12}$ | $2^{-8}$ | $2^{-11}$ | $2^{-11}$ |
| 70 | $2^{-15}$ | $2^{-12}$ | $2^{-12}$ | $2^{-12}$ | $2^{-10}$ | $2^{-12}$ | $2^{-9}$ | $2^{-12}$ | $2^{-9}$ | $2^{-12}$ | $2^{-12}$ |
| 71 | $2^{-16}$ | $2^{-13}$ | $2^{-13}$ | $2^{-12}$ | $2^{-12}$ | $2^{-13}$ | $2^{-10}$ | $2^{-14}$ | $2^{-10}$ | $2^{-12}$ | $2^{-12}$ |

**Table 6.** Maximum probability of differential characteristics of each set in $E_1$

| Round | **set0** | **set1** | **set2** | **set3** | **set4** | **set5** | **set6** | **set7** | **set8** | **set9** | **set10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 65 | $2^{-6}$ | $2^{-10}$ | $2^{-7}$ | $2^{-8}$ | $2^{-9}$ | $2^{-8}$ | $2^{-7}$ | $2^{-7}$ | $2^{-8}$ | $2^{-7}$ | $2^{-6}$ |
| 66 | $2^{-7}$ | $2^{-11}$ | $2^{-7}$ | $2^{-9}$ | $2^{-9}$ | $2^{-9}$ | $2^{-7}$ | $2^{-8}$ | $2^{-9}$ | $2^{-8}$ | $2^{-7}$ |
| 67 | $2^{-8}$ | $2^{-11}$ | $2^{-8}$ | $2^{-10}$ | $2^{-10}$ | $2^{-11}$ | $2^{-7}$ | $2^{-9}$ | $2^{-10}$ | $2^{-9}$ | $2^{-8}$ |
| 68 | $2^{-9}$ | $2^{-13}$ | $2^{-9}$ | $2^{-10}$ | $2^{-11}$ | $2^{-12}$ | $2^{-7}$ | $2^{-10}$ | $2^{-10}$ | $2^{-10}$ | $2^{-8}$ |
| 69 | $2^{-11}$ | $2^{-13}$ | $2^{-10}$ | $2^{-12}$ | $2^{-13}$ | $2^{-12}$ | $2^{-8}$ | $2^{-11}$ | $2^{-11}$ | $2^{-11}$ | $2^{-8}$ |
| 70 | $2^{-12}$ | $2^{-13}$ | $2^{-12}$ | $2^{-12}$ | $2^{-14}$ | $2^{-13}$ | $2^{-9}$ | $2^{-11}$ | $2^{-12}$ | $2^{-12}$ | $2^{-8}$ |
| 71 | $2^{-15}$ | $2^{-14}$ | $2^{-13}$ | $2^{-12}$ | $2^{-15}$ | $2^{-14}$ | $2^{-10}$ | $2^{-12}$ | $2^{-14}$ | $2^{-12}$ | $2^{-9}$ |

To construct a distinguisher, we choose 70 rounds of set 8 whose probability is highest of all the sets. $E_0$ has 8 characteristics with probability $p = 2^{-9}$, 16 characteristics with probability $p = 2^{-10}$, 16 characteristics with probability $p = 2^{-11}$ and 64 characteristics with probability $p = 2^{-12}$, which are generated from the same input. Thus, the overall probability for $E_0$ is

$$\hat{p} = \sqrt{(2^{-9})^2 \cdot 8 + (2^{-10})^2 \cdot 16 + (2^{-11})^2 \cdot 16 + (2^{-12})^2 \cdot 64} \approx 2^{-7.1}.$$

Table 11 in Appendix gives a single differential trail of $E_0$ with probability of $2^{-9}$, where round 0 means initial differences, *i.e.*, differences of a plaintext.

Since KATAN employs the LFSR-based key scheduling, all 508 subkey bits can be calculated from any consecutive 80 subkey bits. It means that we can use the efficient differential characteristics search strategy from any round by regarding the consecutive 80 subkey bits as the master key bits. Thus, we search for differential characteristics of $E_1$ from round 71 with the same strategy.

Table 6 shows the maximum probability of differential characteristics of each set in $E_1$ starting from round 71. We choose set 1 as $E_1$ with probability $2^{-8}$. In addition, $E_1$ has 4 characteristics with probability $2^{-8}$, 8 characteristics with probability $2^{-9}$ and 32 characteristics with probability $2^{-10}$, which produce the same output difference. Thus, the total probability for $E_1$ is estimated as

$$\hat{q} = \sqrt{(2^{-8})^2 \cdot 4 + (2^{-9})^2 \cdot 8 + (2^{-10})^2 \cdot 32} \approx 2^{-6.5}.$$

Table 12 in Appendix gives a single differential trail of $E_1$ with probability $2^{-8}$.

Combining these two-type differential characteristics, 140 (=70+70)-round related-key boomerang distinguisher can be constructed with probability of

$$\hat{p}^2 \cdot \hat{q}^2 = (2^{-7.1})^2 \cdot (2^{-6.5})^2 = 2^{-27.2} \ (> 2^{-32}).$$

The probability of the boomerang distinguisher, $2^{-27.2}$, is possible to verify practically. We performed the experiment on a standard PC and found right quartets within a few minutes. One example is shown in Table 10 in Appendix.

## 4 Related-Key Recovery Attack on KATAN32

In this section, a related-key attack on KATAN32 is proposed given the 140-round boomerang distinguisher. One of the challenging problems is how to reduce the candidate quartets. This is usually achieved by studying the propagation of the difference to the ciphertext in order to filter out definitely wrong quartets. For the KATAN family, this may not be the best option. When we extend the attacking rounds as long as possible, the difference propagation will leave us with no clue. Instead, we try to choose plaintext so that the characteristic for the first several rounds are always satisfied. This strategy is also used in previous KATAN attacks [5,8]. We further optimize the key recovery phase by exploiting the property of the round function, in order to reduce the complexity as well as increasing the number of attacked rounds.

### 4.1 Conditions for Chosen Plaintexts

In the collision step for $E_0$, we have calculated that $p_{col} = 2^{-4}$. Recall that for two inputs to an AND gate, one input with value 1 will guarantee the propagation of the difference from the other input, and difference will disappear when the value is fixed to 0. Thus we can assure the difference propagation with probability 1 by fixing some of the plaintext bits. For KATAN32, the probability for the collision steps can be increased to 1. The conditions on plaintext bits are $L_2[0] = L_2[3] = L_2[7] = L_1[5] = 0$, and the increased probability for $E_0$ is

$$\hat{p} = \sqrt{(2^{-9+4})^2 \cdot 8 + (2^{-10+4})^2 \cdot 16 + (2^{-11+4})^2 \cdot 16 + (2^{-12+4})^2 \cdot 64} \approx 2^{-3.1}.$$

This indicates that we can expect one right quartet in $2^{51.2}(= (2^{-3.1})^2 \cdot (2^{-6.5})^2 \cdot 2^{-32})$. As a result, the number of quartet candidates is reduced to $2^{51.2}$.

### 4.2 Optimizing Key Recovery Phase

Suppose that we append $x$ rounds to the end of the 140-round distinguisher. The attacker queries $N$ pairs of plaintexts to oracles with $K_1$ and $K_2$. She also queries $N$ pairs of plaintexts to oracles with $K_3$ and $K_4$. Then, $N^2$ quartets are constructed. We set $N \leftarrow \hat{p}^{-1} \cdot \hat{q}^{-1} \cdot 2^{n/2}$ so that a right quartet is generated.

To recover subkeys for the last $x$ rounds with a straight-forward method, the attacker guesses all subkeys for the last $x$ rounds, and performs partial decryptions until the end of the 140-round distinguisher for each of $N^2$ quartets. Let $g_i$, where $i \in \{1, 2, 3, 4\}$ be a set of subkey bits used in the last $x$

rounds for the $K_i$ oracle. Because KATAN uses two subkey bits in each round, each $g_i$ contains $2x$ subkey bits. We denote the $x$-round partial decryption for a ciphertext $C_i$ with a guessed key $g_i$ by $D_{g_i}(C_i)$. Note that if the guess for $K_1$ oracle, $g_1$, is determined, the corresponding $g_2, g_3$, and $g_4$ are determined uniquely. If the guessed value is correct, the attacker will find one quartet such that $D_{g_1}(C_1) \oplus D_{g_3}(C_3) = D_{g_2}(C_2) \oplus D_{g_4}(C_4) = \delta$. If such a quartet is not found, the guess is wrong. Unfortunately, the complexity of this approach is too high. Let $\#g$ be the number of subkey bits in each of $g_i$, namely $2x$. The approach requires $N^2 \cdot \#g \cdot 4$ partial decryptions, where a factor of $N^2$ is too high.

**Pairwise Approach.** We propose a more efficient method. For each guess of $g_1$ and corresponding $g_2, g_3, g_4$, we perform the partial decryption for $N$ pairs of $(C_1, C_2)$ and $N$ pairs of $(C_3, C_4)$ independently, and identify the right quartet by checking their match as follows:

1. Make a guess for $g_1$ and determine the corresponding values for $g_2, g_3, g_4$.
2. For all $N$ pairs of $(C_1, C_2)$, compute $(D_{g_1}(C_1) \oplus \delta, D_{g_2}(C_2) \oplus \delta)$ and store them in a table with $N$ entries.
3. For all $N$ pairs of $(C_3, C_4)$, compute $(D_{g_3}(C_3), D_{g_4}(C_4))$ and store them in another table.
4. If the guess is correct, a match is found. Otherwise, the guess is discarded.

This method requires only $N \cdot \#g \cdot 2$ partial decryptions for Step 2 and Step 3 respectively, in total $N \cdot \#g \cdot 4$ partial decryptions. The memory requirement is $2N$ state. The memory for Step 3 can be saved by checking the match as soon as we obtain a pair. Each guess is judged as a right-key candidate if one of $N^2$ quartets satisfies two $n$-bit relations $\delta$. We denote this probability by $P_{right}$, which is $N^2 \cdot 2^{-2n}$. After the analysis, the key space will be $\#g \cdot P_{right} = \#g \cdot N^2 \cdot 2^{-2n}$. The remaining key space will be later examined by the exhaustive search.

**Exploiting Linear Subkey Insertion.** We further optimize the attack by exploiting the round function structure. Recall Fig. 2. If the output *value* for some round $r$ is known, the input *difference* for round $r$ can be computed without guessing subkeys. This is because the 1-round decryption uses subkey values only in the linear operation. The situation continues until unknown values are used as an input of AND operations. In the end, the difference after the $x$-round decryption can be computed only with guessing subkeys for the last $x-4$ rounds.

**Partial Matching.** Another optimization is possible by exploiting the property that only 2 bits are updated in each decryption round. Let us see what will happen if we go back 5 rounds without guessing subkeys. As mentioned above, the difference in all bits can be computed up to 4 rounds. In the next round, the attacker cannot compute the difference of the updated bit $L_1[12]$, while she knows the difference of the other 31 bits ($L_2[18]$ can be computed at this stage). Hence, the match can be performed for 31 bits. The analysis is summarized in

**Table 7.** Partial-matching technique for KATAN32

| #Skipped rounds | Number of bits with unknown differences | | | $P_{right}$ |
|---|---|---|---|---|
| | $L_1$ | $L_2$ | Total | |
| 1–4 | 0 | 0 | 0 | $N^2 \cdot 2^{-64}$ |
| 5 | 1 | 0 | 1 | $N^2 \cdot 2^{-62}$ |
| 6 | 2 | 0 | 2 | $N^2 \cdot 2^{-60}$ |
| 7 | 3 | 1 | 4 | $N^2 \cdot 2^{-56}$ |
| $r(\geq 6)$ | $r-4$ | $r-6$ | $2r-10$ | $N^2 \cdot 2^{-84+4r}$ |

If one subkey bit for the first skipped round is guessed, $P_{right}$ decreases by $2^2$.

Table 7. Let $r$ be the number of rounds which we compute without guessing subkeys. Let $z$ be the number of bits with unknown difference. The match is performed for $32-z$ bits. From Table 7, $z = 2r - 10$ for $r \geq 6$. Because 2 pairs exist in a quartet, $P_{right}$ is $N^2 \cdot 2^{-2(32-z)}$, which is $N^2 \cdot 2^{-84+4r}$. As long as $P_{right}$ is small enough, subkeys can be recovered faster than the exhaustive search.

The idea of checking the difference only for a part of the state is similar to the early abort technique [20]. Our idea is different because the pairwise approach is used and the match of difference cannot be checked round by round.

**Partial Key Guessing.** The last technique for the optimization is partially guessing a subkey, *i.e.*, only guessing 1 bit of a subkey in the first skipped rounds. In Table 7, this makes the number of unknown bits be $2r - 11$ and $P_{right}$ be $N^2 \cdot 2^{-86+4r}$ when $r \geq 6$. Intuitively, the technique increases the computational complexity by 1 bit due to the additional guessed bit, while it increases the efficiency of the filtering function by 2 bits due to two pairs in a quartet.

## 4.3 Attack Procedure and Complexity Evaluation

We append $x = 34$ rounds to the end of the 140-round distinguisher. The number of rounds which we do not guess subkey values, $r$, is 8, but we use the partial key guessing technique. Therefore, $\#g = 53$, where each $g_i$ consists of 52 bits for the last 26 rounds and 1 bit of subkey (either bit is fine) for the 27th last round.

1. Choose $N = 2^{25.6}$ plaintext pairs $(P_1, P_2)$ so that $P_1 \oplus P_2 = \alpha$ and satisfy the 4-bit conditions $L_2[0] = L_2[3] = L_2[7] = L_1[5] = 0$. Query them to the oracles with $K_1$ and $K_2$, and store the corresponding $2^{25.6}$ pairs of $(C_1, C_2)$.
2. Do the same for $(P_3, P_4)$ to obtain $N = 2^{25.6}$ ciphertext pairs $(C_3, C_4)$.
3. Guess $g_1$ and the corresponding $g_2, g_3, g_4$. For each guess, do as follows.
   (a) For $2^{25.6}$ pairs of $(C_1, C_2)$, decrypt them for 26 rounds. Then, further decrypt them by 8 rounds to obtain differences in $32 - (2 \times 8 - 11) = 27$ bits, and take the XOR with $\delta$. Store them in a table with $2^{25.6}$ entries.
   (b) For $2^{25.6}$ pairs of $(C_3, C_4)$, do as follows.
      i. Similarly decrypt the pair for $26 + 8 = 34$ rounds to obtain the differences in 27 bits.
      ii. Check if the match exists between the stored values. If no match is found, delete the guess from the candidate. Otherwise, do as follows.

iii. For exhaustive guesses of $80 - 53 = 27$-bit subkeys which are not guessed yet, check the correctness of the guess by using any pair of plaintext and ciphertext (32-bit match). It it passes the check, then further check the correctness of the guess with two more plaintext-ciphertext pairs. If it passes all checks, output it as the correct key.

For Step 1 and 2, we need $4 * 2^{25.6} = 2^{27.6}$ chosen plaintexts. Step 3a requires $2^{53+25.6} \cdot 2 \cdot 34/174 \approx 2^{77.25}$ 174-round KATAN32 computations. The memory requirement for Step 3a is about $2 \cdot 2^{25.6} = 2^{26.6}$ state values. Step 3(b)i also requires $2^{77.25}$ computations. After Step 3(b)ii, $2^{53} \cdot P_{right} = 2^{53} \cdot (2^{51.2} \cdot 2^{-86+4 \cdot 8}) = 2^{50.2}$ key candidates will remain. Step 3(b)iii requires $2^{50.2+27} = 2^{77.2}$ 174-round KATAN32 computations for the first plaintext-ciphertext pair. Only $2^{77.2-32} = 2^{45.2}$ key candidates are examined for the second pair, and only $2^{45.2-32} = 2^{13.2}$ candidates are examined for the third pair. Hence, the complexity for Step 3(b)iii is $2^{77.2} + 2^{45.2} + 2^{13.2} \approx 2^{77.2}$ 174-round KATAN32 computations.

In summary, the data complexity is $2^{27.6}$ chosen plaintexts, the time complexity is $2^{77.25} + 2^{77.25} + 2^{77.2} \approx 2^{78.8}$ 174-round KATAN32 computations. The memory requirement is $2^{25.6}$ state.

Note that our attack succeeds only if the right quartet is obtained *i.e.*, the differential with a probability of $2^{-51.2}$ is satisfied with $2^{51.2}$ quartets. Hence, the success probability of our attack is $1 - 1/e \approx 0.63$. On the other hand, the success probability of the brute force attack with $2^{78.8}$ trials is 0.44. Hence, our attack is better than the brute force attack with the same complexity.

Also note that the advantage of our attack becomes clearer if the number of rounds is reduced more. For example, the complexity for 173 or 172 rounds is $2^{77.5}$ or $2^{76.2}$ computations, respectively, with the same data and memory.

## 5 Related-Key Boomerang Attack on KATAN48/64

### 5.1 Differential Characteristics and Plaintext Conditions

First we give differential characteristic for KATAN48. Similar to KATAN32, we start from finding collision steps, and by changing the starting point of the collision steps, we go backwards to derive the input differences and key differences. As a result we build a 119-round boomerang distinguisher for KATAN48. Table 13 and 14 in Appendix demonstrate one characteristic for $E_0$ and $E_1$. We use a fixed characteristic between rounds 1 to 49 of $E_0$ and rounds 70 to 119 for $E_1$, while we use a differential for the other rounds. In total, for $E_0$ there are 32 characteristics with probability $2^{-14}$, 128 characteristics with probability $2^{-15}$ and 128 characteristics with probability $2^{-16}$. For $E_1$ there are 128 characteristics with probability $2^{-12}$. As a result, $\hat{p} = \sqrt{(2^{-14})^2 \cdot 32 + (2^{-15})^2 \cdot 128 + (2^{-16})^2 \cdot 128} = 2^{-10.9}$, $\hat{q} = \sqrt{(2^{-12})^2 \cdot 128} = 2^{-8.5}$.

Differential characteristics for $E_0$ and $E_1$ of KATAN64 are summarized in Table 15 and 16 in Appendix. Due to the more scrambling in each round, the number of the collision steps and the brute force steps are reduced. We use a fixed characteristic between rounds 1 to 46 of $E_0$ and rounds 103 to 113 for

**Table 8.** Partial-matching for KATAN48

| #skipped | #bits with unknown diff. | | | $P_{right}$ |
|---|---|---|---|---|
| rounds | $L_1$ | $L_2$ | Total | |
| 1 | 0 | 0 | 0 | $N^2 \cdot 2^{-96}$ |
| 2 | 1 | 0 | 1 | $N^2 \cdot 2^{-94}$ |
| 3 | 3 | 0 | 3 | $N^2 \cdot 2^{-90}$ |
| 4 | 5 | 1 | 6 | $N^2 \cdot 2^{-84}$ |
| 5 | 7 | 3 | 10 | $N^2 \cdot 2^{-76}$ |
| 6 | 9 | 5 | 14 | $N^2 \cdot 2^{-68}$ |
| $r(\geq 4)$ | $2r-3$ | $2r-7$ | $4r-10$ | $N^2 \cdot 2^{-116+8r}$ |

If one subkey bit for the first skipped round is guessed, $P_{right}$ decreases by $2^4$.

**Table 9.** Partial-matching for KATAN64

| #skipped | #bits with unknown diff. | | | $P_{right}$ |
|---|---|---|---|---|
| rounds | $L_1$ | $L_2$ | Total | |
| 1 | 0 | 0 | 0 | $N^2 \cdot 2^{-128}$ |
| 2 | 2 | 1 | 3 | $N^2 \cdot 2^{-122}$ |
| 3 | 5 | 4 | 9 | $N^2 \cdot 2^{-110}$ |
| 4 | 8 | 7 | 15 | $N^2 \cdot 2^{-98}$ |
| $r(\geq 3)$ | $3r-4$ | $3r-5$ | $6r-9$ | $N^2 \cdot 2^{-146+12r}$ |

If one subkey bit for the first skipped round is guessed, $P_{right}$ decreases by $2^6$.

$E_1$, while we use a differential for the other rounds. For $E_0$ there are 64, 256, 512, 1024, and 1024 characteristics with probability $2^{-16}$, $2^{-17}$, $2^{-18}$, $2^{-19}$, and $2^{-20}$, respectively. For $E_1$ there are 4, 24, 88, 224, 416, 608, 704, 640, and 256 characteristics with probability $2^{-16}$, $2^{-17}$, $2^{-18}$, $2^{-19}$, $2^{-20}$, $2^{-21}$, $2^{-22}$, $2^{-23}$, and $2^{-24}$, respectively. As a result, $\hat{p} = 2^{-12.25}$, and $\hat{q} = 2^{-13.8}$.

The probabilities of the collision steps of $E_0$ for KATAN48/64 are both $2^{-7}$, but this can be improved by $2^7$ by choosing the plaintext satisfying the conditions. The conditions are given below along with the improved probability for $\hat{p}$. $\hat{q}$ is not affected by the chosen plaintext.

**KATAN48.** Conditions: $L_2[0] = L_2[1] = L_2[2] = L_2[11] = L_2[17] = 0, L_2[10] \neq L_2[18]$. $\hat{p} = \sqrt{(2^{-14+7})^2 \cdot 32 + (2^{-15+7})^2 \cdot 128 + (2^{-16+7})^2 \cdot 128} = 2^{-3.9}$. We expect $2^{72.8} (= (2^{3.9})^2 \cdot (2^{8.5})^2 \cdot 2^{48})$ quartets before a right one shows up.

**KATAN64.** Conditions: $L_2[6] = L_2[7] = L_2[8] = L_2[21] = L_2[30] = 0, L_2[20] \neq L_2[32], L_2[19] \neq L_2[31]$. $\hat{p}$ becomes $2^{-5.25}$. We expect $2^{102.1} (= (2^{5.25})^2 \cdot (2^{13.8})^2 \cdot 2^{64})$ quartets before a right one shows up.

## 5.2 Optimization and Summary of Key Recovery Attacks

The overall strategy is the same as the one for KATAN32. The only difference from KATAN32 is the impact of the partial-matching technique, which comes from the different register sizes $|L_1|, |L_2|$ and input-bit positions for AND operations. The results are summarized in Table 8 and Table 9.

**145-Round KATAN48.** The attack generates $\hat{p}^{-1} \cdot \hat{q}^{-1} \cdot 2^{48/2} = 2^{3.9+8.5+24} = 2^{36.4}$ pairs of $(P_1, P_2)$, and $2^{36.4}$ pairs of $(P_3, P_4)$. This makes $2^{72.8}$ quartets, which include a right quartet with probability 0.63. We append 26 rounds after the 119-round distinguisher. Hence, 145 rounds are attacked. In the key recover phase, we guess 42 bits of subkeys for the last 21 rounds. Therefore, the number of skipped steps, $r$, is 5. This makes the time complexity for the analysis for $P_1, P_2$ pairs be $2^{36.4+42} \cdot 2 \cdot 26/145 \approx 2^{76.9}$ 145-round KATAN48 computations. The memory requirement is $2 \cdot 2^{36.4} = 2^{37.4}$ state values. The analysis for $P_3, P_4$ pairs also requires $2^{76.9}$ 145-round KATAN48 computations. $P_{right}$ is $2^{72.8} \cdot 2^{-76} = 2^{-3.2}$. Hence, the complexity for the exhaustive check becomes $2^{80} \cdot P_{right} = 2^{76.8}$. In the end, the data complexity is $4 \cdot 2^{36.4} = 2^{38.4}$ chosen plaintexts. The computational complexity is $2^{76.9} + 2^{76.9} + 2^{76.8} \approx 2^{78.5}$ 145-round KATAN48 computations. The success probability of our attack is 0.63, while the success probability of the brute force attack with the same complexity is 0.35.

**130-Round KATAN64.** The attack generates $\hat{p}^{-1} \cdot \hat{q}^{-1} \cdot 2^{64/2} = 2^{5.25+13.8+32} = 2^{51.05}$ pairs of $(P_1, P_2)$, and $2^{51.05}$ pairs of $(P_3, P_4)$. This makes $2^{102.1}$ quartets, which include a right quartet with probability 0.63. We append 17 rounds after the 113-round distinguisher. Hence, 130 rounds are attacked. In the key recover phase, we guess 28 bits of subkeys for the last 14 rounds. Therefore, the number of skipped steps, $r$, is 3. This makes the time complexity for the analysis for $P_1, P_2$ pairs be $2^{51.05+28} \cdot 2 \cdot 17/130 \approx 2^{77.1}$ 130-round KATAN64 computations. The memory requirement is $2 \cdot 2^{51.05} \approx 2^{52.1}$ state values. The analysis for $P_3, P_4$ pairs also requires $2^{77.1}$ 130-round KATAN64 computations. $P_{right}$ is $2^{102.1} \cdot 2^{-110} = 2^{-7.9}$. Hence, the complexity for the exhaustive check becomes $2^{80} \cdot P_{right} = 2^{72.1}$. In the end, the data complexity is $4 \cdot 2^{51.05} \approx 2^{53.1}$ chosen plaintexts. The computational complexity is $2^{77.1} + 2^{77.1} + 2^{72.1} \approx 2^{78.1}$ 130-round KATAN64 computations. The success probability of our attack is 0.63, while the success probability of the brute force attack with the same complexity is 0.27.

## 6 Conclusion

In this paper, we proposed the related-key boomerang attack to 174, 145 and 130 rounds of KATAN32/48/64, respectively, which dramatically improved the number of attacked rounds compared with the previous results. Examples of the right quartet on KATAN32 confirmed the feasibility of our attack. As far as we know, this is the best result achieved on the KATAN family.

## References

1. 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)
2. 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)
3. 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)

4. Ågren, M.: Some Instant- and Practical-Time Related-Key Attacks on KTANTAN32/48/64. In: Miri, A., Vaudenay, S. (eds.) SAC 2011. LNCS, vol. 7118, pp. 213–229. Springer, Heidelberg (2012)
5. Knellwolf, S., Meier, W., Naya-Plasencia, M.: Conditional Differential Cryptanalysis of NLFSR-Based Cryptosystems. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 130–145. Springer, Heidelberg (2010)
6. Isobe, T., Shibutani, K.: All Subkeys Recovery Attack on Block Ciphers: Extending Meet-in-the-Middle Approach. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 202–221. Springer, Heidelberg (2013)
7. Albrecht, M.R., Leander, G.: An All-In-One Approach to Differential Cryptanalysis for Small Block Ciphers. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 1–15. Springer, Heidelberg (2013)
8. Knellwolf, S., Meier, W., Naya-Plasencia, M.: Conditional Differential Cryptanalysis of Trivium and KATAN. In: Miri, A., Vaudenay, S. (eds.) SAC 2011. LNCS, vol. 7118, pp. 200–212. Springer, Heidelberg (2012)
9. Knellwolf, S.: Accelerated Key Search for the KATAN Family of Block Ciphers. In: ECRYPT Workshop on Lightweight Cryptography (2011)
10. Biham, E., Dunkelman, O., Keller, N.: Related-Key Boomerang and Rectangle Attacks. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 507–525. Springer, Heidelberg (2005)
11. Hong, S., Kim, J., Lee, S., Preneel, B.: Related-Key Rectangle Attacks on Reduced Versions of SHACAL-1 and AES-192. In: Gilbert, H., Handschuh, H. (eds.) FSE 2005. LNCS, vol. 3557, pp. 368–383. Springer, Heidelberg (2005)
12. Kim, J., Kim, G., Hong, S., Lee, S., Hong, D.: The Related-Key Rectangle Attack – Application to SHACAL-1. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 123–136. Springer, Heidelberg (2004)
13. Wagner, D.: The Boomerang Attack. In: Knudsen, L.R. (ed.) FSE 1999. LNCS, vol. 1636, pp. 156–170. Springer, Heidelberg (1999)
14. Biham, E.: New Types of Cryptanalytic Attacks Using Related Keys. J. Cryptology 7(4), 229–246 (1994)
15. Kelsey, J., Schneier, B., Wagner, D.: Key-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 237–251. Springer, Heidelberg (1996)
16. Biham, E., Dunkelman, O., Keller, N.: A Unified Approach to Related-Key Attacks. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 73–96. Springer, Heidelberg (2008)
17. Kelsey, J., Kohno, T., Schneier, B.: Amplified Boomerang Attacks Against Reduced-Round MARS and Serpent. In: Schneier, B. (ed.) FSE 2000. LNCS, vol. 1978, pp. 75–93. Springer, Heidelberg (2001)
18. Biham, E., Dunkelman, O., Keller, N.: The Rectangle Attack - Rectangling the Serpent. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 340–357. Springer, Heidelberg (2001)
19. Aoki, K., Sasaki, Y.: Meet-in-the-Middle Preimage Attacks Against Reduced SHA-0 and SHA-1. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 70–89. Springer, Heidelberg (2009)
20. Lu, J., Kim, J., 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)

## Appendix

**Table 10.** Example of confirmed boomerang quartets for KATAN32

| | | | | | |
|---|---|---|---|---|---|
| $P_1$ | `0x46ec3236` | $C_1$ | `0xee39e8a1` | $K_1$ | `0x22fe640869975423bce9` |
| $P_2$ | `0x4eed3216` | $C_2$ | `0xf19133e1` | $K_2$ | `0x22fe640869975c23bde9` |
| $P_3$ | `0xd2379460` | $C_3$ | `0xee11e925` | $K_3$ | `0xa6ffe4826d8d3228d6c1` |
| $P_4$ | `0xda369440` | $C_4$ | `0xf1b93265` | $K_4$ | `0xa6ffe4826d8d3a28d7c1` |
| $P_1 \oplus P_2$ | `0x08010020` | $C_1 \oplus C_3$ | `0x00280184` | $K_1 \oplus K_2 = K_3 \oplus K_4$ | `0x00000000000008000100` |
| $P_3 \oplus P_4$ | `0x08010020` | $C_2 \oplus C_4$ | `0x00280184` | $K_1 \oplus K_3 = K_2 \oplus K_4$ | `0x8401808a041a660b6a28` |

**Table 11.** Differential characteristic of KATAN32 $E_0$ (1 - 70)

| Round | $L_2[0] \ldots L_2[18]$ | $L_1[0] \ldots L_1[13]$ | $K_a$ | $K_b$ | Pr. |
|---|---|---|---|---|---|
| 0 | 0000010000000000100 | 0000000010000 | 0 | 0 | 1 |
| 1 | 0000001000000000010 | 0000000001000 | 0 | 0 | $2^{-1}$ |
| 2 | 0000000100000000001 | 0000000000100 | 0 | 0 | $2^{-1}$ |
| 3 | 0000000010000000000 | 0000000000010 | 0 | 0 | $2^{-1}$ |
| 4 | 0000000001000000000 | 0000000000001 | 1 | 0 | $2^{-2}$ |
| 5 | 0000000000100000000 | 0000000000000 | 0 | 0 | $2^{-2}$ |
| 6 | 0000000000010000000 | 0000000000000 | 0 | 0 | $2^{-3}$ |
| 7 | 0000000000001000000 | 0000000000000 | 0 | 0 | $2^{-3}$ |
| 8 | 0000000000000100000 | 0000000000000 | 0 | 0 | $2^{-4}$ |
| 9 | 0000000000000010000 | 0000000000000 | 0 | 0 | $2^{-4}$ |
| 10 | 0000000000000001000 | 0000000000000 | 0 | 0 | $2^{-4}$ |
| 11 | 0000000000000000100 | 0000000000000 | 0 | 0 | $2^{-4}$ |
| 12 | 0000000000000000010 | 0000000000000 | 0 | 0 | $2^{-4}$ |
| 13 | 0000000000000000001 | 0000000000000 | 0 | 1 | $2^{-4}$ |
| 14 | 0000000000000000000 | 0000000000000 | 0 | 0 | $2^{-4}$ |
| 53 | 0000000000000000000 | 0000000000000 | 0 | 1 | $2^{-4}$ |
| 54 | 0000000000000000000 | 1000000000000 | 0 | 0 | $2^{-4}$ |
| 55 | 0000000000000000000 | 0100000000000 | 0 | 0 | $2^{-4}$ |
| 56 | 0000000000000000000 | 0010000000000 | 0 | 0 | $2^{-4}$ |
| 57 | 0000000000000000000 | 0001000000000 | 0 | 0 | $2^{-4}$ |
| 58 | 0000000000000000000 | 0000100000000 | 0 | 0 | $2^{-4}$ |
| 59 | 0000000000000000000 | 0000010000000 | 0 | 0 | $2^{-4}$ |
| 60 | 0000000000000000000 | 0000001000000 | 1 | 0 | $2^{-5}$ |
| 61 | 1000000000000000000 | 0000000100000 | 0 | 0 | $2^{-5}$ |
| 62 | 1100000000000000000 | 0000000010000 | 0 | 0 | $2^{-5}$ |
| 63 | 0110000000000000000 | 0000000001000 | 0 | 0 | $2^{-6}$ |
| 64 | 0011000000000000000 | 0000000000100 | 0 | 0 | $2^{-6}$ |
| 65 | 0001100000000000000 | 0000000000010 | 0 | 0 | $2^{-7}$ |
| 66 | 0000110000000000000 | 0000000000001 | 0 | 1 | $2^{-8}$ |
| 67 | 1000011000000000000 | 1000000000000 | 0 | 0 | $2^{-8}$ |
| 68 | 0100001100000000000 | 0100000000000 | 0 | 0 | $2^{-8}$ |
| 69 | 0010000110000000000 | 1010000000000 | 0 | 0 | $2^{-8}$ |
| 70 | 0001000011000000000 | 1101000000000 | 0 | 0 | $2^{-9}$ |

**Table 12.** Differential characteristic of KATAN32 $E_1$ (71 - 140)

| Round | $L_2[0] \ldots L_2[18]$ | $L_1[0] \ldots L_1[13]$ | $K_a$ | $K_b$ | Pr. |
|---|---|---|---|---|---|
| 70 | 0000100000000001000 | 0000000100001 | 0 | 0 | 1 |
| 71 | 0000010000000000100 | 0000000010000 | 0 | 0 | 1 |
| 72 | 0000001000000000010 | 0000000001000 | 0 | 0 | $2^{-1}$ |
| 73 | 0000000100000000001 | 0000000000100 | 0 | 0 | $2^{-1}$ |
| 74 | 0000000010000000000 | 0000000000010 | 0 | 0 | $2^{-1}$ |
| 75 | 0000000001000000000 | 0000000000001 | 1 | 0 | $2^{-2}$ |
| 76 | 0000000000100000000 | 0000000000000 | 0 | 0 | $2^{-2}$ |
| 77 | 0000000000010000000 | 0000000000000 | 0 | 0 | $2^{-3}$ |
| 78 | 0000000000001000000 | 0000000000000 | 0 | 0 | $2^{-3}$ |
| 79 | 0000000000000100000 | 0000000000000 | 0 | 0 | $2^{-4}$ |
| 80 | 0000000000000010000 | 0000000000000 | 0 | 0 | $2^{-4}$ |
| 81 | 0000000000000001000 | 0000000000000 | 0 | 0 | $2^{-4}$ |
| 82 | 0000000000000000100 | 0000000000000 | 0 | 0 | $2^{-4}$ |
| 83 | 0000000000000000010 | 0000000000000 | 0 | 0 | $2^{-4}$ |
| 84 | 0000000000000000001 | 0000000000000 | 0 | 1 | $2^{-4}$ |
| 124 | 0000000000000000000 | 0000000000000 | 0 | 1 | $2^{-2}$ |
| 125 | 0000000000000000000 | 1000000000000 | 0 | 0 | $2^{-2}$ |
| 126 | 0000000000000000000 | 0100000000000 | 0 | 0 | $2^{-3}$ |
| 127 | 0000000000000000000 | 0010000000000 | 0 | 0 | $2^{-3}$ |
| 128 | 0000000000000000000 | 0001000000000 | 0 | 0 | $2^{-3}$ |
| 129 | 0000000000000000000 | 0000100000000 | 0 | 0 | $2^{-4}$ |
| 130 | 0000000000000000000 | 0000010000000 | 0 | 0 | $2^{-4}$ |
| 131 | 0000000000000000000 | 0000001000000 | 1 | 0 | $2^{-5}$ |
| 132 | 1000000000000000000 | 0000000100000 | 0 | 0 | $2^{-5}$ |
| 133 | 1100000000000000000 | 0000000010000 | 0 | 0 | $2^{-5}$ |
| 134 | 0110000000000000000 | 0000000001000 | 0 | 0 | $2^{-6}$ |
| 135 | 0011000000000000000 | 0000000000100 | 0 | 0 | $2^{-6}$ |
| 136 | 0001100000000000000 | 0000000000010 | 0 | 0 | $2^{-7}$ |
| 137 | 0000110000000000000 | 0000000000001 | 0 | 1 | $2^{-8}$ |
| 138 | 1000011000000000000 | 1000000000000 | 0 | 0 | $2^{-8}$ |
| 139 | 0100001100000000000 | 0100000000000 | 0 | 0 | $2^{-8}$ |
| 140 | 0010000110000000000 | 1010000000000 | 0 | 0 | $2^{-8}$ |

**Table 13.** Differential characteristic of KATAN48 $E_0$ (1 - 60)

| Round | $L_2(L_2[0]\dots L_2[28])$ | $L_1(L_1[0]\dots L_1[18])$ | $K_a$ | $K_b$ | Pr. |
|---|---|---|---|---|---|
| 0 | 00000000011000000011000000011 | 0000000000000000011 | 1 | 0 | 1 |
| 1 | 00000000000110000000110000000 | 0000000000000000000 | 0 | 0 | 1 |
| 2 | 00000000000001100000001100000 | 0000000000000000000 | 0 | 0 | $2^{-2}$ |
| 3 | 00000000000000011000000011000 | 0000000000000000000 | 0 | 0 | $2^{-4}$ |
| 4 | 00000000000000000110000000110 | 0000000000000000000 | 0 | 0 | $2^{-5}$ |
| 5 | 00000000000000000001100000001 | 0000000000000000000 | 0 | 0 | $2^{-5}$ |
| 6 | 00000000000000000000011000000 | 0000000000000000000 | 0 | 0 | $2^{-6}$ |
| 7 | 00000000000000000000000110000 | 0000000000000000000 | 0 | 0 | $2^{-7}$ |
| 8 | 00000000000000000000000001100 | 0000000000000000000 | 0 | 0 | $2^{-7}$ |
| 9 | 00000000000000000000000000011 | 0000000000000000000 | 0 | 1 | $2^{-7}$ |
| 10 | 00000000000000000000000000000 | 0000000000000000000 | 0 | 0 | $2^{-7}$ |
| 49 | 00000000000000000000000000000 | 0000000000000000000 | 0 | 1 | $2^{-7}$ |
| 50 | 00000000000000000000000000000 | 1100000000000000000 | 0 | 0 | $2^{-7}$ |
| 51 | 00000000000000000000000000000 | 0011000000000000000 | 0 | 0 | $2^{-7}$ |
| 52 | 00000000000000000000000000000 | 0000110000000000000 | 0 | 0 | $2^{-7}$ |
| 53 | 10000000000000000000000000000 | 0000001100000000000 | 0 | 0 | $2^{-7}$ |
| 54 | 00100000000000000000000000000 | 0000000011000000000 | 0 | 0 | $2^{-9}$ |
| 55 | 00001000000000000000000000000 | 0000000000110000000 | 0 | 0 | $2^{-9}$ |
| 56 | 10000010000000000000000000000 | 0000000000001100000 | 1 | 0 | $2^{-9}$ |
| 57 | 10100000100000000000000000000 | 0000000000000011000 | 0 | 0 | $2^{-10}$ |
| 58 | 00101000001000000000000000000 | 0000000000000000110 | 0 | 0 | $2^{-12}$ |
| 59 | 10001010000010000000000000000 | 0000000000000000001 | 0 | 0 | $2^{-12}$ |
| 60 | 01100010100000100000000000000 | 0000000000000000000 | 0 | 0 | $2^{-14}$ |

**Table 14.** Differential characteristic of KATAN48 $E_1$ (61 - 119)

| Round | $L_2(L_2[0]\dots L_2[28])$ | $L_1(L_1[0]\dots L_1[18])$ | $K_a$ | $K_b$ | Pr. |
|---|---|---|---|---|---|
| 60 | 00000000011000000011000000011 | 0000000000000000011 | 1 | 0 | 1 |
| 61 | 00000000000110000000110000000 | 0000000000000000000 | 0 | 0 | 1 |
| 62 | 00000000000001100000001100000 | 0000000000000000000 | 0 | 0 | $2^{-2}$ |
| 63 | 00000000000000011000000011000 | 0000000000000000000 | 0 | 0 | $2^{-4}$ |
| 64 | 00000000000000000110000000110 | 0000000000000000000 | 0 | 0 | $2^{-5}$ |
| 65 | 00000000000000000001100000001 | 0000000000000000000 | 0 | 0 | $2^{-5}$ |
| 66 | 00000000000000000000011000000 | 0000000000000000000 | 0 | 0 | $2^{-6}$ |
| 67 | 00000000000000000000000110000 | 0000000000000000000 | 0 | 0 | $2^{-7}$ |
| 68 | 00000000000000000000000001100 | 0000000000000000000 | 0 | 0 | $2^{-7}$ |
| 69 | 00000000000000000000000000011 | 0000000000000000000 | 0 | 1 | $2^{-7}$ |
| 70 | 00000000000000000000000000000 | 0000000000000000000 | 0 | 0 | $2^{-7}$ |
| 109 | 00000000000000000000000000000 | 0000000000000000000 | 0 | 1 | $2^{-7}$ |
| 110 | 00000000000000000000000000000 | 1100000000000000000 | 0 | 0 | $2^{-7}$ |
| 111 | 00000000000000000000000000000 | 0011000000000000000 | 0 | 0 | $2^{-7}$ |
| 112 | 00000000000000000000000000000 | 0000110000000000000 | 0 | 0 | $2^{-7}$ |
| 113 | 10000000000000000000000000000 | 0000001100000000000 | 0 | 0 | $2^{-7}$ |
| 114 | 00100000000000000000000000000 | 0000000011000000000 | 0 | 0 | $2^{-9}$ |
| 115 | 00001000000000000000000000000 | 0000000000110000000 | 0 | 0 | $2^{-9}$ |
| 116 | 10000010000000000000000000000 | 0000000000001100000 | 1 | 0 | $2^{9}$ |
| 117 | 10100000100000000000000000000 | 0000000000000011000 | 0 | 0 | $2^{-10}$ |
| 118 | 00101000001000000000000000000 | 0000000000000000110 | 0 | 0 | $2^{-12}$ |
| 119 | 10001010000010000000000000000 | 0000000000000000001 | 0 | 0 | $2^{-12}$ |

**Table 15.** Differential characteristic of KATAN64 $E_0$ (1 - 56)

| Round | $L_2(L_2[0]\ldots L_2[38])$ | $L_1(L_1[0]\ldots L_1[24])$ | $K_a$ | $K_b$ | Pr. |
|---|---|---|---|---|---|
| 0 | 000000000000000000111000000000011100000 | 0000000000000000000000000 | 0 | 0 | 1 |
| 1 | 000000000000000000000111000000000011100 | 0000000000000000000000000 | 0 | 0 | $2^{-3}$ |
| 2 | 000000000000000000000000111000000000011 | 0000000000000000000000000 | 0 | 0 | $2^{-4}$ |
| 3 | 000000000000000000000000000111000000000 | 0000000000000000000000000 | 0 | 0 | $2^{-4}$ |
| 4 | 000000000000000000000000000000111000000 | 0000000000000000000000000 | 0 | 0 | $2^{-4}$ |
| 5 | 000000000000000000000000000000000111000 | 0000000000000000000000000 | 0 | 0 | $2^{-6}$ |
| 6 | 000000000000000000000000000000000000111 | 0000000000000000000000000 | 0 | 1 | $2^{-7}$ |
| 7 | 000000000000000000000000000000000000000 | 0000000000000000000000000 | 0 | 0 | $2^{-7}$ |
| 46 | 000000000000000000000000000000000000000 | 0000000000000000000000000 | 0 | 1 | $2^{-7}$ |
| 47 | 000000000000000000000000000000000000000 | 1110000000000000000000000 | 0 | 0 | $2^{-7}$ |
| 48 | 000000000000000000000000000000000000000 | 0001110000000000000000000 | 0 | 0 | $2^{-7}$ |
| 49 | 000000000000000000000000000000000000000 | 0000001110000000000000000 | 0 | 0 | $2^{-7}$ |
| 50 | 000000000000000000000000000000000000000 | 0000000001110000000000000 | 0 | 0 | $2^{-7}$ |
| 51 | 000000000000000000000000000000000000000 | 0000000000001110000000000 | 0 | 0 | $2^{-10}$ |
| 52 | 110000000000000000000000000000000000000 | 0000000000000001110000000 | 0 | 0 | $2^{-10}$ |
| 53 | 001110000000000000000000000000000000000 | 0000000000000000001110000 | 1 | 0 | $2^{-10}$ |
| 54 | 111001110000000000000000000000000000000 | 0000000000000000000001110 | 0 | 0 | $2^{-13}$ |
| 55 | 110111001110000000000000000000000000000 | 0000000000000000000000001 | 0 | 0 | $2^{-14}$ |
| 56 | 001110111001110000000000000000000000000 | 0000000000000000000000000 | 0 | 0 | $2^{-16}$ |

**Table 16.** Differential characteristic of KATAN64 $E_1$ (57 - 113)

| Round | $L_2(L_2[0]\ldots L_2[38])$ | $L_1(L_1[0]\ldots L_1[24])$ | $K_a$ | $K_b$ | Pr. |
|---|---|---|---|---|---|
| 56 | 000000000000000111000000000011100000000 | 0000000000000000000000000 | 0 | 0 | 1 |
| 57 | 000000000000000000111000000000011100000 | 0000000000000000000000000 | 0 | 0 | 1 |
| 58 | 000000000000000000000111000000000011100 | 0000000000000000000000000 | 0 | 0 | $2^{-3}$ |
| 59 | 000000000000000000000000111000000000011 | 0000000000000000000000000 | 0 | 0 | $2^{-4}$ |
| 60 | 000000000000000000000000000111000000000 | 0000000000000000000000000 | 0 | 0 | $2^{-4}$ |
| 61 | 000000000000000000000000000000111000000 | 0000000000000000000000000 | 0 | 0 | $2^{-4}$ |
| 62 | 000000000000000000000000000000000111000 | 0000000000000000000000000 | 0 | 0 | $2^{-6}$ |
| 63 | 000000000000000000000000000000000000111 | 0000000000000000000000000 | 0 | 1 | $2^{-7}$ |
| 64 | 000000000000000000000000000000000000000 | 0000000000000000000000000 | 0 | 0 | $2^{-7}$ |
| 103 | 000000000000000000000000000000000000000 | 0000000000000000000000000 | 0 | 1 | $2^{-7}$ |
| 104 | 000000000000000000000000000000000000000 | 1110000000000000000000000 | 0 | 0 | $2^{-7}$ |
| 105 | 000000000000000000000000000000000000000 | 0001110000000000000000000 | 0 | 0 | $2^{-7}$ |
| 106 | 000000000000000000000000000000000000000 | 0000001110000000000000000 | 0 | 0 | $2^{-7}$ |
| 107 | 000000000000000000000000000000000000000 | 0000000001110000000000000 | 0 | 0 | $2^{-7}$ |
| 108 | 000000000000000000000000000000000000000 | 0000000000001110000000000 | 0 | 0 | $2^{-10}$ |
| 109 | 110000000000000000000000000000000000000 | 0000000000000001110000000 | 0 | 0 | $2^{-10}$ |
| 110 | 001110000000000000000000000000000000000 | 0000000000000000001110000 | 1 | 0 | $2^{-10}$ |
| 111 | 111001110000000000000000000000000000000 | 0000000000000000000001110 | 0 | 0 | $2^{-13}$ |
| 112 | 110111001110000000000000000000000000000 | 0000000000000000000000001 | 0 | 0 | $2^{-14}$ |
| 113 | 001110111001110000000000000000000000000 | 0000000000000000000000000 | 0 | 0 | $2^{-16}$ |

# Highly Accurate Key Extraction Method for Access-Driven Cache Attacks Using Correlation Coefficient*

Junko Takahashi[1], Toshinori Fukunaga[2], Kazumaro Aoki[1], and Hitoshi Fuji[1]

[1] NTT Secure Platform Laboratories,
3-9-11, Midori-cho Musashino-shi, Tokyo 180-8585, Japan
{takahashi.junko,aoki.kazumaro,fuji.hitoshi}@lab.ntt.co.jp
[2] NTT Technology Planning Department,
3-1, Otemachi 2-chome, Chiyoda-ku, Tokyo 100-8116, Japan
toshi.fukunaga@hco.ntt.co.jp

**Abstract.** This paper proposes a new highly-accurate key extraction method for access-driven cache attacks (CAs). We show that a mathematical correlation method can be utilized to evaluate quantitatively the access-driven CAs. To the best of our knowledge, this is the first study on CAs that clarifies precisely and mathematically the key candidate space based on memory allocation, and analyzes quantitatively how the correlation values change based on the number of plaintexts. We show empirical improvement of the proposed method based on real processors. We correctly examine the correlation between the access timing data and the key within a few minutes even in a noisy environment. Based on the proposed method, we show the key candidate space with the mathematical proof and find the relationship between the correlation values and the number of plaintexts needed to examine the required number of plaintexts for a successful attack.

**Keywords:** Side-Channel Attacks, Cache Attacks, Access-Driven Cache Attacks, Block Ciphers, AES, Software Implementation.

## 1 Introduction

Nowadays, side-channel attacks (SCAs), which intentionally reveal a secret key using physical leakage, represent a real threat to cryptographic devices. A cache attack (CA) is a well-known SCA that utilizes timing differences whether data are loaded from the main memory (hereafter simply memory) or the cache memory. In fact, the cache memory access speed is typically up to two orders of magnitude faster than that for the memory. When some applications access memory, the processor first looks for data in the cache memory. If data are already in the cache memory, a *cache hit* occurs and data are accessed from the cache memory

* This paper is an extended and improved version of two technical reports: concept [1] and its application [2].

without touching the memory. If not, a *cache miss* occurs and data are accessed from the memory or a higher level cache memory. Thus, there are large gaps in the accessing speed between these cases, and a CA utilizes the characteristics of the timing differences. A CA is considered to be powerful because the key can be remotely extracted.

The initial idea of CAs was presented in [3] and [4], and the theoretical studies based on CAs were given later in [5] and [6]. CAs against some block ciphers were experimentally presented in [7] and [8]. It is well-known that CAs are effective against cryptographic implementations that use substitution box (S-box) tables, which are employed in most block ciphers to speed up cryptographic calculation. Three kinds of CAs were proposed based on timing measurements: time-driven CAs, trace-driven CAs, and access-driven CAs. In time-driven CAs [5,8,9,10], the information of the total calculation time is used. In trace-driven CAs [5,11,12], the traces of the timing of the cache hits or cache misses such as those related to power consumption are employed. In access-driven CAs [13,14], the cache behavior is precisely examined by dynamically accessing the cache memory that is shared with a victim. We focus on the access-driven CA because this attack is more powerful compared to the other CAs.

The original idea of access-driven CAs was proposed in [13] and [14]. These studies employed a simplified cryptanalysis to analyze the correct key by predicting the memory accesses. The key analysis is mainly based on a picture analysis of a distribution of the timing data. It is examined whether the predictions are consistent with the correct data to calculate the candidate scores using a guessed key. In fact, the AES key is found using 8000 plaintexts on Athlon 64 without any knowledge about address mapping. In this approach, a rough estimate of the key hypotheses is possible; however, precise and quantitative evaluation cannot be achieved. Furthermore, an erroneous decision may easily occur especially in an environment with severe noise when a picture analysis is employed. Some extended studies of access-driven CAs were proposed in [15], [16], as examples. Neve and Seifert [15] theoretically showed the expected number of outputs for the final round key without noise. Xinjie *et al.* [16] showed an improved attack using 350 plaintexts on AMD 64 and showed the key candidate space could be reduced based on the memory allocation of the S-box tables.

In this paper, we propose a quantitative approach to evaluate the correlation between the timing data and the key. We show that a mathematical correlation method can be quantitatively utilized to evaluate access-driven CAs. The contributions of this paper are: (1) we quantitatively and correctly analyze the correlation between the access timing data and the key even when it is difficult to distinguish the correct key in a noisy environment using a picture analysis; (2) we precisely and mathematically clarify the key candidate space based on the S-box table allocation in the memory; and (3) we quantitatively analyze the relationship between the correlation values and an increase in the number of plaintexts. We experimentally show the above contributions when we measure the timing data on real processors. We believe that we can quantitatively analyze the correctness of the key hypotheses even in noisy environments such

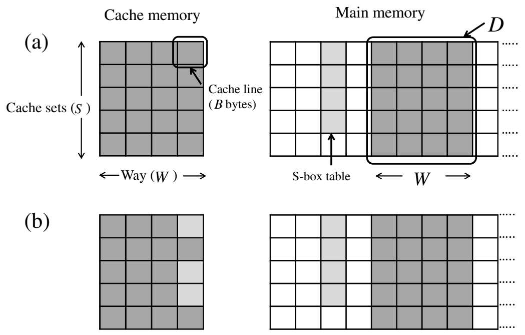


**Fig. 1.** Depiction of the Prime+Probe method

as virtualized environment in which the possibility of CAs was indicated as in [18].

The remainder of this paper is organized as follows. Section 2 gives an overview of access-driven CAs. We present the concept of the proposed method in Section 3 and present the experimental results in Section 4. In Section 5, we give the discussion about the key candidate space based on the memory allocation and the relationship between the correlation values and the number of plaintexts. Finally, our conclusions are given in Section 6.

## 2 Overview of Access-Driven CAs

This section describes the access-driven CAs originally proposed in [13] and [14]. An access-driven CA comprises two attack procedures: the timing data measurements and off-line analysis of the key based on the measured timing data. This paper targets the cache memory with a set-associative cache scheme for the attack. Refer to Appendix A for the details of the cache scheme.

There are two timing measurements: Evict+Time and Prime+Probe methods in [13] and [14]. In this paper, we use Prime+Probe method because it is more practical. Prime+Probe method shown in Fig. 1 comprises in the following steps. First, we allocate a $(S \cdot B \cdot W)$-byte array of the memory $D$, whose start address is a multiple of $S \cdot B$, where $S$ is the number of cache sets, $B$ is the cache line size, and $W$ is the number of ways as described in Fig. 1. It is normally at least as large as the size of the cache memory as indicated by the dark gray area in Fig. 1 (a). We read a value from every memory block in $D$ (Fig. 1(a)) and data will be automatically filled in the cache memory. Then, a single encryption is performed. After the encryption, we read the same value in $D$ again. Thus, we can discern which parts of the data were pushed out of the cache memory (Fig. 1(b)). If a cache line is accessed (untouched) by the encryption, a cache miss (hit) will occur when we read a value in $D$ again. Thus, the timing differences are induced.

In the analysis of the key, we try to examine the correlation between the correct key and the measured timing data given a known plaintext. In the analysis, we deduce unit of the key (usually one byte) based on the picture analysis of the cache memory access patterns. The interested reader can refer to the previous studies in [13] and [14]. We can guess that the timing data correlate to the correct key, when a visible diagonal line appears in the picture display. Thus, we repeatedly examine whether the visible diagonal line appears or not in the picture display of the cache memory access pattern for all key hypotheses.

## 3 Proposed Quantitative Approach

This section describes the concept and the details of the proposed approach.

### 3.1 Concept Behind Proposed Approach

As described in Section 2, a rough analysis of the key estimation for access-driven CAs was achieved in previous studies in [13] and [14]. This approach only uses simple analysis based on the prediction of memory access to examine the correlation with the correct key. Thus, we cannot achieve precise and quantitative analysis to evaluate the key in details.

We employ a mathematical correlation analysis to evaluate quantitatively the relationship between the access timing data and the correct key. And, we show that this method is useful in quantitatively evaluating the access-driven CAs. A correlation analysis is based on deducing a correlation coefficient between all possible computed memory accesses and the real memory accesses. Then, we consider that the highest correlation value reveals the correct key. The correlation method is also used in the field of power analysis to construct a power trace model [19]. In the field of CAs, this method is used to construct an analytical model for time-driven CAs and the required number of cache traces for a successful attack is theoretically estimated based on the perfect environment without noise [10].

In this paper, using a correlation analysis, we show that we can correctly examine the correlation between the timing data and the key even in the noisy environment. And, we precisely and mathematically show that the remaining key candidate space can be reduced even further compared to the previous results [13,14] when the first element of the S-box table is not allocated with the boundary of the cache memory corresponded to the memory (we call this case as a misalignment). Additionally, we present a mathematical proof of the size of the key candidate space in general. We further present the transition of the correlation values with the number of plaintexts, which was not predicted in the previous analyses [10,13,14,15]. We obtain interesting results that differ from the results of other SCAs such as power analysis.

### 3.2 Detailed Description of Proposed Approach

An overview of the proposed method is given in Fig. 2. We describe the measured timing data and construct an ideal data based on all possible computed memory accesses. And, we calculate the correlation coefficient between the measured timing data and the ideal data.

**Description of Measured Timing Data.** We describe the measured timing data collected on the real processors. To measure the timing data effectively, after the single encryption, we access the data in a row of $D$ (in the direction of cache set in Fig. 1) corresponded in one way in the cache memory and measure the timing data in a row of $D$. Then, the timing data consists of $S$ elements that is the same with the number of cache sets. We select the timing data measured after the encryption in which the target byte of the plaintext is $p$ ($p = \texttt{0x00}, \ldots, \texttt{0xFF}$) and calculate the average of them for each $p$. We set the averaged timing data per $p$ that contain $S$ elements as $\tilde{t}(p)$ and we describe $\tilde{t}(p)$ as follows,

$$\tilde{t}(p) = (\tau_0(p), \tau_1(p), \ldots, \tau_{S-1}(p)), \tag{1}$$

where each element $\tau_j(p)$ ($j = 0, \ldots, S-1$) represents the measured timing data for a cache set. Here, we consider that the index $j$ is calculated as $j$ modulo $S$.

In general, an S-box table is allocated to a part of a cache set $S$ that corresponds to the memory. Then, the measured timing data corresponded to the S-box table calculation are a part of $\tilde{t}(p)$. We set the region of an S-box table as $l$ $(= 2^8/\delta = 2^8 \cdot L/B)$ depending on the sizes of the cache line $B$ and an S-box table element $L$. Notation $\delta$ $(= B/L)$ is the number of the S-box table elements in a cache line. Here, we assume that $l$ $(= 2^8/\delta)$ is known.[1]

As shown in the framed rectangle of $\tilde{t}(p)$ in Fig. 2, the measured timing data that correspond to the S-box table calculation are expressed as $(\tau_a(p), \ldots, \tau_{a+m-1}(p))$, where $m = l$ when $\Delta = 0$ and $m = l+1$ when $\Delta \neq 0$ ($\Delta$ is the offset from the cache memory boundary to the first element of the S-box table and the details are given in the next section). We select the unit of $l$ and $a$ as the cache line size. Notation $a$ is the offset from the start address of the cache memory corresponding to $D$ that is first allocated in the memory to the cache memory boundary including the first element of the S-box table. Because the value of $a$ depends on the S-box table allocation in the memory and it is determined by the compiled results, $a$ is an unknown value. We calculate the correlation coefficient while changing $a$ and we obtain the highest correlation value when $a$ matches the actual allocation in the memory.

In order to calculate the correlation coefficient at one time, we select the elements of the timing data, $t_a(p) = (\tau_a(p), \ldots, \tau_{a+m-1}(p))$ per $p$, and represent them as shown in the dotted arrow line in Fig. 2. We denote the timing data as

$$T_a = (t_a(\texttt{0x00}), t_a(\texttt{0x01}), \ldots, t_a(\texttt{0xFF})). \tag{2}$$

Then, we calculate the correlation coefficient between $T_a$ and the ideal data.

**Construction of Ideal Data.** The ideal data are constructed by theoretically simulating the distribution of the memory access using plaintext when we focus

[1] When we do not know this value, we can guess it by trying all acceptable $\delta$ $(= B/L)$ values. The value of $\delta$ may only accept a few patterns because the size of cache line $B$ and S-box table element $L$ accept a limited number of values based on the general cryptographic implementation and the specifications for the processors.

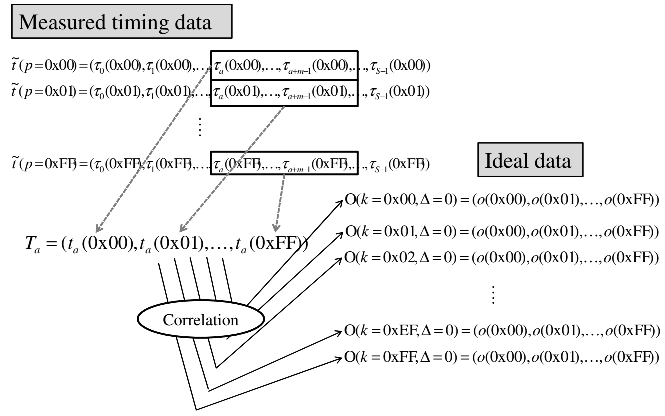


**Fig. 2.** Overview of proposed method. As an example, we show an ideal data ($\varDelta = 0$).

on an S-box table input corresponding to the target one-byte key $k$. We construct the ideal data $O(k, \varDelta)$ that can be described by $k$ and offset $\varDelta$ that represents a misalignment in the memory block. Offset $\varDelta$ $(0 \le \varDelta < \delta)$ is defined as shown in Fig. 3 and select the unit of $\varDelta$ as an element of the S-box table $L$.[2] In general, $\varDelta$ is an unknown value and we construct ideal data by changing $\varDelta$ to calculate the correlation coefficient precisely. We obtain the highest correlation value when $\varDelta$ matches the actual allocation in the memory.

The ideal data $O(k, \varDelta)$ is constructed by $o(p, k, \varDelta)$ and is expressed as:

$$O(k, \varDelta) = (o(\texttt{0x00}, k, \varDelta), o(\texttt{0x01}, k, \varDelta), \ldots, o(\texttt{0xFF}, k, \varDelta)), \tag{3}$$
$$o(p, k, \varDelta) = (\omega_0(p, k, \varDelta), \omega_1(p, k, \varDelta), \ldots, \omega_{m-1}(p, k, \varDelta)),$$

where $p$ is the target byte of plaintext (from 0x00 to 0xFF), $m = l$ when $\varDelta = 0$ and $m = l + 1$ when $\varDelta \neq 0$. Each element, $\omega_i(p, k, \varDelta)$ $(i = 0, \ldots, m - 1)$, is described as 1 when the access timing data are slow, or as 0 when the access timing data are fast.

Here, we describe how to construct $o(p, k, \varDelta)$, that is, $\omega_i(p, k, \varDelta)$ by considering the location in the cache memory corresponded to the memory, where the S-box tables are used in encryption. When the S-box tables are used in encryption, the access timing data after the encryption are slow and described as $\omega_i(p, k, \varDelta) = 1$ $(i \in \{0, \ldots, m - 1\})$. The S-box table elements used in encryption are located at the $\lfloor \frac{\varDelta + (p \oplus k)}{\delta} \rfloor$-th cache line $(\lfloor \frac{\varDelta + (p \oplus k)}{\delta} \rfloor = 0, 1, 2, \ldots)$ from the start address of the cache memory including the first element of the S-box table in the cache memory. Thus, the access timing data at this location are set as $\omega_i(p, k, \varDelta) = 1$. Then, the access timing data at other locations are fast and are set as $\omega_i(p, k, \varDelta) = 0$ $(i \in \{0, \ldots, m - 1\})$. Notation $\omega_i(p, k, \varDelta)$ can be set as

[2] We consider that an S-box table element is mapped into the same cache line even when an S-box table element consists of multiple bytes because the operating system or compiler usually assigns an S-box table element to the same cache line.

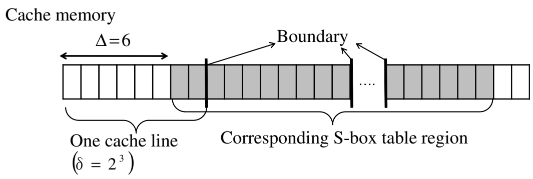


**Fig. 3.** Schematic of cache memory with offset $\varDelta$

described above and $o(p, k, \varDelta)$ is constructed using these values. As an example, when $p = \texttt{0x05}$, $k = \texttt{0x80}$, $\varDelta = 2$, $\delta = 2^4$, and $\lfloor \frac{\varDelta + (p \oplus k)}{\delta} \rfloor = 8$, $o(p, k, \varDelta)$ is

$$o(\texttt{0x05}, \texttt{0x80}, 2) = (0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, \ldots, 0, 0). \tag{4}$$

We generate $2^8 \cdot \delta$ patterns of an ideal data $O(k, \varDelta)$ by changing key hypothesis $k(= 2^8)$ and offset $\varDelta$ (the number of $\varDelta$ is $\delta$).

**Calculation of Correlation Coefficient.** The correlation coefficient is commonly used to mathematically measure a relationship between two data [19]. We employ this method in order to examine mathematically the statistical dependence between $T_a$ and $O(k, \varDelta)$. We calculate the correlation coefficient $R(a, k, \varDelta)$ by changing $a$, $k$ and $\varDelta$,

$$R(a, k, \varDelta) = \frac{\sum_{p=0}^{255} \sum_{i=0}^{m-1} (\tau_{(a+i)}(p) - \bar{\tau})(\omega_i(p, k, \varDelta) - \bar{\omega})}{\sqrt{\sum_{p=0}^{255} \sum_{i=0}^{m-1} (\tau_{(a+i)}(p) - \bar{\tau})^2} \sqrt{\sum_{p=0}^{255} \sum_{i=0}^{m-1} (\omega_i(p, k, \varDelta) - \bar{\omega})^2}}, \tag{5}$$

where $\bar{\tau}$ or $\bar{\omega}$ is an arithmetic average of $\tau_i(p)$ or $\omega_i(p, k, \varDelta)$. We determine $k$ as the correct key when $R(a, k, \varDelta)$ is the highest value among other key hypotheses.

## 4 Experimental Results

In this section, we describe the experimental results when examining the correlation between the access timing data and the key using the proposed approach.

### 4.1 Experimental Conditions

We describe the experimental conditions.

- We target an AES encryption with a 128-bit secret key implemented in C code of OpenSSL 1.0.1e [20]. We select the C code without countermeasures against CAs. This code consists of four kinds of S-box tables (referred to as T-tables) in the data part of the encryption, and each element of a T-table is 4 bytes and the total size is 1024 ($= 256 \cdot 4$) bytes.
- We try to deduce the first byte of the first round key of AES. We set the value to `0x96`.
- The plaintext is known value and is randomly set.
- We measure the access timing for the L1 data cache memory.

**Table 1.** Processors Used in Experiments

| Processor | Number of Processor Cores | Number of Cache Sets ($S$) | Cache Line Size ($B$) | Way ($W$) |
|---|---|---|---|---|
| Intel Core Solo | 1 | 64 | 64 bytes | 8-way set associative |
| Intel Core i7 620M | 2† | 64 | 64 bytes | 8-way set associative |

† Four logical cores. All of the processor cores have their own cache.

## 4.2 Experimental Results

We apply the proposed approach described in Section 3.2 using the actual access timing data. Hereafter, we target the first byte of the key; however, we can obtain all 16 bytes of the AES key by repeatedly performing the same procedures. Table 1 gives the target processors used in the experiments.[3] Each CPU has 32 KB of L1 data cache memory. We perform the timing measurements on a Core Solo processor which operating system is Windows XP (32 bits) and on a Core i7 processor which operating system is Windows 7 (64 bits). We use Microsoft Visual C++ 2008 on Core Solo processor and Microsoft Visual C++ 2010 on Core i7 processor to compile source code.[4]

**Timing Data Measurements.** We measure the access timing data after a single AES encryption using Prime+Probe method and repeat the procedure of Prime+Probe method 2400 times. When we measure the timing data, we use the standard sequence of `cpuid`, `rdtsc`, */* access the data in area of the memory to which we first allocate */*, `cpuid`, `rdtsc`, so that the RDTSC instruction is sequentially-processed. To measure the timing data effectively, after an encryption, we access the data in a row of $D$ which we first allocate (see in Fig. 1). Furthermore, we consider the access timing data that exceed the average timing data as noise and eliminate them from the analysis of the key.

**Analysis of the Key Using the Proposed Method.** Before calculating the correlation coefficients, we perform the following steps. First, we collect the timing data which the first byte of the plaintext is the same and calculate the average of them per byte. Second, we try to remove the effects other than that from the encryption such as the background noise of the operating systems to obtain high correlation coefficients. Details are given in Appendix B. Third, we normalize the average timing data, i.e., the slowest timing data are set to 1 and the fastest timing data are set to 0. Finally, we generate $2^{12}$ $(= 2^8 \cdot 2^4)$ patterns ($\delta$ is $2^4$ in both processors) of the ideal data $O(k, \Delta)$. Then, we calculate the correlation coefficients between the measured timing data and the ideal data. We implement the codes to calculate the above using MATLAB software (R2011b).

[3] In the Core i7 620M processor, the instruction set of AES (AES-NI) can be used; however, we do not use this instruction in the experiment to verify the effect of the cache memory.

[4] We compile the code of the timing measurements without optimization.

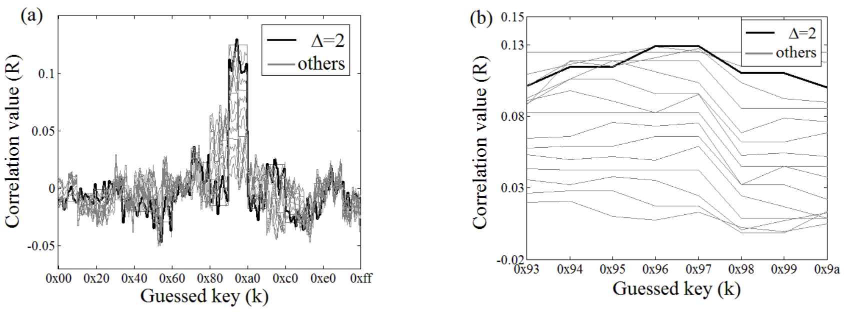


**Fig. 4.** (a) Correlation coefficients on a Core Solo processor. Key hypotheses are represented on the horizontal axis and the correlation coefficients are represented on the vertical axis. (b) Enlarged figure of (a) around $k =$ `0x96`.

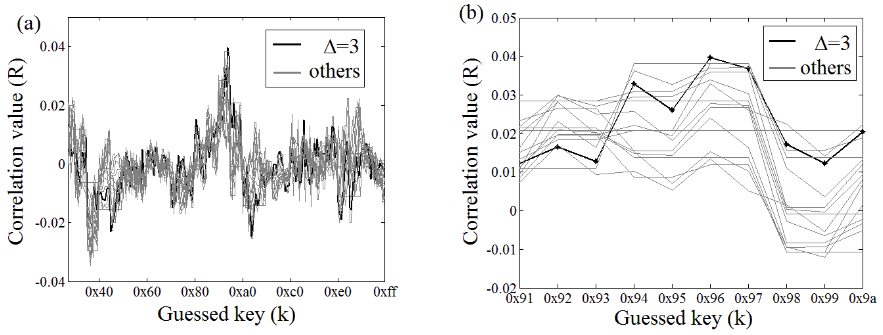


**Fig. 5.** (a) Correlation coefficients on a Core i7 processor. The horizontal and vertical axes are the same as Fig. 4. (b) Enlarged figure of (a) around $k =$ `0x96`.

***Results on Calculating Correlation Coefficient.*** Hereafter, we show the examples of the experimental results on both processors. Fig. 4 (a) shows the results of the correlation coefficient on a Core Solo processor when we set $a = 38$. In Fig. 4 (b), an extended view is presented with the correlation coefficients around $k =$ `0x96`. These figures show the maximum values at $k =$ `0x96` and $k =$ `0x97` among all key hypotheses when $\Delta = 2$. Thus, we can reduce the size of the key candidate space from $2^8$ to $2^1$. Fig. 5 (a) shows the results on a Core i7 processor when we set $a = 48$. In Fig. 5 (b), an extended view is presented around $k =$ `0x96` and show the maximum value only at $k =$ `0x96` when $\Delta = 3$. Thus, we can uniquely determine the correct key. In both cases, we can calculate the maximum value within a few minutes, which is a practical computational time. In the above, we can distinguish the correct key when $\Delta = 2$ and $\Delta = 3$. In the case of other values of $\Delta$, we can also obtain the correct results.

We note that the experimental results show that the key candidate space is reduced to less than $\delta$. Thus, the consideration of the effect of $\Delta$ gives the precise

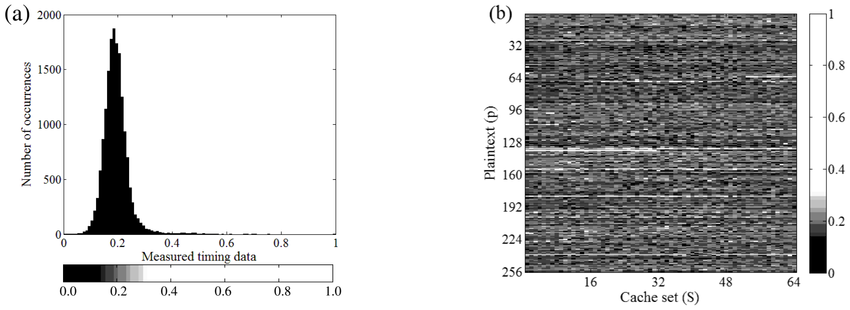


**Fig. 6.** (a) Histogram of the access timing data. Total number of samples is $2^{14}$ (the number of $k$ · the number of $S$). The horizontal axis is the access timing data and the vertical axis is the number of occurrences. Based on the histogram, we use the color bar shown below the figure. (b) Picture display of the access timing data. The horizontal axis is the cache set number and the vertical axis is the first byte of the plaintext. We cannot discern a visible diagonal line by appropriately changing the color bar setting.

results of the key candidate space. In Section 5.1, we examine in detail how much the size of the key candidate space can be reduced depending on $\Delta$.

***Results on Histogram and Picture Analysis for Reference.*** For reference, Fig. 6 (a) shows a histogram of the same timing data as that used in Fig. 5 measured on a Core i7 processor and Fig. 6 (b) shows the same timing data when we use a previous picture display. Based on the distribution of the timing data in the histogram, we appropriately set the color bar shown below the figure to see the figure clearly. In this case, even when we appropriately change the color bar setting, we cannot discern a visible diagonal in the picture display. Then, it is difficult to examine whether or not the access timing data are correlated with the correct key in such an environment using the previous approach.

## 5 Discussion

In this section, we precisely examine the key candidate space and we show that the proposed approach can quantitatively examine the relationship between the correlation values and the number of plaintexts.

### 5.1 Evaluation of Size of Key Candidate Space

We mathematically show the key candidate space can be reduced to $2^{\mathrm{ntz}(\Delta)}$ depending on the offset $\Delta$, where $\mathrm{ntz}(\cdot)$ is the number of trailing zeros and it represents the number of sequences of 0s from the least significant bit in the binary representation. In [13] and [14], the authors claimed that the number of recoverable key bits of a one-byte key was limited by the number of S-box tables that are allocated to one cache line. In [16] and [21], the authors claimed that the size of the key candidate space could be reduced to less than $\delta$ when the S-box table was not aligned to the multiple of $B$, the size of the cache line; however, it was not clear to what degree size of the key candidate space could be reduced in general. Later, [17] showed the reduction of the key candidate space in all cases of the misalignment when $\delta = 16$; however, they did not show the verification or the proof of the size of the key candidate space based on the misalignment. In the following discussion, we show the size of the key candidate space in general depending on the offset $\Delta$ and give a mathematical proof of it.

Here, we assume that the size of one cache line is an integral multiple of an S-box table element. This assumption is applicable to an architecture of general processor and the cryptographic implementation. We also assume that $\delta$ is the power-of-two. We focus on the $(\delta - \Delta)$-elements of the S-box table from the first element, which are allocated in the same cache line. When $\Delta$ is constant, we can calculate the ideal data $O(k, \Delta)$ for all $k$ because the number of $k$s is the same with the number of the index of the S-box table. At this time, there are some keys $k$ that accept the same ideal data, but the values of $k$ are different. Then, the number of the ideal data patterns reduces to less than $2^8$ for one-byte key.

In the following discussion, we show that the key candidate space can be reduced to $2^{\mathrm{ntz}(\Delta)}$. As an example, when $\Delta$ is odd, $\mathrm{ntz}(\Delta) = 0$, then, the key candidate space is $2^0$ $(= 1)$. We prove the following lemma to show that the key candidate space is reduced to $2^{\mathrm{ntz}(\Delta)}$ (detail proof is given in Appendix C).

**Lemma 1.** *For arbitrary key values $k_a$ and $k_b$ that satisfy $k_a \& (\overline{2^{\mathrm{ntz}(\Delta)} - 1}) \neq k_b \& (\overline{2^{\mathrm{ntz}(\Delta)} - 1})$, there exists input $p$ such that $p \oplus k_a < \Delta \leq p \oplus k_b$, where $\overline{x}$ is the bitwise complement of $x$.*

In AES as an example, $k_a$ and $k_b$ are one-byte keys and $p$ is a one-byte plaintext. When $\Delta$ is odd, the lemma states that $p \oplus k_a$ and $p \oplus k_b$ $(k_a \neq k_b)$ do not exist in the same cache line with appropriate selection of $p$. Then, we can uniquely determine a one-byte key. The key candidate space is represented as $2^{\mathrm{ntz}(\Delta)}$. When $\Delta$ is even, the lemma states that the high $(8 - \mathrm{ntz}(\Delta))$ bits of $k_a$ and $k_b$ can be distinguished and the key candidate space is equal to $2^{\mathrm{ntz}(\Delta)}$ (the detailed proof is given in Appendix D). Therefore, this proves that the key candidate space is $2^{\mathrm{ntz}(\Delta)}$ for $0 \leq \Delta < \delta$.

## 5.2 Significance of S-box Table Alignment for Implementation

We claim that it is better to align the S-box table in the memory. When the S-box table is aligned, the key candidate space for 16 bytes will remain to $(2^4)^{16} = 2^{64}$ at least when $\delta = 2^4$ and the brute-force search is not practical. However, when the S-box table is not aligned, the key candidate space can further be reduced shown in Section 5.1 and we can uniquely determine a 16-byte key especially in the case that $\Delta$ is odd. Thus, for the cryptographic implementation, the S-box table alignment is significant. To prevent the misalignment, as an example, when we use an assembly language for the implementation, it is recommended that an alignment macro is inserted in the code. Using the C code, it is recommended that the “align” attribute is added to the declaration of the S-box table.

### 5.3 Relationship Between Correlation and Number of Plaintexts

We quantitatively examine the relationship between the correlation values and the number of plaintexts. In power analysis (PA), the transition of the correlation values with the number of plaintexts is essential to examine the required number of plaintexts for a successful attack [19]. In CAs, the expected number of plaintexts to deduce the correct key was theoretically estimated in [10] and [15]; however, there is no method and results to experimentally evaluate the transition in the correlation values with the number of plaintexts to examine the required number of plaintexts. In the following discussion, we quantitatively examine the correlation coefficients, while increasing the number of plaintexts.

Fig. 7 shows examples of correlation coefficients on a Core Solo and a Core i7 processors with the number of plaintexts. In the experiments, we select the first byte of the plaintexts to uniformly appear from `0x00` to `0xFF`, and other 15 bytes are randomly set. The total number of plaintexts is calculated as $256 \cdot$ (the number of vertical lines). From the figures, we can distinguish the highest correlation values while increasing the number of plaintexts and we can find the correct hypothesis. This feature is very similar to that of PA [19].

In PA, it is well-known that the correlation coefficient converges to a constant values with increasing the number of plaintexts [19]. On the contrary, in Fig. 7, the correlation values do not converge to a constant value. As shown in (I) of Fig. 7 (a) and (I) of Fig. 7 (b), the correlation values increase with more plaintexts even when we use the wrong key hypotheses; however the correlation values are not higher than that of the correct key hypothesis. As described in Section 5.1, in CAs, multiple elements of the S-box table are stored in one cache line. Then, more of the same cache lines are accessed when we use the correct key and some wrong keys. That is, the cache access patterns are almost the same and the correlation values increase with more plaintexts even when we assume the wrong keys, which have some low-bit differences from the correct key. Thus, the correlation values increase even when we guess the wrong keys.

From figures, we can find the required number of plaintexts to achieve a successful attack. In Fig. 7 (a), $256 \cdot 2$ $(= 2^9)$ plaintexts are needed and we reduce the size of the key candidate space to $2^1$ when $\Delta = 2$ represented as the plots in black in the figure. Similarly, in Fig. 7 (b), $256 \cdot 17$ $(\approx 2^{12})$ plaintexts are needed and we reduce the size of the key candidate space to $2^2$ when $\Delta = 4$.

Therefore, using the proposed method, we quantitatively evaluate the number of plaintexts in CAs even when it is difficult to do using the previous approach.

## 6 Conclusions

This paper proposed a key extraction method that quantitatively examines the correlation between the access timing data and the key. We showed that a mathematical correlation method can be utilized to evaluate quantitatively the access-driven CAs. To the best of our knowledge, this was the first study to examine the key candidate space precisely and mathematically by considering the effect of the S-box table allocation in the memory and show the relationship between the

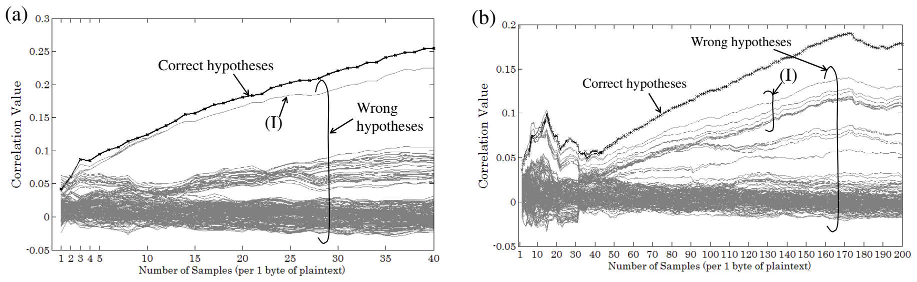


**Fig. 7.** Correlation values for 256 key hypotheses on a Core Solo CPU (a) and on a Core i7 CPU (b). Key hypotheses including correct key are plotted in black, while other key hypotheses are plotted in gray. The horizontal line represents the number of appearances for each byte and the vertical line represents correlation values.

correlation values and the number of plaintexts. Using the proposed approach, we could examine the correlation correctly in a noisy environment even when it was difficult to do using a previous picture analysis. The experimental results on two kinds of processors confirmed the effectiveness of the proposed approach. We first showed that the key candidate space can be reduced to $2^{\mathrm{ntz}(\Delta)}$ with the mathematical proof, where $\mathrm{ntz}(\cdot)$ is the number of trailing zeros. And, we showed that we could quantitatively analyze the number of plaintexts. We believe that the proposed approach contributes to the development of a key analysis for CAs and this approach can be applied in evaluating on other cryptographic primitives against CAs. In the future, we study the boundary condition exactly where this approach would work well based on the amount of noise from other operations.

**Acknowledgments.** The authors would like to thank Fumitaka Hoshino, Hikaru Sakamoto and Prof. Kazuo Sakiyama for giving us valuable comments on this paper. The authors also thank anonymous reviewers for suggestions to improve the quality of this paper.

## References

1. Takahashi, J., Sakamoto, H., Fukunaga, T., Fuji, H., Sakiyama, K.: Automatic Evaluation Method of Access-Driven Cache Attack. In: The 29th Symposium on Cryptography and Information Security (SCIS 2012), p. 2C2-2, 7 pages (2012) (in Japanese)
2. Takahashi, J., Fukunaga, T.: Analysis on Number of Plaintexts for Cache Attacks Using Highly Accurate Key Extraction Method. In: The 30th Symposium on Cryptography and Information Security (SCIS 2013), p. 3E3-3, 8 pages (2013) (in Japanese)
3. Kocher, P.C.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996)

4. Kelsey, J., Schneier, B., Wagner, D., Hall, C.: Side channel cryptanalysis of product ciphers. In: Quisquater, J.-J., Deswarte, Y., Meadows, C., Gollmann, D. (eds.) ESORICS 1998. LNCS, vol. 1485, pp. 97–110. Springer, Heidelberg (1998)
5. Page, D.: Theoretical Use of Cache Memory as a Cryptanalytic Side-Channel. Technical Report CSTR-02-003, Department of Computer Science, University of Bristol (2002)
6. Page, D.: Defending against cache based side-channel attacks. Information Security Technical Report 8(1), 30–44 (2003)
7. Tsunoo, Y., Tsujihara, E., Minematsu, K., Miyauchi, H.: Cryptanalysis of Block Ciphers Implemented on Computers with Cache. In: Proc of ISITA 2002 (2002)
8. Tsunoo, Y., Saito, T., Suzaki, T., Shigeri, M., Miyauchi, H.: Cryptanalysis of DES Implemented on Computers with Cache. In: Walter, C.D., Koç, Ç.K., Paar, C. (eds.) CHES 2003. LNCS, vol. 2779, pp. 62–76. Springer, Heidelberg (2003)
9. Bernstein, D.J.: Cache Timing Attacks on AES (April 2005), `http://cr.yp.to/antiforgery/cachetiming-20050414.pdf`
10. Tiri, K., Acıiçmez, O., Neve, M., Andersen, F.: An Analytical Model for Time-Driven Cache Attacks. In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 399–413. Springer, Heidelberg (2007)
11. Acıiçmez, O., Koç, Ç.K.: Trace-Driven Cache Attacks on AES (Short Paper). In: Ning, P., Qing, S., Li, N. (eds.) ICICS 2006. LNCS, vol. 4307, pp. 112–121. Springer, Heidelberg (2006)
12. Bertoni, G., Zaccaria, V., Breveglieri, L., Monchiero, M., Palermo, G.: AES Power Attack Based on Induced Cache Miss and Countermeasure. In: ITCC 2005, vol. 1, pp. 586–591. IEEE Computer Society (2005)
13. Tromer, E., Osvik, D.A., Shamir, A.: Efficient cache attacks on AES, and countermeasures. Journal of Cryptology 23(1), 37–71 (2010)
14. Osvik, D.A., Shamir, A., Tromer, E.: Cache attacks and countermeasures: The case of AES. In: Pointcheval, D. (ed.) CT-RSA 2006. LNCS, vol. 3860, pp. 1–20. Springer, Heidelberg (2006)
15. Neve, M., Seifert, J.-P.: Advances on Access-Driven Cache Attacks on AES. In: Biham, E., Youssef, A.M. (eds.) SAC 2006. LNCS, vol. 4356, pp. 147–162. Springer, Heidelberg (2007)
16. Xinjie, Z., Tao, W.: Dong, Mi., Yuanyuan, Z., Zhaoyang, L.: Robust First Two Rounds Access Driven Cache Timing Attack on AES. In: CSSE 2008, pp. 785–788. IEEE Computer Society (2008)
17. Spreitzer, R., Plos, T.: Cache-Access Pattern Attack on Disaligned AES T-Tables. Pre-Proceedings of the Fourth International Workshop on Constructive Side-Channel Analysis and Secure Design, COSADE 2013 (2013)
18. Ristenpart, T., Tromer, E., Shacham, H., Savage, S.: Hey, You, Get Off of My Cloud: Exploring Information Leakage in Third-Party Compute Clouds. In: ACM CCS 2009, pp. 199–212 (2009)
19. Mangard, S., Oswald, E., Popp, T.: Power Analysis Attacks -Revealing the Secret of Smart Cards. Springer-Verlag New York Inc. (C); ISBN: 978-0-387-30857-9
20. OpenSSL, Cryptography and SSL/TLS Toolkit, `http://www.openssl.org/`
21. Xinjie, Z., Tao, W.: Improved Cache Trace Attack on AES and CLEFIA by Considering Cache Miss and S-box Misalignment. IACR Cryptology ePrint Archive 2010/056 (2010)

## Appendices

## A Cache Memory Organization

Cache memory is small high-speed memory that contains the most recently accessed piece of the main memory. In Fig. 8, a set associative cache memory and the memory organizations are shown. In general, the cache memory is broken into a *cache line* comprising $B$ bytes and the size of a cache line is determined by both the processor and the cache design. The access to the cache memory is performed per the size of the cache line. Cache memory is classified as groups of row of cache lines that is referred to a *cache set* $S$, each containing $W$ cache lines. Example values of $B$, $S$, and $W$ on the processors are shown in Table 1. Each memory block is cached only in a specific cache set shown as a double headed outlined arrow in Fig. 8.[5] Memory block starting address $d$ can be cached only in the $W$ cache lines belonging to a cache set $\lfloor d/B \rfloor$ mod$S$ and replaces the most previous content in $W$ cache lines. As an example, when the S-box table is allocated in the memory in Fig. 8, each element of it is cached in the corresponding cache sets in Fig. 8 when a cache miss occurs.

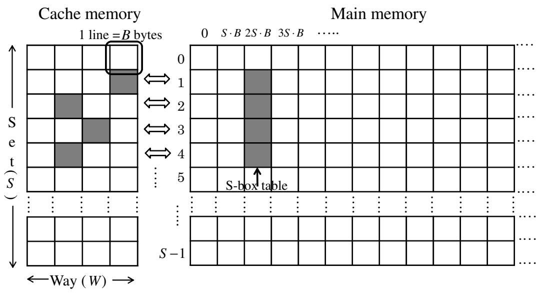


**Fig. 8.** Schematic of memory organizations. Each row of memory blocks is mapped to the corresponding row in the cache memory as indicated by a double headed outlined arrows. The gray color represents S-box tables in the encryption. The memory address is represented as the values in the vertical and horizontal lines in the main memory.

## B Noise Reduction of Timing Data

In the experiments in Section 4.2, we try to remove the effect of other operations that are independent of encryption. As an example, we subtract the average timing data from each set of measured access timing data [13], to remove the effect of other operations, $\tau_i(p) - \bar{\tau}_i$ where $\tau_i(p)$ ($p = \texttt{0x00}, \ldots, \texttt{0xFF}$, $i = 0, \ldots, S-1$) is the original timing data and $\bar{\tau}_i$ is the average timing data per cache set.

[5] The unit of the memory block corresponds to the cache line size.

## C Proof of Key Candidate Space Depending on S-Box Table Allocation in Memory (Proof of Lemma 1)

We prove the lemma in Section 5.1. Without loss of generality, assume $k_a = 0$ and $k_b = k2^{\mathrm{ntz}(\Delta)}$ where $k \neq 0$ (we can represent other values if we calculate the XOR values for $k_a$ and set the arbitrary $k$ ($k = 1, 2, 3, \ldots$) for $k_b$). We let $\lfloor p/2^{\mathrm{ntz}(\Delta)} \rfloor \to p'$ and $\Delta/2^{\mathrm{ntz}(\Delta)} \to \Delta'$. Then, we prove the following.

*For arbitrary $k$ ($k \neq 0$), there exists $p'$ such that $p' < \Delta' \leq p' \oplus k$.*
At this time, we derive $0 < \Delta'$ and $\Delta'$ is odd from the definition.

Case 1: $\Delta' \leq k$,
 Select $p' = 0$. Clearly, $0 < \Delta'$ and $\Delta' \leq 0 \oplus k$ from the assumption.
Case 2: $k \oplus \Delta' < \Delta'$,
 Select $p' = k \oplus \Delta'$. Clearly, $p' = k \oplus \Delta' < \Delta'$ from the assumption. In addition, $\Delta' = p' \oplus k$ satisfies.
Case 3: Otherwise,
 Select $p' = \Delta' \oplus 1$. As $\Delta'$ is odd, $p' = \Delta' \oplus 1 < \Delta'$.
 In addition, calculate,

$$\begin{aligned} p' \oplus k &= \Delta' \oplus k \oplus 1 \\ &= \begin{cases} (\Delta' \oplus k) + 1 & (k \text{ is odd}) \\ (\Delta' \oplus k) - 1 & (k \text{ is even}) \end{cases} \\ &\geq (\Delta' \oplus k) - 1 > \Delta' - 1 \end{aligned}$$

 From $\Delta' - 1 \leq (\Delta' \oplus k) - 1$, then, $\Delta' \leq (\Delta' \oplus k) - 1$. Thus, $\Delta' \leq p' \oplus k$. □

## D Proof That Key Candidate Space Equals $2^{\mathrm{ntz}(\Delta)}$ When $\Delta$ Is Even

We prove that the key candidate space is $2^{\mathrm{ntz}(\Delta)}$, i.e., the low ($\mathrm{ntz}(\Delta)$) bits cannot be distinguished when $\Delta$ is even. Without loss of generality, assume $k_a = 0$ and $k_b = k_a \oplus \tilde{k}$ ($= 0 \oplus \tilde{k}$), $\tilde{k} = \{1, 2, \ldots, 2^{\mathrm{ntz}(\Delta)} - 1\}$. From $2^{\mathrm{ntz}(\Delta)} \leq \Delta$, $\tilde{k} \leq \Delta$. Then, $k_b \leq \Delta$, and $p \oplus k_a$ and $p \oplus k_b$ are within $\Delta$ in the same cache line, i.e., the low ($\mathrm{ntz}(\Delta)$) bits are not distinguished. Thus, the key candidate space is $2^{\mathrm{ntz}(\Delta)}$. □

# Upper Bounds for the Security of Several Feistel Networks

Yosuke Todo

NTT Corporation, Japan
todo.yosuke@lab.ntt.co.jp

**Abstract.** In this paper, we are dealing with upper bounds for the security of some Feistel networks. Such a topic has been discussed since the introduction of Luby-Rackoff construction, but it is unrealistic construction because its round functions must be chosen at random from the set of all functions. Knudsen dealt with more practical construction where its round functions are chosen at random from a family of $2^k$ randomly chosen functions, and showed an upper bound for the security by demonstrating generic key recovery attacks. However it is still difficult for designers to choose functions randomly. Then, this paper considers the security of some Feistel networks which have more efficient and practical round functions and are indeed used by some Feistel ciphers in practice. For this Feistel ciphers, we discover new properties using the relation of plaintexts and ciphertexts. By using our properties, we propose new generic key recovery attacks, and confirm the feasibility by implementing the attack for small block sizes. Our results indicate that the 6 round networks are not enough to complicate the relationship between plaintexts and ciphertexts, and how to insert a round key is very influential in the upper bound for the security. This feature should be taken into account when the round function is designed in future. Moreover, for immunity to our attacks and maintenance of the efficiency, we show design principles for efficient and secure Feistel ciphers.

**Keywords:** Block cipher, Feistel networks, Round functions, Key recovery attacks.

## 1 Introduction

By using Feistel networks, we can construct $n$-bit pseudo-random permutations from $n/2$-bit pseudo-random functions effectively. Luby and Rackoff [8] proved

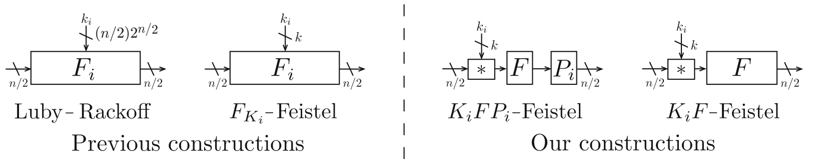


**Fig. 1.** Some Constructions for the round function

that 3 and 4 round Feistel networks are sufficient to make a pseudo-random permutation and a super pseudo-random permutation, respectively, when the round functions are pseudo-random functions. Since then, many results for pseudo-randomness on Feistel networks have been proposed [11,9,12,14,6]. Luby and Rackoff also showed that attackers can distinguish the 3 round Feistel network from random permutations by using an adaptive chosen plaintext and ciphertext attack with 3 texts. Since then, many generic attacks on Feistel networks have been proposed [4,13,5,15]. If the round functions are pseudo-random, there exists the distinguishing attack on the 5 round Feistel network by using a chosen plaintext attack (CPA) with $\mathcal{O}(2^{3n/4})$ texts [13]. Moreover, if the round functions are pseudo-random permutations, there exists the distinguishing attack on the 5 round Feistel network by using CPA with $\mathcal{O}(2^{n/2})$ texts [4].

Luby-Rackoff construction has the round functions which are chosen at random from a family of $2^{\frac{n}{2}\times 2^{n/2}}$ randomly chosen functions (see the leftmost of Fig.1). This means the key size for the $r$-round Feistel network is $r \times \frac{n}{2}2^{n/2}$-bit, and it is unrealistic for real Feistel ciphers to have the enormous key size. Then, Knudsen introduced more practical Feistel network where the round functions are chosen at random from a family of $2^k$ randomly chosen functions, and we call this network the $F_{K_i}$-Feistel (see the second from the left of Fig.1). In this Feistel network, attackers can search for one round function by an exhaustive search over all $2^k$ possible round functions. Moreover, Knudsen proposed the key recovery attack on 5 and 6 round Feistel networks by using CPA. Time complexity of these attacks is $2^{k+\frac{n+6}{4}}$ and $2^{k+n/2+1}$, respectively, but unfortunately the Knudsen's attack for 6 rounds is as expensive as the brute force attack when the round key size is the same as the input size of round function ($k = n/2$-bit) and the master key size is the same as the block length ($n$-bit).

Now, we revisit the Knudsen's Feistel network where the round functions are defined as $2^k$ random functions or random permutations. To implement Knudsen's Feistel network, designers must design ideal compression functions which have $k + n/2$-bit input and $n/2$-bit output. However, it is difficult to design such ideal compression functions. On the other hand, the ideal permutations seem to be researched better than the ideal compression functions. Then we introduce more efficient and simple Feistel networks where the round function consists of ideal permutations not compression functions. In our Feistel networks, we split the round function into "the key insert operation $*$" and "the nonlinear bijective function $F$," where the operation $*$ is a function to mix an input with a round key by a simple operation, for example, we deal with an exclusive OR ($\oplus$) and a modular addition ($\boxplus$) as the operation $*$. In the function $F$, a mixed data of an input and a round key is confused and is diffused.

Next we consider how to construct the function $F$. In the secure construction, we use different functions $F$ in every round. However it is not efficient because it must have $r$ ideal bijective functions with $r$ rounds. Then, in this paper, we consider two practical constructions. The function $F$ of many Feistel ciphers consists of the confusion and the diffusion, and it is more difficult to design the confusion than to design the diffusion. In the first construction, we only use the

**Table 1.** Summary of attacks on several Feistel networks whose round key size is $n/2$-bit

| Target | 5 rounds | | | 6 rounds | | | Reference |
|---|---|---|---|---|---|---|---|
| | Time | Number of texts | | Time | Number of texts | | |
| $F_{K_i}$ | $2^{3n/4+3/2}$ | $\sqrt{n}2^{n/4}$ | CPA | $2^{n+1}$ | $\frac{n}{2}2^{n/2}$ | CPA | [4] |
| $K_iF(\oplus)$ | $2^{n/2+3}$ | $2^{n/2+2}$ | KPA | $\frac{n}{2}2^{n/2}$ | $\frac{n}{2}2^{n/2}$ | CPA | Ours |
| $K_iFP_i(\oplus)$ | $t2^{n/2}$ | $t2^{n/2}$ | KPA | $2^{7n/8+1}$ | $2^{7n/8}$ | CPA | Ours |
| $K_iF(\boxplus)$ | $2^{n/2+3}$ | $2^{n/2+2}$ | KPA | - | - | | Ours |

same confusion function $F$, and use different functions for the diffusion in every round. This method is called the diffusion switching mechanism [16], which is an effective mechanism for design of practical and secure block ciphers. We call this network the $K_iFP_i$-Feistel (see the third from the left of Fig.1). In the second construction, we consider more efficient construction than the $K_iFP_i$-Feistel, namely we only use the same function $F$, do not use a diffusion. We call this network the $K_iF$-Feistel (see the rightmost of Fig.1). This is very efficient, for example, Camellia [1], SEED [7] and GOST cipher [10] have this construction.

In this paper, we first show the upper bound for the security of the 5 round $K_iF$-Feistel. For the 5 round $K_iF$-Feistel, we can know a plaintext and the ciphertext by querying plaintext, but we can not know the internal states. However we show that there exists the special relationship between plaintext and ciphertext, if an internal state of the encryption circuit has the specific property. Therefore we can distinguish whether the text has the specific property by observing the relationship between plaintext and ciphertext. By using the distinguisher, we propose a new key recovery attack which is similar to the all subkeys recovery attack [3]. The time complexity is $\mathcal{O}(2^{n/2})$ and the attack model is a known plaintext attack (KPA) not a chosen plaintext attack (CPA). Moreover we show the upper bound for the security of the 6 round $K_iF$-Feistel. Unfortunately, for the 6 round $K_iF$-Feistel, we can not distinguish whether the text has the specific property, because we can not know the value of $R_6$ which is ciphertext for the 5 round $K_iF$-Feistel. Then we aim to separate $2^n$ all texts into several sets, and one of the separated sets includes exactly one text satisfying this property. If the key insert operation $*$ is XOR, we can separate into such sets efficiently in a CPA. By analyzing in every set, we propose a new key recovery attack on the 6 round $K_iF$-Feistel, and the time complexity is only $\mathcal{O}(k2^{n/2})$.

Next, we show the upper bound for the security of the 5 round $K_iFP_i$-Feistel. In the 5 round $K_iFP_i$-Feistel, we can not distinguish exactly whether the text has the specific property by observing the relationship between plaintext and ciphertext, because different round functions are used in every round. However if the key insert operation is XOR, we can distinguish it probabilistically. By using the probabilistic distinguisher, we propose a new key recovery attack on the 5 round $K_iFP_i$-Feistel. The time complexity is $\mathcal{O}(2^{n/2})$ and the attack model is a KPA not a CPA. Moreover, we show the upper bound for the security of the 6 round $K_iFP_i$-Feistel. For the 6 round $K_iFP_i$-Feistel, we decrypt one round by exhaustive search over all $2^{n/2}$ round keys, and use the property for the 5 round $K_iFP_i$-Feistel. However the bias of the 5 round property is very small to recover round key efficiently. Then we propose a new technique called the chosen texts technique. By using this technique, we can choose a subset of plaintexts whose members have more bias than that of original. By using this chosen subset, we propose a new key recovery attack on the 6 round $K_iFP_i$-Feistel. The time complexity is $\mathcal{O}(2^{7n/8})$ and the attack model is a CPA.

We summarize existing and our results in Table 1, where $t$ is function of $n$ and see Section 4.1 in detail. From Table 1, we can conclude that how to insert round key is very influential in the upper bound for the security. Moreover we confirm the feasibility of all of our attacks by a machine experiment for small block sizes. We show the experimental report in Appendix A. Finally, we discuss our new results and show design principles for efficient and secure Feistel network. To prevent our attack with maintaining the efficiency, we show the diffusion part should use a nonlinear operation with respect to XOR.

This paper is organized as follows. Section 2 gives notations, Feistel networks evaluated in this paper, and the Knudsen's attacks. Section 3 gives new properties and key recovery attacks for the $K_iF$-Feistel, and Section 4 gives new properties and key recovery attacks for the $K_iFP_i$-Feistel. We discuss results of our attack and show design principles for efficient and secure Feistel ciphers in Section 5 and conclude this paper in Section 6.

## 2 Preliminary

### 2.1 Notations

Let $n$ be the block length for the Feistel cipher. $p$ and $c$ denote plaintexts and ciphertexts, respectively. Let $k_i$ be a round key of $i$-th round. For a value $x$, $x_L$ denotes the left half of $x$ and $x_R$ denotes the right half of $x$. In the $r$-round Feistel network, the ciphertext is calculated as $c = E_K(p,k) = Swap \circ \Psi_{k_r} \circ \cdots \circ \Psi_{k_1}(p_L, p_R)$, where $\Psi_{k_i}$ is defined as follows:

$$(L_{i+1}, R_{i+1}) = \Psi_{k_i}(L_i, R_i) = (y_i \oplus R_i, L_i) = (\psi_{k_i}(L_i) \oplus R_i, L_i),$$

where $L_i$ and $R_i$ denotes left half and right half of the $i$-th round input, and $\psi_{k_i}$ denotes round function of the $i$-th round. Moreover, let $y_i$ be an output of $i$-th round function.

In the Luby-Rackoff construction, $\psi_{k_i}$ must be chosen at random from the set of all functions in every round, and 4 rounds are sufficient to make a super pseudo-randomness. Unfortunately, it is an unrealistic construction, because the secret information of this construction is $r$ round functions $\psi_{k_1} \cdots \psi_{k_r}$ and the key size is very huge.

### 2.2 Feistel Networks

We define some Feistel networks discussed in this paper. For more practical construction than the Luby-Rackoff construction, Knudsen defined the ideal round function as follows:

**Definition 1 (Knudsen's ideal round function).**
*Let $F_k$ be an $n/2$-bit function or permutation chosen from a family of $2^k$ function. Then $F_k$ is called ideal, if exhaustively search at least $2^k$ possible function is necessary to find the correct function.*

We call this construction $F_{K_i}$-Feistel. It is more practical than the Luby-Rackoff construction, because the size of secret information is only $r \cdot k$-bit in the $F_{K_i}$-Feistel with $r$ rounds. However, to achieve the $F_{K_i}$-Feistel, designers must design ideal compression functions which have $(k + n/2)$-bit input and $n/2$-bit output, and it is not easy to design such functions. On the other hand, the ideal permutations seem to be researched better than the ideal compression functions. Then we introduce more efficient and simple Feistel networks where the round functions consist of ideal permutations not compression functions.

We define a construction called $K_iF$-Feistel which does not have an ideal compression function. In the $K_iF$-Feistel, we split a round function into "the key insert operation *" and "the ideal bijective function $F$," where the operation * is a function to mix an input with a round key by a simple operation, for example, an exclusive OR ($\oplus$) or a modular addition ($\boxplus$). In this time, we consider only when the function $F$ is bijective, because many Feistel ciphers use bijective functions for the function $F$. Moreover, there exist impossible outputs of the round function independent of the round key when the function $F$ is not bijective, and it appears that the network is not secure[1]. The $K_iF$-Feistel is used widely for design of Feistel ciphers. Examples of the construction are Camellia [1], SEED [7] and GOST cipher [10]. As an existing result for the security of the $K_iF$-Feistel, Knudsen and Rijmen proposed the known-key distinguishing attack [5].

Now we consider a more secure construction. Surely, the $K_iF$-Feistel is efficient and simple, but it appears that it is not secure because a common round function is used in every round. Then, we want to use different round functions in every round, but the construction is not efficient because it must have $r$ ideal bijective functions with $r$ rounds. Then, we add the diffusion part $P$ after the functions $F$, and apply the diffusion switching mechanism [16]. In the diffusion switching mechanism, different functions are used for diffusion in every round, and it is an effective mechanism for design of practical and secure block ciphers. In this paper, we assume the diffusion part $P$ contains only linear operation of XOR. Specifically, the multiplication by the constant over $GF(2^{n/2})$ is available to the diffusion part $P$. We express the constant value of the $i$-th round as $P_i$, and can use different constants in every round. Moreover we express the multiplicative inverse of $P_i$ as $P_i^{-1}$. We call this network $K_iFP_i$-Feistel, and it is more secure than the $K_iF$-Feistel.

## 2.3 Knudsen's Attack

Here, we show the outline of the Knudsen's attack on the $F_{K_i}$-Feistel whose round functions consist of permutations. For the detailed procedures and evaluations

[1] The round function of DES is not bijective. However, DES comes under the $F_{K_i}$-Feistel because it has the expansion permutation before the key mixing.

for the 5 round and 6 round key recovery attacks, see Sect.3.3 and Sect.3.4 in the original paper [4], respectively.

For a key recovery attack on the 5 round $F_{K_i}$-feistel, attackers use the following impossible differential [2] characteristic of the 4 round $F_{K_i}$-Feistel:

$$(0,\alpha) \xrightarrow{\Psi_1} (\alpha,0) \xrightarrow{\Psi_2} (\beta,\alpha) \xrightarrow{\Psi_3} (\gamma,\cdot) \neq (\alpha,\cdot) \xleftarrow{\Psi_4^{-1}} (\cdot,\alpha),$$

where $\alpha \neq 0$, $\beta \neq 0$ and $\gamma \neq \alpha$ because round functions are permutations. The probability that it satisfies $(0,\alpha) \rightarrow (\cdot,\alpha)$ is $2^{-n/2}$ for random permutations, but it is an impossible event for the $F_{K_i}$-Feistel. By using $\sqrt{k} \cdot 2^{n/4+1/2}$ chosen plaintexts, they can recover the correct key with a high probability. The time complexity is at most $2^{n/4+1/2}(2^k + 2^{k-1} + \cdots + 2 + 1) \approx 2^{k+\frac{n/2+3}{2}}$.

For a key recovery attack on the 6 round $F_{K_i}$-Feistel, attackers use the following impossible differential characteristic of the 5 round $F_{K_i}$-Feistel:

$$(0,\alpha) \xrightarrow{\Psi_1} (\alpha,0) \xrightarrow{\Psi_2} (\beta,\alpha) \xrightarrow{\Psi_3} (\gamma,\beta) \neq (\alpha,\cdot) \xleftarrow{\Psi_4^{-1}} (0,\alpha) \xleftarrow{\Psi_5^{-1}} (\alpha,0),$$

where $\alpha \neq 0$, $\beta \neq 0$ and $\gamma \neq \alpha$ because round functions are permutations. The probability that it satisfies $(0,\alpha) \rightarrow (\alpha,0)$ is $2^{-n}$ for random permutations, but it is an impossible event for the $F_{K_i}$-Feistel. For distinguishing attack on the 5 round $F_{K_i}$-Feistel, with $2^{n/2}$ chosen texts, the probability of success for the distinguishing attack on the 5 round $F_{K_i}$-Feistel is $1 - e^{-1/2} \approx 0.393$ because $\binom{2^{n/2}}{2}2^{-n} \approx \frac{1}{2}$. By using $k \cdot 2^{n/2}$ chosen plaintexts, they can recover the correct key with a high probability. The time complexity is at most $2^{n/2}(2^k + 2^{k-1} + \cdots + 2 + 1) \approx 2^{k+1+n/2}$.

This key recovery attack can not attack the 6 round $F_{K_i}$-Feistel where the round functions are chosen at random from a family of $2^{n/2}$ random permutations and the master key size is $n$-bit. Hereafter, we assume that the round key size $k$ is $n/2$-bit and the master key size is $n$-bit. In this setting, the time complexity of the 6 round key recovery attack is at most $2^{n+1}$ which is the same as the brute force attack. Finally we remark that the Knudsen's attack can be executed to our new constructions: the $K_iF$-Feistel and the $K_iFP_i$-Feistel, because our new constructions are subset of the $K_iF$-Feistel. Note that our new attacks explained in later sections are specialized for the new constructions and more efficient than the Knudsen's attack.

## 3 New Key Recovery Attack on the $K_iF$-Feistel

### 3.1 New Key Recovery Attack for the 5 Round $K_iF$-Feistel

First we propose a new property which uses some relationship between plaintext and ciphertext of the 5 round $K_iF$-Feistel. Based on this property, we propose a new key recovery attack for the 5 round $K_iF$-Feistel (see Fig. 2). This attack can be executed with time complexity smaller than the Knudsen's attack, and it is KPA not CPA.

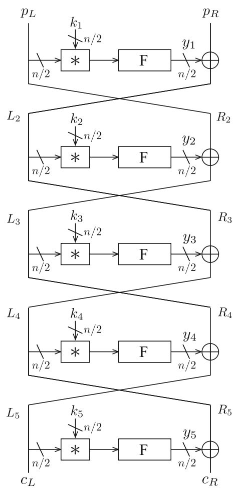


**Fig. 2.** The 5 round $K_iF$-Feistel

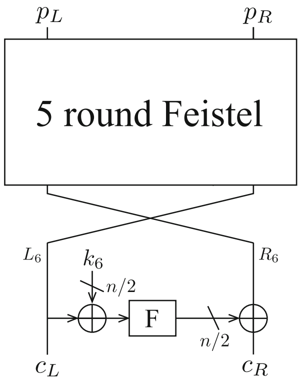


**Fig. 3.** The 6 round $K_iF$-Feistel

**Overview.** Our attack aims to detect some relationship between plaintext $(p_L, p_R)$ and ciphertext $(c_L, c_R)$. For the 5 round $K_iF$-Feistel, $c_L$ can be expressed only with $p_L, y_2$ and $y_4$ as follows:

$$c_L = p_L \oplus y_2 \oplus y_4, \tag{1}$$

and it indicates $p_L \oplus c_L = y_2 \oplus y_4$. Therefore if there exists some linear relationship between $y_2$ and $y_4$, we can observe some linear relationship between $p_L$ and $c_L$. Unfortunately, there does not exist general linear relationship between $y_2$ and $y_4$, because $y_2$ and $y_4$ are outputs of the round function which is a non-linear function. However if $y_2 = y_4$ is satisfied, it always satisfies $F^{-1}(y_2) = F^{-1}(y_4)$ and then $L_2 * k_2 = L_4 * k_4$. At the same time, $y_2 = y_4$ guarantees $p_L = c_L$ from Equation 1. Therefore we can choose texts which satisfy $L_2 * k_2 = L_4 * k_4$ by observing $p_L$ and $c_L$. We can neglect round keys $k_2$ and $k_4$ by computing differences of two chosen texts. Namely it satisfies $(L_2 * k_2) - (L'_2 * k_2) = (L_4 * k_4) - (L'_4 * k_4)$ and then $L_2 - L'_2 = L_4 - L'_4$, where we use XOR or modular subtraction as the operation "$-$," if an operation "$*$" is XOR or modular addition, respectively. As a result, for two distinct texts satisfying $p_L = c_L$, the relationship $\Delta L_2 = \Delta L_4$ is always satisfied. Now $L_2$ is computed by plaintext and round key $k_1$, and $L_4$ is computed by ciphertext and round key $k_5$. Consequently, we can execute a meet-in-the-middle-attack using two differences $\Delta_{L_2} = L_2 - L'_2$ and $\Delta_{L_4} = L_4 - L'_4$ which are calculated by an exhaustive search over all $2^{n/2}$ possible round keys $k_1$ and $k_5$, respectively.

**Theorem 1.**
*For the 5 round $K_iF$-Feistel, a text always satisfies $p_L = c_L$ if the text satisfies $L_2 * k_2 = L_4 * k_4$. Moreover a text always satisfies $L_2 * k_2 = L_4 * k_4$ if the text satisfies $p_L = c_L$.*

**Procedure and Evaluation.** The key recovery attack goes as follows. In the first step, we analyze $2^{n/2}$ known plaintexts and pick one text satisfying an $n/2$-bit relation $p_L = c_L$. We express this text as $(p, c)$. By an exhaustive search over all $2^{n/2}$ possible round keys $k_1$, calculate and store $X[k_1] = p_R \oplus F(p_L * k_1)$. In parallel, by an exhaustive search over all $2^{n/2}$ possible round keys $k_5$, we calculate and store $Y[k_5] = c_R \oplus F(c_L * k_5)$. In the second step, get another text satisfying $p_L = c_L$, and we express this chosen text as $(p', c')$. Like the first step, calculate $X'[k_1]$ and $Y'[k_5]$, respectively, and calculate and store $\Delta X[k_1] = X[k_1] - X'[k_1]$ and $\Delta Y[k_5] = Y[k_5] - Y'[k_5]$, respectively. In this time, round key pairs $(k_1, k_5)$ satisfying $\Delta X[k_1] = \Delta Y[k_5]$ are left as key candidates. Finally, by executing the second step several times, we can find the correct key $k_1$ and $k_5$.

For each key guess, the probability satisfying $\Delta X[k_1] = \Delta Y[k_5]$ is $2^{-n/2}$, then about $2^n \times 2^{-n/2} = 2^{n/2}$ keys will be left for each execution of the second step. By executing the second step 3 times, we can discard all key candidates except for the correct key, because $2^n \times (2^{-n/2})^3 = 2^{-n/2}$ keys will be left by this attack and it is under one. For each text, computing $X[k_1]$ requires $2^{n/2}$ and computing $Y[k_5]$ requires $2^{n/2}$. We analyze 1 text for the first step and 3 texts for the second step. Therefore the time complexity is $(1+3) \times 2 \times 2^{n/2} = 2^{n/2+3}$ by using $(1+3) \times 2^{n/2} = 2^{n/2+2}$ known plaintexts.

## 3.2 New Key Recovery Attack for the 6 Round $K_iF$-Feistel

Next if the key insert operation uses XOR, we propose an extension of our property. By applying this extended property, we propose a new key recovery attack for the 6 round $K_iF$-Feistel (see Fig.3) which is immune to the Knudsen's attack.

**Overview.** For the 5 round $K_iF$-Feistel, we could exactly pick the texts satisfying $L_2 * k_2 = L_4 * k_4$ by observing $p_L = c_L$. Unfortunately, for the 6 round, we can not pick those texts efficiently, because we can not know the value of $R_6$ which is $c_L$ for the 5 round. Then we aim to detect a set in which exactly one text satisfying $L_2 * k_2 = L_4 * k_4$ is included. We can detect such a set efficiently if the key insert operation is XOR. Our new attack always suggests the correct round key from the text satisfying $L_2 \oplus k_2 = L_4 \oplus k_4$, on the other hand, it never suggests the correct key from the text satisfying $L_2 \oplus k_2 \neq L_4 \oplus k_4$. Consequently, we execute our new attack for all texts in every set. We can know that key candidates are wrong if the number of the suggestions is not one. By repeating this attack for $n/2$ sets, we can recover the correct key.

**Why the Attack Is Applied Only for XOR?** From the Overview, we must pick a set in which only one text satisfying $L_2 * k_2 = L_4 * k_4$ is included. We can pick such a set efficiently if the key insert operation is XOR.

If the key insert operation is XOR, it satisfies $L_2 \oplus k_2 = L_4 \oplus k_4$ and then $L_2 \oplus L_4 = k_2 \oplus k_4$. From Fig. 2, $L_3$ is expressed as $F^{-1}(L_2 \oplus L_4) \oplus k_3$. When $L_2 \oplus L_4$ is a constant value which only depends on the round key values, $L_3$ is also a constant value which only depends on the round key values. Due to the condition $L_2 \oplus L_4 = k_2 \oplus k_4$, actually $L_3$ only depends on the round key values. Moreover $L_3$ is expressed by $p_L, p_R, k_1$ and $k_2$ as follows:

$$L_3 = F(F(p_L \oplus k_1) \oplus p_R \oplus k_2) \oplus p_L, \tag{2}$$

where $p_R$ is used only once. When we pick a set whose $2^{n/2}$ members have the structure that $p_L$ is a constant and $p_R$ is chosen from all values, the values of $L_3$ are all values ranging from 0 to $2^{n/2} - 1$ exactly once because of Equation 2. In this time, if a text satisfies $L_2 \oplus k_2 = L_4 \oplus k_4$, $L_3$ is the constant value which only depends on the round key values. Consequently, by using all $2^{n/2}$ chosen plaintexts such that $p_L$ is a constant and $p_R$ is a variable, we can pick a set in which exactly one text satisfying $L_2 \oplus k_2 = L_4 \oplus k_4$ is included.

Now we can pick a set in which exactly one text satisfying $L_2 \oplus k_2 = L_4 \oplus k_4$ is included. From theorem 1, we can pick a set in which exactly one text satisfying $p_L = R_6$ is included. By using this property, the following lemma is satisfied.

**Lemma 1.**
*For a set whose members are all $2^{n/2}$ chosen plaintexts such that $p_L$ is a constant and $p_R$ is a variable, exactly one text satisfies $F^{-1}(p_L \oplus c_R) \oplus c_L = k_6$.*

*Proof.* In a first step, we suppose the texts satisfying $L_2 \oplus k_2 = L_4 \oplus k_4$. From theorem 1, it always satisfies $R_6 = p_L$. Then it satisfies the following equation:

$$k_6 = F^{-1}(R_6 \oplus c_R) \oplus c_L = F^{-1}(p_L \oplus c_R) \oplus c_L.$$

On the other hand, if it satisfies $L_2 \oplus k_2 \neq L_4 \oplus k_4$, it always satisfies $R_6 \neq p_L$. Then it satisfies the following equation:

$$k_6 = F^{-1}(R_6 \oplus c_R) \oplus c_L \neq F^{-1}(p_L \oplus c_R) \oplus c_L.$$

Thus we complete the proof. □

**Procedure and Evaluation.** By using lemma 1, we can attack the 6 round $K_iF$-Feistel. We assume that attackers can access the inverse of the function $F$, and it is popular because the inverse functions are open to the public in various environment.

The key recovery attack by using lemma 1 goes as follows. First, choose all $2^{n/2}$ chosen plaintexts such that $p_L$ is a constant and $p_R$ is a variable, $p[i] = (p_L, p_R[i])$ for $i = 1, \ldots, 2^{n/2}$. Query these chosen plaintexts and get the ciphertexts. Next, for all chosen texts, calculate $v = F^{-1}(p_L \oplus c_R[i]) \oplus c_L[i]$ $(1 \leq i \leq 2^{n/2})$, and count the number of occurrences of each $v$. From lemma 1, the number $\#v$ is always 1 for any $p_L$ when it satisfies $v = k_6$. However the probability of $\#v = 1$ is given $e^{-1}$ in every wrong key, and keys which satisfy $\#v \neq 1$ are discarded. By repeating the attack $n/2$ times, the probability that a wrong key is left is given $e^{-n/2}$ and it is negligible. Therefore attackers can recover the correct key with high probability in time $\frac{n}{2}2^{n/2}$ by using CPA with $\frac{n}{2}2^{n/2}$ texts.

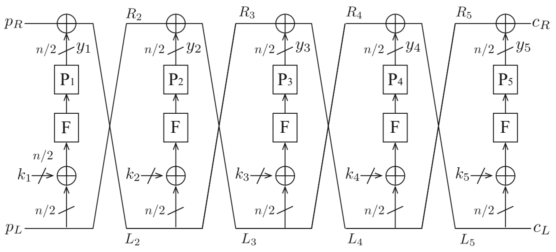


**Fig. 4.** The 5 round $K_iFP_i$-Feistel

## 4 New Key Recovery Attacks on the $K_iFP_i$-Feistel

### 4.1 A New Property for the 5 Round $K_iFP_i$-Feistel

Similarly to the $K_iF$-Feistel, we aim to detect a property about relationship between plaintext and ciphertext of the 5 round $K_iFP_i$-Feistel. Based on this property, we propose a new key recovery attack for the 5 round $K_iFP_i$-Feistel. This attack can be executed with time complexity smaller than the Knudsen's attack, and it is KPA not CPA.

**Overview.** The $K_iF$-Feistel is an efficient construction, but the upper bound for the security is lower than that of the $F_{K_i}$-Feistel. One of the reasons is that attackers can pick the texts which satisfy $L_2 * k_2 = L_4 * k_4$ easily by observing $p_L = c_L$ from theorem 1. Now we consider the $K_iFP_i$-Feistel. In this construction, a text satisfying $p_L = c_L$ does not always satisfy $L_2 * k_2 = L_4 * k_4$ because different round functions are used in every round. Then we can not pick the texts which satisfy $L_2 * k_2 = L_4 * k_4$ efficiently. Therefore one may think that the $K_iFP_i$-Feistel is more secure than the $K_iF$-Feistel. However, if the key insert operation is XOR, attackers can still exploit the property for $L_2 * k_2 = L_4 * k_4$ in a probabilistic manner.

For the 5 round $K_iFP_i$-Feistel (see Fig. 4), we focus our attention on the texts satisfying $L_2 \oplus k_2 = L_4 \oplus k_4$ similarly to the $K_iF$-Feistel. In this time, $L_3$ is a constant value which only depends on the round key values because $L_3 = F^{-1}(P_3^{-1}(L_2 \oplus L_4)) \oplus k_3$. For the 5 round $K_iFP_i$-Feistel, the plaintext is expressed as $p_L = P_2(F(L_2 \oplus k_2)) \oplus L_3$ and the ciphertext is expressed as $c_L = P_4(F(L_4 \oplus k_4)) \oplus L_3$. Then if it satisfies $L_2 \oplus k_2 = L_4 \oplus k_4$, $c_L$ is expressed by $p_L$ and $k'$ which only depends on the round key values. Therefore the relationship between plaintext and ciphertext is expressed by the constant value which only depends on the round key values. By using this relationship, we propose an algorithm to recover the value of $k'$. Our algorithm always returns the correct key $k'$ from the text satisfying $L_2 \oplus k_2 = L_4 \oplus k_4$. On the other hand, from the others, namely $L_2 \oplus k_2 \neq L_4 \oplus k_4$, our algorithm returns one value at random from all $2^{n/2}$ values. Consequently, our algorithm suggests the correct key with probability $(2^{-n/2} \cdot 1) + ((1 - 2^{-n/2}) \cdot 2^{-n/2}) \approx 2^{-n/2+1}$, but suggests other wrong keys with probability $((1 - 2^{-n/2}) \cdot 2^{-n/2}) = 2^{-n/2}$. By analyzing several texts, our algorithm suggests the correct key two times more than other wrong keys.

**An Algorithm for the Key Recovery Attack of 5 Round $K_iFP_i$-Feistel.** We propose an algorithm which returns the correct key $k'$ with probability $2^{-n/2+1}$, and the probability is double compared with that of other wrong keys. This algorithm uses the following property:

**Theorem 2 (New property for the 5 round $K_iFP_i$-Feistel).** *For the 5 round, the relationship between plaintext and ciphertext is expressed as follows:*

$$c_L = P_4(P_2^{-1}(p_L)) \oplus k' \tag{3}$$

*with probability about $2^{-n/2+1}$. Then $k'$ is a constant value which is determined by round keys.*

*Proof.* To prove theorem 3, we separate one incident satisfying Equation 3 from two cases: one satisfies $L_2 \oplus L_4 = k_2 \oplus k_4$ and Equation 3, the other satisfies $L_2 \oplus L_4 \neq k_2 \oplus k_4$ and Equation 3. In a first step, we analyze texts satisfying $L_2 \oplus L_4 = k_2 \oplus k_4$, then $L_3$ is expressed as $k_3 \oplus F^{-1}(P_3^{-1}(k_2 \oplus k_4))$, and $L_3$ is a constant value which is determined by round keys $k_2$, $k_3$ and $k_4$. In this time, the relationship between plaintext and ciphertext is expressed as follows:

$$\begin{aligned} c_L &= P_4(F(L_4 \oplus k_4)) \oplus L_3 \\ &= P_4(P_2^{-1}(P_2(F(L_4 \oplus k_4)))) \oplus L_3 \\ &= P_4(P_2^{-1}(P_2(F(L_2 \oplus k_2)))) \oplus L_3 = P_4(P_2^{-1}(p_L \oplus L_3)) \oplus L_3 \\ &= P_4(P_2^{-1}(p_L)) \oplus P_4(P_2^{-1}(L_3)) \oplus L_3 = P_4(P_2^{-1}(p_L)) \oplus k'. \end{aligned}$$

The probability satisfying $L_2 \oplus L_4 = k_2 \oplus k_4$ is $2^{-n/2}$, and $c_L = P_4(P_2^{-1}(p_L)) \oplus k'$ is always satisfied under this condition. Thus the probability satisfying Equation 3 is $2^{-n/2} \times 1 = 2^{-n/2}$.

Next, we consider case with $L_2 \oplus k_2 \neq L_4 \oplus k_4$. In this time, the following equation:

$$\begin{aligned} c_L &= P_4(F(L_4 \oplus k_4)) \oplus L_3 \\ &= P_4(P_2^{-1}(p_L)) \oplus P_4(P_2^{-1}(p_L)) \oplus P_4(F(L_4 \oplus k_4)) \oplus L_3 \\ &= P_4(P_2^{-1}(p_L)) \oplus P_4(F(L_2 \oplus k_2)) \oplus P_4(P_2^{-1}(L_3)) \oplus P_4(F(L_4 \oplus k_4)) \oplus L_3 \end{aligned}$$

is satisfied. If the following equation

$$k' = P_4(F(L_2 \oplus k_2)) \oplus P_4(P_2^{-1}(L_3)) \oplus P_4(F(L_4 \oplus k_4)) \oplus L_3$$

is satisfied by chance, it satisfies Equation 3, and its probability is $2^{-n/2}$. The probability satisfying $L_2 \oplus L_4 \neq k_2 \oplus k_4$ is $1 - 2^{-n/2}$, and the probability satisfying Equation 3 is $2^{-n/2}$ when it satisfies $L_2 \oplus L_4 \neq k_2 \oplus k_4$. Thus the probability satisfying Equation 3 is $(1 - 2^{-n/2}) \times 2^{-n/2} \approx 2^{-n/2}$.

Consequently, the total probability is given by about $2^{-n/2} + 2^{-n/2} = 2^{-n/2+1}$ and we complete the proof. □

**Procedure and Evaluation.** The key recovery attack by using theorem 2 goes as follows. First, analyze $t \cdot 2^{n/2}$ known plaintexts, and calculate $v = c_L \oplus P_4(P_2^{-1}(p_L))$, count the number of occurrences of each $v$. From theorem 2, the

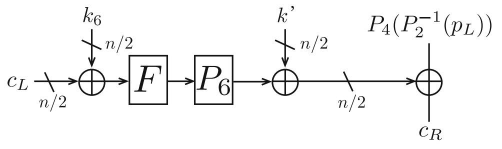


**Fig. 5.** The relationship between plaintext and ciphertext of the 6 round $K_iFP_i$-Feistel

number $\#v$ is about $t \cdot 2^{n/2} \cdot 2^{-n/2+1} = 2t$ for $v = k'$ On the other hand, the number $\#v$ is about $t \cdot 2^{n/2} \cdot 2^{-n/2} = t$ on average for $v \neq k'$. Then we can recover the correct $k'$ by analyzing $t \cdot 2^{n/2}$, where $t$ is a function of $n$. This attack can only recover $k'$, but once $k'$ is recovered, we can recover $k_1$ and $k_5$ to execute similar attack for the 5 round $K_iF$-Feistel.

## 4.2 New Key Recovery Attack on the 6 Round $K_iFP_i$-Feistel

**Overview.** For the 5 round $K_iF$-Feistel, we showed $c_L$ is expressed by $p_L$ and the constant value $k'$ which only depends on the round key values with probability $2^{-n/2+1}$. Therefore by using the relationship between $p_L$ and $c_L$ with a double bias, we could recover $k'$. On the other hand, for the 6 round, we can not use this method because we can not know the value of $R_6$ which is $c_L$ for the 5 round. Then we first assume that all texts satisfy $L_2 \oplus k_2 = L_4 \oplus k_4$, and calculate $R_6$ as follows:

$$R_6 = P_4(P_2^{-1}(p_L)) \oplus k',$$

and the relationship between ciphertext and plaintext is expressed as follows:

$$c_R \oplus P_4(P_2^{-1}(p_L)) = P_6(F(c_L \oplus k_6)) \oplus k', \tag{4}$$

where $k_6$ and $k'$ are unknown constant values which only depend on the round key values. Figure 5 shows the circuit of the Equation 4. Then if we exhaustively search for over all $2^{n/2}$ possible round keys $k_6$, we can recover $k'$ from Equation 4 by using several texts. Unfortunately, this attack is not effective. The reason is that the probability satisfying the Equation 4 is at most probability $2^{-n/2+1}$. If we aim to recover $k'$ by using this bias, we have to search for all $2^{n/2}$ possible round keys $k_6$ against over $2^{n/2}$ texts. This time complexity is over $2^n$.

Then we consider the chosen ciphertext attack. In this attack, attackers can choose all $2^{n/2}$ chosen ciphertexts such that $c_L$ is a constant and $c_R$ is a variable. In this model, we propose a new technique called "the chosen texts technique." By using the chosen texts technique, we can choose a set whose members satisfy Equation 4 with higher probability than $2^{-n/2+1}$. Namely the chosen set from the chosen texts technique is more biased than original set. For this chosen set, we exhaustively search for over all $2^{n/2}$ possible round keys $k_6$ and recover $k'$. We stress that we can attack the 6 round $K_iFP_i$-Feistel by using CPA, because the encryption and the decryption for this cipher have symmetry. Here, keys which are recoverd by this attack are round key $k_1$ and constant value $k'$ which is uniquely determined by round key $k_3, k_4$ and $k_5$. However we show this attack by using a chosen ciphertext attack for simplicity in this section.

**Overview of the Chosen Texts Technique.** We propose the chosen texts technique by which we can choose a set whose members satisfy Equation 4 with higher probability than $2^{-n/2+1}$. First, we analyze all $2^{n/2}$ chosen ciphertexts such that $c_L$ is a constant and $c_R$ is a variable. From Equation 4, if $c_L$ is a constant, we know $c_R \oplus P_4(P_2^{-1}(p_L))$ is uniquely determined by $k_6$ and $k'$. Moreover we know that Equation 4 is satisfied with probability $2^{-n/2+1}$. Among several texts with satisfying Equation 4, with a fixed value of $c_L$ but different values of $c_R$, the values of $c_R \oplus P_4(P_2^{-1}(p_L))$ are the same, and the probability is $2^{-n/2+1}$ which is double compared with the others. Namely, we calculate $v = c_R \oplus P_4(P_2^{-1}(p_L))$ and count the number of occurrences of each $v$. The number $\#v$ is about $2^{n/2} \cdot 2^{-n/2+1} = 2$ when it satisfies $v = P_6(F(c_L \oplus k_6)) \oplus k'$. On the other hand, the number $\#v$ is about $2^{n/2} \cdot 2^{-n/2} = 1$ when it satisfies $v \neq P_6(F(c_L \oplus k_6)) \oplus k'$. By using this slight bias, the chosen texts technique choose a set whose members satisfy Equation 4 with high probability than $2^{-n/2+1}$.

**Lemma 2 (Chosen texts technique).**
*Choose all $2^{n/2}$ chosen ciphertexts such that $c_L$ is a constant and $c_R$ is a variable, and query these chosen ciphertexts and get the plaintexts. Count the number of occurrences of each $v = c_R \oplus P_4(P_2^{-1}(p_L))$. Choose $(c_L, v)$ whose number of occurrences is maximum. In this time, $P_T$ denotes the probability that the chosen text satisfies Equation 4, and the probability $P_T$ is expressed as*

$$P_T > \sum_{i=0}^{2^{n/2}} \frac{2^i}{i!} e^{-2} \left( \frac{1}{e} \sum_{j=0}^{i} \frac{1}{j!} \right)^{2^{n/2}} > 2^{-n/2}.$$

**Procedure and Evaluation.** By using lemma 2, we show a key recovery attack on the 6 round $K_iFP_i$-Feistel, and the attack goes as follows. In the first step, choose all $2^{n/2}$ chosen ciphertexts such that $c_L$ is a constant and $c_R$ is a variable, $c[i] = (c_L, c_R[i])$ for $i = 1, \ldots, 2^{n/2}$. Query these chosen ciphertexts and get the plaintexts. In the second step, execute the chosen texts technique. Namely, calculate $v = c_R[i] \oplus P_4(P_2^{-1}(p_L[i]))$, count the number of occurrences of each $v$, and pick $(c_L, v)$ whose number of occurrences is maximum. In the third step, for the picked text $(c_L, v)$, exhaustively search for over all $2^{n/2}$ possible round keys $k_6$ and calculate $k'$ from Equation 4, and vote $2^{n/2}$ key candidates $(k_6, k')$ to the key space of size $2^n$. For a picked text, if the text satisfying Equation 4 is chosen in the second step, it votes the correct $(k_6, k')$ and $2^{n/2} - 1$ random key candidates except the correct key. If the text satisfying Equation 4 is not chosen in the second step, it votes the $2^{n/2}$ random key candidates except the correct key. Finally, by repeating the first, second and third step $2^d$ times, we recover the correct key $(k_6, k')$. The correct key $(k_6, k')$ is voted at least $2^d \cdot P_t$, but the wrong key is voted $2^d \cdot 2^{n/2} \cdot 2^{-n} = 2^{d-n/2}$ on average.

Now we give the suitable number of chosen ciphertext. Here we consider $n = 128$ because many current block ciphers have 128-bit block size. For $n = 128$, it satisfies $P_t > 2^{-44}$ because of lemma 2. By repeating the first, second and third step $2^{3n/8}$ times (namely, using $2^{7n/8}$ chosen ciphertexts), The correct key $(k_6, k')$ is voted at least $2^{3n/8} \cdot 2^{-44} = 16$. On the other hand, the wrong key is voted about $2^{7n/8}$ key candidates to the key space of size $2^n$ at random. The probability that one of the wrong keys is voted over 16 is at most $\frac{1}{16!}$ [17]. Consequently, our new attack can recover the correct $(k_6, k')$.

Next we show the time complexity. First we must calculate the calculation $2^{7n/8}$ times for the second step. Next we must evaluate in every round key $(k_6, k')$, and the time complexity is $2^{7n/8}$. Therefore the time complexity for this attack is $2^{7n/8+1}$. We show the experimental report of this attack in Appendix A.

## 5 Discussions

In this section, we discuss our results and give observations on the design of Feistel ciphers. From our results, we can conclude that the $F_{K_i}$-Feistel and the $K_iF$-Feistel have different upper bounds for the security, so it is an important problem to consider how to insert round keys. Instead of the $K_iF$-Feistel, we can use the $K_iFP_i$-Feistel which is more secure than the $K_iF$-Feistel and easier to design than the $F_{K_i}$-Feistel. Surely, from the time complexity and the number of texts, the network is more secure on our attacks than the $K_iF$-Feistel, but we can still attack the 6 round network whose operation is XOR. Namely, the $K_iFP_i$-Feistel is still more insecure than the $F_{K_i}$-Feistel. On the other hand, when the key insert operation is a modular addition, we can not attack the 6 round network, because our properties for the 6 round key recovery attacks does not work. It seems these results show a conclusion that use of a modular addition is more secure than that of XOR for the key insert operation. Accordingly, we analyze a peculiar Feistel network whose $\Psi_{k_i}$ is defined as follows:

$$(L_{i+1}, R_{i+1}) = (R_i \boxplus P_iF(L_i \boxplus k_i), L_i),$$

where $P_i$ is the linear operation of a modular addition. As a result, we can attack on this 6 round network by the similar attack to that for the 6 round $K_iFP_i$-Feistel. Our attacks use some relationship between plaintext and ciphertext when it satisfies $L_2 * k_2 = L_4 * k_4$. Generally, it is not efficient to have a modular addition for the hardware implementation, so we consider that the diffusion part should be used a nonlinear operation with respect to XOR. Even if we observe the relationship between plaintext and ciphertext, we can not distinguish whether the text satisfies $L_2 * k_2 = L_4 * k_4$. More detailedly, if it uses a nonlinear operation which satisfies $P_ix \oplus P_jx = (P_i \oplus P_j)x$ with a negligible probability, the network is immune to our attacks.

## 6 Conclusion

We revisited the Knudsen's Feistel networks where the round functions are chosen at random from a family of $2^k$ randomly chosen function. In this paper, we considered that simple construction, the $K_iF$-Feistel, and showed that the 6 round construction was not secure. Moreover, for the $K_iFP_i$-Feistel which is more secure than $K_iF$-Feistel, we showed that the 6 round construction was not as secure as the $K_iF$-Feistel. This means that how to insert round key is very influential in the upper bound for the security. Moreover this means that the 6 round networks are not enough to complicate the relationship between plaintext and ciphertext. To complicate the relationship, we propose the diffusion part for the $K_iFP_i$-Feistel should be used a nonlinear operation with respect to XOR. This construction has the immunity to our attacks and maintenance of the efficiency.

## 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. 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)
3. Isobe, T., Shibutani, K.: All Subkeys Recovery Attack on Block Ciphers: Extending Meet-in-the-Middle Approach. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 202–221. Springer, Heidelberg (2013)
4. Knudsen, L.R.: The Security of Feistel Ciphers with Six Rounds or Less. J. Cryptology 15(3), 207–222 (2002)
5. 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)
6. Lampe, R., Patarin, J.: Security of Feistel Schemes with New and Various Tools. IACR Cryptology ePrint Archive 2012, 131 (2012)
7. Lee, H., Lee, S., Yoon, J., Cheon, D., Lee, J.: The SEED Encryption Algorithm RFC4269 (2005)
8. Luby, M., Rackoff, C.: How to construct pseudorandom permutations from pseudorandom functions. SIAM J. Comput. 17(2), 373–386 (1988)
9. Lucks, S.: Faster Luby-Rackoff Ciphers. In: Gollmann, D. (ed.) FSE 1996. LNCS, vol. 1039, pp. 189–203. Springer, Heidelberg (1996)
10. National Soviet Bureau of Standards: Information Processing System – Cryptographic Protection – Cryptographic Algorithm GOST 28147-89 (1989)
11. Patarin, J.: How to Construct Pseudorandom and Super Pseudorandom Permutations from one Single Pseudorandom Function. In: Rueppel, R.A. (ed.) EUROCRYPT 1992. LNCS, vol. 658, pp. 256–266. Springer, Heidelberg (1993)
12. Patarin, J.: Luby-Rackoff: 7 Rounds Are Enough for $2^{n(1-\epsilon)}$Security. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 513–529. Springer, Heidelberg (2003)
13. Patarin, J.: Security of Random Feistel Schemes with 5 or More Rounds. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 106–122. Springer, Heidelberg (2004)
14. Patarin, J.: Security of balanced and unbalanced Feistel Schemes with Linear Non Equalities. IACR Cryptology ePrint Archive 2010, 293 (2010)
15. 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)

16. Shirai, T., Shibutani, K.: On Feistel Structures Using a Diffusion Switching Mechanism. In: Robshaw, M. (ed.) FSE 2006. LNCS, vol. 4047, pp. 41–56. Springer, Heidelberg (2006)
17. Suzuki, K., Tonien, D., Kurosawa, K., Toyota, K.: Birthday paradox for multi-collisions. In: Rhee, M.S., Lee, B. (eds.) ICISC 2006. LNCS, vol. 4296, pp. 29–40. Springer, Heidelberg (2006)

## Appendix A: The Feasibility by Implementing the Attack for Small Block Sizes

For feasibility evaluation of our attack, we execute experiments for our attacks on the $K_iF$-Feistel and $K_iFP_i$-Feistel which have small block sizes. In our experiments, we evaluate by the average of 1,000 samples in every block length. In all samples, we choose functions $F$ and round keys randomly. In this paper, we show only key recovery attack on the 6 round $K_iFP_i$-Feistel due to limitations of space.

### Key Recovery Attack on the 6 Round $K_iFP_i$-Feistel

We show the experimental results for the key recovery attack on the 6 round $K_iFP_i$-Feistel. When the correct key $(k_6, k')$ is one of the most voted candidates, we define it as "the success." We summarize experimental results in Figure 6. Here the horizontal axis shows block lengths, the right vertical axis shows the average number of the vote to the correct key $(k_6, k')$, and the left vertical axis shows the success probability on our attack. In our experiments, we always use $2^{3n/8}$ for the value $2^d$. From Fig. 6, the average number of the vote to the correct key is always higher than the lower bound, it means the chosen texts technique is valid. Moreover, the longer the block length becomes, the more quantity the average number of the vote to the correct key is. Accordingly, the probability of success is close to one.

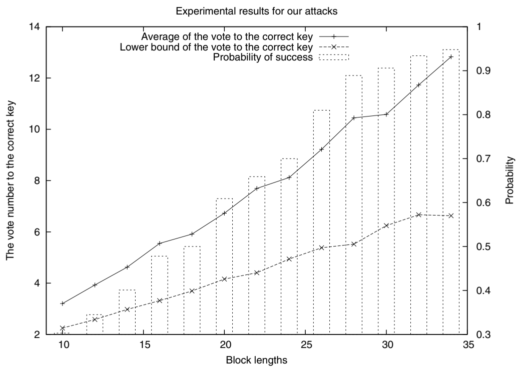


**Fig. 6.** Experimental results for new attacks of the 6 round $K_iFP_i$-Feistel

# Fairness in Concurrent Signatures Revisited

Willy Susilo[1,⋆], Man Ho Au[1], Yang Wang[1], and Duncan S. Wong[2]

[1] Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Australia
{wsusilo,aau}@uow.edu.au,
yw990@uowmail.uow.edu.au

[2] City University of Hong Kong, Hong Kong
duncan@cityu.edu.hk

**Abstract.** Concurrent signature, introduced by Chen, Kudla and Paterson, is known to just fall short to solve the long standing fair exchange of signature problem without requiring any trusted third party (TTP). The price for not requiring any TTP is that the initial signer is always having some advantage over the matching signer in controlling whether the protocol completes or not, and hence, whether the two ambiguous signatures will bind concurrently to their true signers or not. In this paper, we examine the notion and classify the advantages of the initial signer into three levels, some of which but not all of them may be known in the literature.

– Advantage level 0 is the commonly acknowledged fact that concurrent signature is not abuse-free since an initial signer who holds a keystone can always choose to complete or abort a concurrent signature protocol run by deciding whether to release the keystone or not.
– Advantage level 1 refers to the fact that the initial signer can convince a third party that both ambiguous signatures are valid without actually making the signatures publicly verifiable.
– Advantage level 2 allows the initial signer to convince a third party that the matching signer agrees to commit to a specific message, and nothing else. We stress that advantage level 2 is *not* about proving the possession of a keystone. Proving the knowledge of a keystone would make the malicious initial signer accountable as this could only be done by the initial signer.

We remark that the original security models for concurrent signature do not rule out the aforementioned advantages of the initial signer. Indeed, we show that theoretically, the initial signer always enjoys the above advantages for any concurrent signatures. Our work demonstrates a clear gap between the notion of concurrent signature and optimistic fair exchange (OFE) in which no party enjoys advantage level 1. Furthermore, in a variant known as Ambiguous OFE, no party enjoys advantage level 1 and 2.

**Keywords:** concurrent signatures, scenarios, fairness.

⋆ W. Susilo is supported by ARC Future Fellowship FT0991397.

## 1 Introduction

Fair exchange of digital signatures has been considered as a fundamental problem in cryptography. It is a useful cryptographic protocol that allows secure and fair e-commerce applications to exchange digital signatures for legal contracts or agreements. Nowadays, goods and services are being exchanged electronically over the Internet. Our main goal is to ensure that the exchanges are *fair*, which means that at the end of an exchange between two parties, either both parties receive the complete items or none of them obtains anything.

Fairness in exchanging signatures is normally achieved with the help of a trusted third party (TTP), which is often offline. There were several attempts where a fair exchange of signatures can be achieved with a "semi-trusted" TTP who can be called upon to handle disputes between signers [2]. This type of fair exchanges is also referred to as an optimistic fair exchange (OFE) [3,4]. A well-known open problem in fair exchange is the requirement of a dispute resolving TTP whose role cannot be replaced by a normal certification authority.

In Eurocrypt 2004 [6], Chen, Kudla and Paterson presented a new cryptographic primitive called "concurrent signature" to allow two parties to produce two ambiguous signatures, such that both signatures do not bind to their true signers. Upon the release of the signatures, any third party cannot identify the true signer who generated the signature. However, upon the release of an extra piece of information called the keystone, both signatures will bind concurrently. The merit of concurrent signature relies on the fact that there is *no* TTP required. This has been considered to be very practical, and hence, concurrent signatures have been promising to be adopted in practice (c.f. OFE where the need for a TTP may not exist in the real scenario). The demand of requiring a TTP in cryptographic schemes has made cryptographic schemes less attractive for adoption in practice, and hence, concurrent signatures have been very promising in bridging the gap between theoreticians and practitioners. Further, Chen, Kudla and Paterson presented a concrete concurrent signature scheme based on a variant of Schnorr based ring signature scheme [1]. In their scheme, any third party cannot be convinced that a signature has indeed been signed by one particular signer, since any signer can always generate this signature by himself/herself.

In a concurrent signature protocol run, there are two parties involved, namely Alice and Bob (or $A$ and $B$, respectively). Since one party is required to create a keystone and sends the first message to the other party, we call this party as the *initial signer*. A party who responds to the initial signature by creating another signature is called a *matching signer*. Without losing generality, throughout this paper, we assume that Alice (or $A$, resp.) is the initial signer and Bob (or $B$, resp.) is the matching signer.

It is acknowledged that concurrent signature is not perfectly fair, since Alice is in control of the time at which the keystone is released and thus, control *when* the ambiguous signatures become binding to their respective signers. Alice can even decide not to complete a concurrent signature protocol run if Alice decided not to to release the keystone ultimately. Nonetheless, the concept of *fairness* in a concurrent signature is defined as follows: "once Alice releases the keystone, both ambiguous signatures from Alice and Bob binds concurrently". Specifically, it is required that Alice cannot output a maliciously crafted keystone so that the ambiguous signature from Bob together with this keystone passes the verification algorithm, yet verification of the ambiguous signature created by Alice (perhaps also maliciously) together with this keystone would output failure. We call this definition of fairness a "white-box" guarantee, as the malicious Alice is required to convert the ambiguous signature corresponding to Bob to a "well-formed" publicly verifiable signature under Bob to be considered successful. This is possibly sufficient for legal contract signing purpose, since the contract is only valid if and only if a "well-formed" signature is present. This definition is adopted in Chen et al.'s paper and the subsequent works.

## 1.1 Fairness in Practice

In this paper, we make the observation that the formal definition of fairness does *not* necessarily capture the fairness in practice completely. For example, there is *no guarantee* that Alice could not convince a third party Carol that Bob has signed a message, $M_B$, that is, committed to $M_B$, *without* revealing the keystone. We identify that a malicious initial signer may enjoy three levels of advantages in concurrent signature.

(a) Level 0 advantage is inherent in concurrent signature. The initial signer can always choose to abort or complete a concurrent signature protocol run.
(b) Level 1 advantage allows the initial signer to demonstrate the fact that he/she is capable of making both signatures valid if he/she wanted to, without actually making both signatures publicly verifiable.
(c) Level 2 advantage allows the initial signer to convince a third party that the matching signer has agreed to committed to a certain message, for example, $M_B$, without revealing anything else.

These advantage levels have concrete implications regarding the use of concurrent signatures in practical scenarios. As discussed by Chen et al. in their seminal paper [6], it is very often the case that the matching signer would not mind sacrificing level 0 advantage. However, regarding level 1 and 2 advantages, concurrent signatures may not be suitable in some other scenarios such as tendering systems (c.f. [6] as we will show with details in the later part of this paper). On the other hand, contrary to the common belief that concurrent signature is applicable to tendering systems (such as [6, 17, 18]), level 1 advantage to a malicious initial signer could be unacceptable to some of the suppliers. Hence, in those scenarios, the OFE [2–4, 7] or Ambiguous OFE [10–12] systems are indeed more suitable compared to concurrent signatures.

## 1.2 Our Contributions

Firstly, we show that any constructions of concurrent signature following Chen, Kudla and Peterson is *always* subject to abuse by the initial signer, with advantage level 1 and 2, in addition to the commonly acknowledged advantage level 0.

We present generic methods that allow a malicious initial signer to convince any third party that he/she has the ability to make both signatures verifiable (level 1), or that the matching signer has committed to his message (level 2). Secondly, we examine one variant of concurrent signatures, namely, asymmetric concurrent signature [13] and demonstrate how a malicious initial signer can exhibit his/her level 1 advantage in an effective manner. Our attack is practical and its implication may discourage the adoption of concurrent signatures in some application scenarios, and particularly, when it is undesirable to allow a malicious initial signer to convince anyone of the binding of the matching signer's signature without making the signature publicly verifiable.

## 1.3 Related Work

Following the seminal work by Chen et al., many subsequent concurrent signature schemes have been proposed (such as [13,14,16,17]). Nguyen [13] proposed an interesting variant that embraces the asymmetric property of concurrent signatures (c.f. the symmetric property of all the previously known concurrent signature schemes). Furthermore, Nguyen noted that the construction techniques of an asymmetric concurrent signature scheme can be used for constructing a multi-party concurrent signature scheme, which is the solution to the open problem stated in Chen et al.'s seminal work [6]. Subsequently, Tonien, Susilo and Safavi-Naini [16] proposed a multi-party concurrent signature scheme that uses a different model from the construction achieved from [13].

In an orthogonal direction, Susilo, Mu and Zhang [14] investigated the privacy issue in concurrent signatures. They observed that prior the release of a keystone, and just from the two ambiguous signatures, any third party can already conclude that the two ambiguous signatures must be created by these two possible signers. At the same time, if the possible signers are believed to be honest, the outsider can already tell who the actual signer is corresponding to each ambiguous signature. They then introduced a stronger requirement called *perfect ambiguity*, which requires the ambiguous signatures to remain "ambiguous" even if the two potential signers are known to be honest. Unfortunately, their scheme is shown to be insecure by Wang, Bao and Zhou [17], and subsequently, Wang et al. proposed a modified scheme that is proven secure under this stronger notion.

Yuen, Wong, Susilo and Huang [18] constructed a concurrent signature variant that supports negotiation between the initial signer and the matching signer on who will control the final binding of the ambiguous signatures. They showed that their model is compatible with the original definition by Chen et al. [6].

Very recently, Tan, Huang and Wong [15] presented the first concurrent signature scheme that is based on the standard assumption. Their ambiguity model is very similar to the one proposed by Yuen et al. [18].

## 1.4 Roadmap

In the next section, we review the notion of concurrent signatures due to Chen et al. and the notations used in the rest of this paper. In Section 3, we present our classification of advantage levels to the initial signer in detail, discuss their implications and present generic techniques which allows the initial signer to enjoy these advantages. In Section 4, we show how an initial signer can enjoy level 1 advantage in the asymmetric concurrent signature scheme due to Nguyen [13]. Finally, we conclude the paper in Section 5.

## 2 Notions and Definitions of Concurrent Signature

### 2.1 Notations

For a finite set $\mathcal{S}$, we will denote by $x \in_R \mathcal{S}$ the operation of selecting an element $x$ uniformly at random from $\mathcal{S}$. If $p$ is a positive integer, we use $\mathbb{Z}_p$ to denote the set $\{0, \ldots, p-1\}$.

### 2.2 Concurrent Signatures

In the following, we review the definition of concurrent signatures from [6]. A concurrent signature comprises four algorithms (SETUP, ASIGN, AVERIFY, VERIFY). Their formal definitions are given below.

SETUP. On input security parameter $1^\lambda$, this probabilistic algorithm outputs the description of the set of participants $\mathcal{U}$, the message space $\mathcal{M}$, the signature space $\mathcal{S}$, the keystone space $\mathcal{K}$, the keystone fix space $\mathcal{F}$, and a function $\mathsf{KGEN} : \mathcal{K} \to \mathcal{F}$. It is also assumed the public keys $\{X_i\}$ and their respective secret keys $\{x_i\}$ are also generated by this algorithm. We use $\pi$ to denote additional system parameters. We assume $(\pi, \mathcal{U}, \mathcal{M}, \mathcal{S}, \mathcal{K}, \mathcal{F}, \mathsf{KGEN}, \{X_i\})$ are available to all participants while each user retain his/her own secret key $x_i$.

ASIGN. On input $(X_i, X_j, x_i, h_2, M)$ where $X_i, X_j \in \{X_i\}$ such that $X_i \neq X_j$, $x_i$ being the secret key corresponds to the public key $X_i$, $h_2 \in \mathcal{F}$, $M \in \mathcal{M}$, this algorithm outputs an ambiguous signature $\sigma = (s, h_1, h_2)$ on message $M$, where $s \in \mathcal{S}$, $h_1, h_2 \in \mathcal{F}$.

AVERIFY. On input $(\sigma, X_i, X_j, M)$, where $\sigma = (s, h_1, h_2)$, $s \in \mathcal{S}$, $h_1, h_2 \in \mathcal{F}$, $X_i, X_j$ are distinct public keys and $M \in \mathcal{M}$, outputs 0/1. It is required that $\mathsf{AVERIFY}((s, h_1, h_2), X_i, X_j, M) = \mathsf{AVERIFY}((s, h_2, h_1), X_j, X_i, M)$.

VERIFY. On input $(k, (\sigma, X_i, X_j, M))$ such that $k \in \mathcal{K}$ and $\sigma = (s, h_1, h_2)$, it outputs 0 if $h_2 \neq \mathsf{KGEN}(k)$. Otherwise, it outputs $\mathsf{AVERIFY}(\sigma, X_i, X_j, M)$.

Below is a recap of the interactive protocol in which the above algorithms are used in the exchange of signatures in a concurrent manner amongst two participants.

1. We assume SETUP has been executed and all participants have their own key pair already. Below we describe how Alice with key pair $(X_A, x_A)$ exchanges signatures with Bob with key pair $(X_B, x_B)$.
2. Alice picks a random $k \in \mathcal{K}$, computes $f = \mathsf{KGEN}(k)$ and obtains $\sigma_A := (s_A, h_A, f)$ from $\mathsf{ASIGN}(X_A, X_B, x_A, f, M_A)$. Alice sends $\sigma_A, M_A$ to Bob.

3. Bob verifies Alice's ambiguous signature by invoking $\mathsf{AVERIFY}(\sigma_A, X_A, X_B, M_A)$. If $\sigma_A$ is valid, Bob obtains $\sigma_B := (s_B, h_B, f)$ from $\mathsf{ASIGN}(X_B, X_A, x_B, f, M_B)$. Bob sends $(\sigma_B, M_B)$ to Alice.
4. Alice verifies Bob's ambiguous signature by invoking $\mathsf{AVERIFY}(\sigma_B, X_B, X_A, M_B)$. If $\sigma_B$ is valid, Alice releases the keystone $k$. Parse $S_A$ as $(k, \sigma_A, X_A, X_B, M_A)$ and $S_B$ as $(k, \sigma_B, X_B, X_A, M_B)$.
5. Everybody can now verify both signatures $S_A$ and $S_B$ using $\mathsf{VERIFY}$.

The correctness is defined in the usual manner. Specifically, if $\sigma = \mathsf{ASIGN}(X_i, X_j, x_i, f, M)$ and $S = (k, \sigma, X_i, X_j, M)$, then $\mathsf{AVERIFY}(\sigma, X_i, X_j, M) = 1$. In addition, if $f = \mathsf{KGEN}(k)$ for some $k \in \mathcal{K}$, then $\mathsf{VERIFY}(S) = 1$.

## 2.3 Security Model

As discussed in [6], a concurrent signature should satisfy three security requirements, namely, *unforgeability*, *ambiguity* and *fairness.* For completeness, we review these security requirements as follows.

**Unforgeability.** The following game is used to capture the existential unforgeability of a concurrent signature.

**Definition 1 (Unforgeability).** *A concurrent signature is unforgeable if no PPT adversary $\mathcal{A}$ can win the following game with a challenger $\mathcal{C}$.*

*Setup. $\mathcal{C}$ invokes $\mathsf{SETUP}(1^\lambda)$ for a security parameter $1^\lambda$ and obtains a set of parameters $(\pi, \mathcal{U}, \mathcal{M}, \mathcal{S}, \mathcal{K}, \mathcal{F}, \mathsf{KGEN}, \{X_i\})$ and the corresponding set of secret keys $\{x_i\}$. $\mathcal{C}$ gives the set of parameters to $\mathcal{A}$ while retaining the set of secret keys $\{x_i\}$.*

*Queries. $\mathcal{A}$ is allowed to issue to following queries in an adaptive manner.*

1. *$\mathsf{KGEN}$: $\mathcal{C}$ randomly generates $k \in_R \mathcal{K}$ and returns $f = \mathsf{KGEN}(k)$ to $\mathcal{A}$.*
2. *Keystone Reveal: On input $f$ such that $f$ is an output of the $\mathsf{KGEN}$ query, $\mathcal{C}$ returns $k$ such that $f = \mathsf{KGEN}(k)$. Otherwise $\mathcal{C}$ returns $\perp$.*
3. *$\mathsf{ASIGN}$: On input $(X_i, X_j, h_2, M)$ such that $X_i, X_j \in \{X_i\}$, $h_2 \in \mathcal{F}$, $M \in \mathcal{M}$, $\mathcal{C}$ replies with $\mathsf{ASIGN}(X_i, X_j, x_i, h_2, M)$.*
4. *Secret Key Reveal: On input $X_i \in \{X_i\}$, $\mathcal{C}$ returns $x_i$.*

*Output. Finally $\mathcal{A}$ outputs a signature $\sigma^* = (s^*, h^*, f^*)$, a message $M^*$ and two public keys $X_c^*$, $X_d^*$. $\mathcal{A}$ wins the game if $\mathsf{AVERIFY}(\sigma^*, X_c^*, X_d^*, M^* = 1)$ and either one of the following is true:*

- *$(X_c^*, X_d^*, f^*, M^*)$ is not an input to the $\mathsf{ASIGN}$ query and $X_c^*$, $X_d^*$ is not an input to the secret key reveal query.*
- *$\mathcal{A}$ has not made any $\mathsf{ASIGN}$ query of the form $(X_c^*, X, f^*, M^*)$ for all $X \in \{X_i\} \setminus \{X_c^*\}$, no secret key reveal query was made with input $X_c^*$ and $f^*$ is the output of $\mathsf{KGEN}$ query or $\mathcal{A}$ also outputs $k^*$ such that $f^* = \mathsf{KGEN}(k^*)$.*

**Ambiguity.** The following game is used to capture ambiguity of a concurrent signature.

**Definition 2 (Ambiguity).** *A concurrent signature is ambiguous if no PPT adversary $\mathcal{A}$ can win the following game with a challenger $\mathcal{C}$.*

*Setup. Same as Setup in the game in Definition 1.*

*Phase 1. $\mathcal{A}$ is allowed to made a sequence of $\mathsf{KGEN}$, Keystone Reveal, $\mathsf{ASIGN}$ and Secret Key Reveal query, which are answered as in the game of subsection 2.3.*

*Challenge. $\mathcal{A}$ outputs two public keys $X_i$, $X_j$ and a message $M$ as challenge. $\mathcal{C}$ randomly picks $k \in_R \mathcal{K}$ and computes $f = \mathsf{KGEN}(k)$. $\mathcal{C}$ then flips a fair coin $b \in_R \{0, 1\}$. If $b = 0$, $\mathcal{C}$ computes $\sigma_0 = \mathsf{ASIGN}(X_i, X_j, x_i, f, M)$. Otherwise, $\mathcal{C}$ computes $(s, h, f) = \mathsf{ASIGN}(X_j, X_i, x_j, f, M)$ and parse $\sigma_1$ as $(s, f, h)$. $\mathcal{C}$ returns $\sigma_b$ to $\mathcal{A}$.*

*Phase 2. $\mathcal{A}$ can make another sequence of queries as in phase 1.*

*Output. Finally $\mathcal{A}$ outputs a guess bit $b'$. $\mathcal{A}$ wins the game if $b = b'$ and $\mathcal{A}$ did not make any Keystone Reveal query on input $f$ or $h$.*

**Fairness.** The following game is used to capture fairness of a concurrent signature.

**Definition 3 (Fairness).** *A concurrent signature is fair if no PPT adversary $\mathcal{A}$ can win the following game with a challenger $\mathcal{C}$.*

*Setup. Same as Setup in the game of subsection 2.3*

*Queries. $\mathcal{A}$ is allowed to made a sequence of $\mathsf{KGEN}$, Keystone Reveal, $\mathsf{ASIGN}$ and Secret Key Reveal query, which are answered as in the game of subsection 2.3.*

*Challenge. $\mathcal{A}$ outputs two public keys $X_i$, $X_j$ and two messages $M_i$, $M_j$, together with $\sigma_i = (s_i, h_1, h_2)$ such that $\mathsf{AVERIFY}((s_i,\ h_1,\ h_2),\ X_i,\ X_j,\ M_i) = 1$. $\mathcal{C}$ returns $\sigma_j = (s_j, h_3, h_2) = \mathsf{ASIGN}(X_j, X_i, x_j, h_2, M_j)$.*

*Output. Finally $\mathcal{A}$ either outputs a value $k$. $\mathcal{A}$ wins the game if $f = \mathsf{KGEN}(k)$ such that $f$ was a previous output from $\mathsf{KGEN}$ query and no Keystone Reveal query on $f$ was made.*

## 3 Abusing Fairness in Concurrent Signatures

In this section, we discuss the various advantage levels enjoyed by the initial signer, their implications and how they could be acquired. At the high level, it is often the case that exhibiting such advantage requires the use of zero-knowledge proof [9], which is reviewed as follows.

## 3.1 Zero-Knowledge Proof

A zero-knowledge proof [9] is an interactive protocol for one party, the prover, to prove to another party, the verifier, that some statement is true, without revealing anything other than the veracity of the statement. In [8], it has been shown that, assuming the existence of one-way function, one can create a zero-knowledge proof system for the NP-complete graph coloring problem with three colors. Since every problem in NP can be efficiently reduced to this problem, it means that all problems in NP have zero-knowledge proofs. Later in [5], it has been shown that anything that can be proved by an interactive proof system can be proved with zero knowledge.

## 3.2 Advantage Level 0

Level 0 advantage is inherent in concurrent signatures, as the initial signer, Alice, is in possession of the keystone, which is under the full control of Alice on when and whether the keystone will be released to the public. Thus, the primitive is not suitable if the matching signer will be at a disadvantage if withholding the keystone or delaying the release of the keystone would cause harm to the matching signer.

**The Implications.** Consider the tendering systems as discussed in the seminal paper of Chen et al. [6]. They described a scenario where $A$ has a bridge-building contract that she would like to put out to tender. Suppose there are two companies $B$ and $C$ that wish to put in proposals to win this contract. In this scenario, $B$ acts as the initial signer, as this is to prohibit $A$ to show this contract to $C$ to get a better proposal from $C$. We note that in this particular scenario, $B$ has the full control of the keystone, since the keystone is selected by $B$. Therefore, if at the end of the tender, if $A$ would like to select $B$ as the winner of the tender, $B$ may still have the liberty for not completing the contract by not releasing the keystone, and hence, it is unfair to $A$.

## 3.3 Advantage Level 1

**The Abuse.** Assume Alice is a malicious initial signer, whose purpose is to convince a third party Carol that the matching signer Bob and herself are about to exchange signatures on messages $M_A$ and $M_B$. Assume Alice and Bob have completed step 3 of the concurrent signature protocol (as described in Section 2.2). That is, Alice is in possession of a keystone $k$ and the ambiguous signature from Bob $\sigma_B := (s_B, h_B, f)$ on message $M_B$. At the same time, Bob is in possession of Alice's ambiguous signature $\sigma_A := (s_A, h_A, f)$ on message $M_A$. In our generic attack, Alice reveals $\sigma_A$, $\sigma_B$, $M_A$, $M_B$ to Carol and then conducts a zero-knowledge proof-of-knowledge of the value $k$ with Carol such that

$$f = \mathsf{KGEN}(k).$$

Carol is convinced that Alice and Bob are exchanging signatures on messages $M_A$ and $M_B$ and that Alice has the ability to complete the transaction.

**The Implications.** Consider an open auction [17, 18] in which Alice's ambiguous signature is a contract to sell a certain goods while Bob's ambiguous signature is a contract of a bid. Level-1 advantage allows Alice to convince Carol that she has the ability to seal the contract with Bob and the bid is specified in $M_B$. This allows Alice to safely urge Carol for a higher bid, and Bob is at a disadvantage.

We note that this implication is also applicable to the tendering systems as discussed previously. However, in this scenario, let us consider the case where $A$, who would like to put the bridge-building tender, is the initial signer. Hence, the company $B$ and $C$ will act as the matching signers. In this setting, $A$ will take the advantage level 1 to convince $C$ about $B$'s tender, so that $C$ will increase the value of her tender, and hence, disadvantaging $B$.

### 3.4 Advantage Level 2

**The Abuse.** Assume Alice is a malicious initial signer, whose purpose is to convince a third party Carol that the matching signer Bob has committed to message $M_B$. Assume Alice and Bob have completed step 3 of the concurrent signature protocol (as described in Section 2.2). That is, Alice is in possession of a keystone $k$ and the ambiguous signature from Bob $\sigma_B := (s_B, h_B, f)$ on message $M_B$. At the same time, Bob is in possession of Alice's ambiguous signature $\sigma_A := (s_A, h_A, f)$ on message $M_A$. Our observation on the incompleteness of the original fairness definition in [6] arises from the fact that to convince Carol about Bob's commitment to $M_B$ *does not necessarily* involve outputting some maliciously crafted keystone $\hat{k}$. Specifically, in our generic attack, Alice conducts a zero-knowledge proof-of-knowledge of the tuple $(k, \sigma_B)$ with Carol such that the following statements are true:

1. $f = \mathsf{KGEN}(k)$, and
2. $\mathsf{AVERIFY}(\sigma_B, X_A, X_B, M_B) = 1$.

This would be sufficient to convince a third party about *Bob's intention* to sign a message $M_B$, without revealing anything about Alice's intention, thus undermining the fairness guarantees of the concurrent signature schemes, and put Bob into disadvantage.

**The Implications.** Note that in the zero-knowledge proof, Alice does not reveal the keystone $k$, the keystone fix $f$, nor Bob's ambiguous signature. Thus, even if Carol is presented with the secret key of Bob, the ambiguous signatures $\sigma_A$ and $\sigma_B$ from Bob, she could not conclude that Alice has committed to $M_A$. In other words, the only thing that Carol can be assured of is that Alice is in possession of a signature from Bob on message $M_B$. Bob is left at an unfair position.

We would like to remark that both level 1 and 2 advantages are outside the security definition presented in Section 2.3. We also do not claim that existing concurrent signature schemes are broken for two reasons. Firstly, they are not within the security model. Secondly, even though they are theoretically possible, it is not always feasible. Thus, people should bear in mind any concurrent signature following this syntax *could* be abused by the initial signer, and hence, the adoption of concurrent signatures in application scenario should be examined to make sure that either the level 1 and 2 advantages of the initial signer are acceptable or that the advantages cannot be claimed efficiently.

## 4 Abusing Fairness in Asymmetric Concurrent Signatures

In this section, we demonstrate a practical abuse in advantage level 1 of the asymmetric concurrent signature due to Nguyen [13]. In this abuse, the initial signer can convince any verifier that he/she has the ability to make both ambiguous signatures verifiable. We would like to stress again that the attack is *outside* their original model and we therefore do not claim that we break Nguyen's original scheme.

### 4.1 Review of Nguyen's asymmetric concurrent signatures

For completeness, we will first review Nguyen's scheme.

*Setup.* Let $\mathbb{G} = \langle g \rangle$ be a group of prime order $p$. The key pair of Alice and Bob are respectively $(X_A = g^{x_A}, x_A) \in \mathbb{G} \times \mathbb{Z}_p$, $(X_B = g^{x_B}, x_B) \in \mathbb{G} \times \mathbb{Z}_p$. Let $H : \mathbb{G} \times \{0,1\}^* \to \mathbb{Z}_p$ be a hash function that would be modeled as random oracle.

*Generation of Alice's ambiguous signature.* On input a message $M_A \in \{0,1\}^*$ and Bob's public key $X_B$, Alice randomly generates $r \in_R \mathbb{Z}_p$ and computes $c = H(g^r, M_A)$, $s = g^{r+cx_A}$. Alice sets her ambiguous signature as $(c, s)$ and sends it to Bob. The keystone is defined as $k = r + cx_A$ such that $s = g^k$.

*Generation of Bob's ambiguous signature.* On input a message $M_B \in \{0,1\}^*$ and an ambiguous signature $(c, s)$ on message $M_A$ from Alice, Bob first check if $c = H(sX_A^{-c}, M_A)$. Bob continues if the check is successful. He computes $s' = s^{x_B}$, $c' = H(s'g^{r'}, M_B)$ and $k' = \frac{r'-c'}{x_B}$. He sets his ambiguous signature as $(c', s', k')$ and sends it to Alice.

*Binding of both signatures.* Upon receiving $(c', s', k')$ from Bob, Alice first checks if

$$c' = H(g^{c'} X_B^{k'} s', M_B)$$

and $s' = X_B^k$. If the check is successful and if Alice decides to have both signatures binded, she releases the value $k$. Both signatures are now publicly verifiable by checking the following verification equations.

Verification of Alice's signature:

$$c \overset{?}{=} H(X_A^{-c} s, M_A) \ \wedge \ s \overset{?}{=} g^k$$

Verification of Bob's signature:

$$c' \overset{?}{=} H(g^{c'} X_B^{k'} s', M_B) \ \wedge \ s' \overset{?}{=} X_B^k$$

*Remark.* Nguyen's construction is asymmetric in the sense that Alice's signature and Bob's signature are of different form with different verification equations.

### 4.2 A Concrete Level-1 Abuse on Nguyen's Scheme

Given the pair of ambiguous signatures $(c, s)$, $(c', s', k')$ on message $M_A$ and $M_B$ respectively, Alice, who is in possession of the keystone $k$ satisfying the relationship $s = g^k$ and $s' = X_B^k$ can convince a third party Carol that she has the ability to bind both ambiguous signatures by proving she knows the value $k$ in zero-knowledge. The full proof protocol is shown below.

1. Alice sends both $(c, s)$, $(c', s', k')$ $M_A$, $M_B$ to Carol.
2. Carol checks $c = H(X_A^{-c}s, M_A)$ and $c' = H(g^{c'}X_B^{k'}s', M_B)$. If yes, she randomly generates two values $t_1, t_2 \in_R \mathbb{Z}_p$ and sends $T_0 = g^{t_1}h^{t_2}$ to Alice.
3. Alice randomly generates a value $t \in_R \mathbb{Z}_p$, computes $T_1 = g^t$, $T_2 = X_B^t$ and sends $T_1, T_2$ to Carol.
4. Carol sends $t_1, t_2$ to Alice
5. Alice checks if $T_0 = g^{t_1}h^{t_2}$. If yes, she computes $z = t - t_1 k$ and sends $z$ to Carol.
6. Carol checks if $T_1 = s^{t_1}g^z$ and $T_2 = {s'}^{t_1}X_B^z$ and accepts the the proof if both equations hold.

## 5 Conclusion

We pointed out the advantage gained by the initial signer in a concurrent signature scheme, and classified into three levels. In fact, any concurrent signature satisfying Chen, Kudla and Paterson's syntax can be abused in different ways. This is a very important observation in particular where concurrent signatures are used in different scenarios. Cautions must be exercised when concurrent signatures are to be adopted in real applications to ensure either the matching signer can accept the inherent unfairness of concurrent signatures or it is hard for the initial signer to claim the advantage. In particular, we demonstrated that concurrent signatures are not suitable for tendering systems (in contrast to the seminal paper of Chen et al. [6]).

**Acknowledgements.** We would like to thank anonymous reviewers of ACISP 2013 who have helped us to improve the clarity and quality of this paper.

## References

1. Abe, M., Ohkubo, M., Suzuki, K.: 1-out-of-n Signatures from a Variety of Keys. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 415–432. Springer, Heidelberg (2002)
2. Asokan, N., Schunter, M., Waidner, M.: Optimistic protocols for fair exchange. In: Proc. 4th ACM Conference on Computer and Communications Security, pp. 8–17 (1997)
3. Asokan, N., Shoup, V., Waidner, M.: Optimistic fair exchange of digital signatures. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 591–606. Springer, Heidelberg (1998)

4. Asokan, N., Shoup, V., Waidner, M.: Optimistic fair exchange of digital signatures. IEEE Journal of Selected Areas Communications 18(4), 593–610 (2000)
5. Ben-Or, M., Goldreich, O., Goldwasser, S., Håstad, J., Kilian, J., Micali, S., Rogaway, P.: Everything provable is provable in zero-knowledge. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 37–56. Springer, Heidelberg (1990)
6. Chen, L., Kudla, C., Paterson, K.G.: Concurrent signatures. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 287–305. Springer, Heidelberg (2004)
7. Dodis, Y., Lee, P.J., Yum, D.H.: Optimistic fair exchange in a multi-user setting. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 118–133. Springer, Heidelberg (2007), Also at Cryptology ePrint Archive, Report 2007/182
8. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. J. ACM 38, 690–728 (1991)
9. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM J. Comput. 18(1), 186–208 (1989)
10. Huang, Q., Wong, D.S., Susilo, W.: Efficient designated confirmer signature and DCS-based ambiguous optimistic fair exchange. IEEE Transactions on Information Forensics and Security 6(4), 1233–1247 (2011)
11. Huang, Q., Wong, D.S., Susilo, W.: The construction of ambiguous optimistic fair exchange from designated confirmer signature without random oracles. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 120–137. Springer, Heidelberg (2012)
12. Huang, Q., Yang, G., Wong, D.S., Susilo, W.: Ambiguous optimistic fair exchange. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 74–89. Springer, Heidelberg (2008)
13. Nguyen, K.: Asymmetric concurrent signatures. In: Qing, S., Mao, W., López, J., Wang, G. (eds.) ICICS 2005. LNCS, vol. 3783, pp. 181–193. Springer, Heidelberg (2005)
14. Susilo, W., Mu, Y., Zhang, F.: Perfect concurrent signature schemes. In: López, J., Qing, S., Okamoto, E. (eds.) ICICS 2004. LNCS, vol. 3269, pp. 14–26. Springer, Heidelberg (2004)
15. Tan, X., Huang, Q., Wong, D.S.: Concurrent signature without random oracles. Cryptology ePrint Archive, Report 2012/576 (2012), `http://eprint.iacr.org/`
16. Tonien, D., Susilo, W., Safavi-Naini, R.: Multi-party concurrent signatures. In: Katsikas, S.K., López, J., Backes, M., Gritzalis, S., Preneel, B. (eds.) ISC 2006. LNCS, vol. 4176, pp. 131–145. Springer, Heidelberg (2006)
17. Wang, G., Bao, F., Zhou, J.: The fairness of perfect concurrent signatures. In: Ning, P., Qing, S., Li, N. (eds.) ICICS 2006. LNCS, vol. 4307, pp. 435–451. Springer, Heidelberg (2006)
18. Yuen, T.H., Wong, D.S., Susilo, W., Huang, Q.: Concurrent signatures with fully negotiable binding control. In: Boyen, X., Chen, X. (eds.) ProvSec 2011. LNCS, vol. 6980, pp. 170–187. Springer, Heidelberg (2011)

# Formalizing Group Blind Signatures and Practical Constructions without Random Oracles

Essam Ghadafi

University of Bristol, United Kingdom

**Abstract.** Group blind signatures combine anonymity properties of both group signatures and blind signatures and offer privacy for both the message to be signed and the signer. The primitive has been introduced with only informal definitions for its required security properties. In this paper, we offer two main contributions: first, we provide foundations for the primitive and present formal security definitions. In the process, we identify and address some subtle issues which were not considered by previous constructions and (informal) security definitions. Our second main contribution is a generic construction that yields practical schemes with a round-optimal signing protocol and constant-size signatures. Our constructions permit dynamic and concurrent enrollment of new members and satisfy strong security requirements. To the best of our knowledge, our schemes are the first provably secure constructions in the standard model. In addition, we introduce some new building blocks which may be of independent interest.

**Keywords:** Blind signatures, group signatures, group blind signatures.

## 1 Introduction

**Background**. Group signatures, introduced by Chaum and van Heyst [9], allow a member of a group to sign a message anonymously on behalf of the group with the option for a tracing authority to revoke anonymity. A Blind Signature ($\mathsf{BS}$), introduced by Chaum [8], allows a user to obtain a signature on a hidden message. The security of blind signatures [17,23] requires that the user is unable to fake new signatures for new messages (unforgeability) and that the signer does not learn the message he is signing nor be able to link a signature to its signing session (blindness).

A Group Blind Signature ($\mathsf{GBS}$), introduced by Lysyanskaya and Zulfikar [19], combines the properties of both a group signature scheme and a blind signature scheme and therefore it maintains the anonymity of the signer as well as the privacy of the message. Group blind signatures provide bi-directional privacy and are thus useful for many applications such as distributed e-cash systems, e.g. [19], where it is required that a digital coin reveals neither the identity of its holder nor that of the issuing bank/branch. Other non-exhaustive list of applications where the primitive is deployed include multi-authority e-voting and e-auction systems.

**Related Work**. The primitive was first introduced by Lysyanskaya and Zulfikar [19] where it was used to design a distributed e-cash system in which coins could be issued by different banks. Group blind signatures are much harder to construct than their traditional group signatures counterparts, especially in the standard model, and that is the reason only a few constructions were proposed [19,21] and all of them require random oracles [3]. The subtlety faced when designing such schemes lies in the dual privacy requirement. On the one hand, the signer needs to hide his identity and parts of the signature that could identify him (i.e. the anonymity requirement). On the other hand, the user needs to hide the message and parts of the signature which could lead to a linkage between a signature and its sign request (i.e. the blindness requirement).

The schemes in [19] are based on variants of the Camenisch-Stadler group signature [7]. Other schemes, e.g. [21], use divertible zero-knowledge proofs [22] to realize those conflicting anonymity requirements. A divertible proof allows a mediator to use a proof it got from a party to prove a statement to a third party. Constructions relying on such proofs require many rounds of interaction in the signing protocol and/or the Fiat-Shamir transformation [10] to eliminate the interaction and hence lying in the random oracle model.

**Our Contribution**. Our first contribution is a formal security model for the primitive. Providing such a model would allow for proving the security of constructions formally and more rigorously. In the process, we identify and address some subtle issues which were not considered by previous constructions and informal security definitions.

Our second contribution is a generic construction which yields practical schemes in the standard model. We exploit new properties of Groth-Sahai proofs [16] and show how to transform a witness-indistinguishable proof into a new witness-indistinguishable/zero-knowledge proof without knowledge of the witness of the original proof. Such observations are useful and may be of independent interest. These observations combined with other properties of the proofs is what allows us to efficiently realize the subtle dual privacy requirement required for the blind signing protocol. We apply the new techniques to some of the recent Groth-Sahai based instantiations of Fischlin's generic construction [11] for obtaining round-optimal blind signatures. By transforming those blind signing protocols, exploiting the different properties of the proofs, combined with suitable and compatible building blocks we construct our schemes.

We provide two example instantiations of the construction. The first instantiation relies solely on falsifiable intractability assumptions [20]. To improve the efficiency, our second instantiation uses a new structure-preserving signature scheme [1] based on a variant of the standard LRSW assumption [18] which we show holds in the generic group model [24]. All our constructions are round-optimal, yield constant-size signatures, and allow for members of the group to join dynamically and concurrently. Moreover, their security is proven in the standard model. We start by showing how to construct CPA-anonymous schemes and then outline how they can be extended to provide full anonymity. We also provide a proof of security for our constructions.

**Paper Organization**. The rest of the paper is organized as follows: In Section 2, we give some preliminary definitions. In Section 3 we define dynamic group blind signatures and provide their security model. We list the building blocks we use in Section 4. In Section 5, we present our constructions and provide a proof of their security. In Section 6, we outline how we can achieve full anonymity.

## 2 Preliminaries

**Notation**. A function $\nu(.) : \mathbb{N} \rightarrow \mathbb{R}^+$ is negligible in $c$ if for every polynomial $p(.)$ and all sufficiently large values of $c$, it holds that $\nu(c) < \frac{1}{p(c)}$. Given a probability distribution $S$, we denote by $y \leftarrow S$ the operation of selecting an element according to $S$. If $A$ is a probabilistic machine, we denote by $A(x_1, \ldots, x_n)$ the output distribution of $A$ on inputs $(x_1, \ldots, x_n)$. By PPT we mean running in probabilistic polynomial time in the relevant security parameter. By $[1, n]$, we mean the set $\{1, 2, \ldots, n\}$. We denote by $\langle A, B\rangle$ an interactive protocol involving algorithms $A$ and $B$. Occasionally we will use the notation $\langle A, B\rangle^i$ for $i \in \mathbb{N}$ denoting the number of times such an interactive protocol is allowed to take place. If $i = *$, such an interaction can be invoked unlimited number of times.

**Bilinear Groups**. A bilinear group is a tuple $\mathcal{P} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, G_1, G_2)$ where $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ are groups of a prime order $p$ and $G_1$ and $G_2$ generate $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. The function $\hat{e}$ is a non-degenerate bilinear map $\mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$. We will use multiplicative notation for all the groups although usually $\mathbb{G}_1$ and $\mathbb{G}_2$ are chosen to be additive groups. We let $\mathbb{G}^\times := \mathbb{G} \setminus \{1_\mathbb{G}\}$.

We use *asymmetric* bilinear groups (which are more efficient), for which there are no known efficiently computable homomorphisms from $\mathbb{G}_1$ to $\mathbb{G}_2$ or vice versa. We assume an algorithm $\mathsf{BGrpSetup}$ which takes as input a security parameter $\lambda$ and produces a description of bilinear groups $\mathcal{P}$.

**Complexity Assumptions**. In this paper we use the following assumptions:

**SXDH.** The DDH assumption holds in both groups $\mathbb{G}_1$ and $\mathbb{G}_2$.

**AWFCDH** [1]. Given $(G_1, G_1^a, G_2) \in {\mathbb{G}_1^\times}^2 \times \mathbb{G}_2^\times$ for $a \leftarrow \mathbb{Z}_p$, it is hard to output a tuple $(G_1^b, G_1^{ab}, G_2^b, G_2^{ab}) \in {\mathbb{G}_1^\times}^2 \times {\mathbb{G}_2^\times}^2$ for an arbitrary $b \in \mathbb{Z}_p$.

**q-ADHSDH** [1]. Given $(G_1, F, K, G_1^x, G_2, G_2^x) \in {\mathbb{G}_1^\times}^4 \times {\mathbb{G}_2^\times}^2$ and $q - 1$ tuples $(A_i := (K \cdot G_1^{r_i})^{\frac{1}{x+c_i}}, C_{1,i} := F^{c_i}, C_{2,i} := G_2^{c_i}, R_{1,i} := G_1^{r_i}, R_{2,i} := G_2^{r_i})_{i=1}^{q-1}$, where $x, c_i, r_i \leftarrow \mathbb{Z}_p$, it is hard to output a new tuple $(A^*, C_1^*, C_2^*, R_1^*, R_2^*)$.

**DH-LRSW.** This a new variant of the LRSW assumption [18] that we introduce in this paper. It stands for Dual-Hidden LRSW and states that given $(G_2^x, G_2^y)$ for random $(x, y) \leftarrow \mathbb{Z}_p^2$ and access to an oracle $\mathsf{O}_{X,Y}(\cdot)$ that, on input a pair $(M_1, M_2) \in \mathbb{G}_1 \times \mathbb{G}_2$ satisfying $\hat{e}(M_1, G_2) = \hat{e}(G_1, M_2)$, picks a random $a \leftarrow \mathbb{Z}_p$ and outputs a DH-LRSW tuple $(G_1^a, G_1^{ay}, M_1^{ay}, G_1^{ax} \cdot M_1^{axy})$, it is hard to compute a DH-LRSW tuple for $(M_1^*, M_2^*)$ that was never queried to the oracle. We show in the full version of the paper [13] that the assumption holds in the generic group model [24].

## 3 Dynamic Group Blind Signatures

Our model builds on the security models for dynamic group signatures [4] and blind signatures [17,23]. The parties involved in a group blind signature are: an authority called the $\mathsf{Issuer}$ who controls who can join the group, an authority called the $\mathsf{Opener}$ who can open signatures and reveal who signed them. A number of signers $\mathsf{Signer}_i$ each of which has a unique identity and can sign on behalf of the group once they have joined the group. External users $\mathsf{User}_i$ which can ask for messages to be blindly signed by members of the group.

In our definition, we do not require that users (i.e. entities who request signatures) are traceable (unless the identifying information is embedded in the messages themselves) and thus we do not assign keys to them. However, the model can be extended to provide that functionality.

A group blind signature scheme $\mathsf{GBS}$ is a tuple of polynomial-time algorithms

$$\mathsf{GBS} := (\mathsf{GKg}, \mathsf{SKg}, \langle \mathsf{Join}, \mathsf{Issue}\rangle, \langle \mathsf{Obtain}, \mathsf{Sign}\rangle, \mathsf{GVf}, \mathsf{Open}, \mathsf{Judge}),$$

$\mathsf{GKg}$ takes as input a security parameter $\lambda$ and generates the group public key $\mathsf{gpk}$, the $\mathsf{Issuer}$'s key $\mathsf{ik}$ and the $\mathsf{Opener}$'s key $\mathsf{ok}$.

$\mathsf{SKg}$ generates a pair of personal secret/public keys $(\mathbf{ssk}[i], \mathbf{spk}[i])$ for a potential group member $\mathsf{Signer}_i$. We assume that the public key table $\mathbf{spk}$ is publicly accessible (possibly via some PKI).

$\langle \mathsf{Join}(\mathsf{gpk}, i, \mathbf{ssk}[i]), \mathsf{Issue}(\mathsf{ik}, i, \mathbf{spk}[i])\rangle$ is an interactive protocol between a signer $\mathsf{Signer}_i$ and the $\mathsf{Issuer}$. After a successful completion of this protocol, $\mathsf{Signer}_i$ becomes a member of the group. If successful, the final state of the $\mathsf{Issue}$ algorithm is stored in the registration table at index $i$ (i.e. $\mathbf{reg}[i]$), whereas that of the $\mathsf{Join}$ algorithm is stored in $\mathbf{gsk}[i]$. We assume that the communication in this protocol takes place over a secure (i.e. private and authentic) channel.

$\langle \mathsf{Obtain}(\mathsf{gpk}, m), \mathsf{Sign}(\mathbf{gsk}[i])\rangle$ is an interactive protocol between a user $\mathsf{User}$ and an anonymous member of the group. If the protocol completes successfully, $\mathsf{User}$ obtains a blind signature $\Sigma$ on the message $m$. If any of the parties abort, $\mathsf{User}$ outputs $\perp$.

$\mathsf{GVf}(\mathsf{gpk}, m, \Sigma)$ is a deterministic algorithm to verify if $\Sigma$ is a valid signature on the message $m$ w.r.t. the group public key $\mathsf{gpk}$. It outputs 1 or 0 accordingly.

$\mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathbf{reg}, m, \Sigma)$ is a deterministic algorithm in which the $\mathsf{Opener}$ uses his key $\mathsf{ok}$ to identify the identity $i$ of the signer form the group blind signature $\Sigma$ and produces a proof $\tau$ attesting to this claim.

$\mathsf{Judge}(\mathsf{gpk}, i, \mathbf{spk}[i], m, \Sigma, \tau)$ is a deterministic algorithm which verifies whether or not the owner of $\mathbf{spk}[i]$ has indeed produced the signature $\Sigma$ returning 0 or 1 accordingly.

**Security of Dynamic Group Blind Signatures**. The security properties we require from a dynamic $\mathsf{GBS}$ scheme are: correctness, anonymity, traceability, non-frameability, and blindness. Those security requirements are formulated via a set of experiments in which the adversary has access to a set of oracles.

In the experiments, the following global lists are maintained: a set $\mathsf{HSL}$ of honest signers; a set $\mathsf{CSL}$ of corrupt signers whose keys have been chosen by the adversary and whose states have been learned by the adversary; a set $\mathsf{BSL}$ of bad signers whose secret keys have been revealed to the adversary; a set $\mathsf{CLA}$ containing the identities of the challenge signers used in the anonymity game; a list $\mathsf{CLB}$ containing pairs of challenge message-signature used in the blindness game; a table **reg** where the element $i$ in this table contains the registration information of the group member $\mathsf{Signer}_i$; a table **spk** where $\mathbf{spk}[i]$ contains the personal public key of the group member $\mathsf{Signer}_i$. The lists $\mathsf{HSL}$, $\mathsf{CSL}$, $\mathsf{BSL}$, $\mathsf{CLA}$ and $\mathsf{CLB}$ are empty at initialization, whereas the entries of the tables **reg** and **spk** are initialized to $\epsilon$. The oracles we use are:

– $\mathsf{AddS}(i)$ is used to add an honest signer $\mathsf{Signer}_i$ to the group.
– $\mathsf{CrptS}(i, \mathsf{pk})$ is used to create a new corrupt signer $\mathsf{Signer}_i$, where $\mathsf{Signer}_i$'s public key $\mathbf{spk}[i]$ is chosen by the adversary. This is usually called in preparation for calling the $\mathsf{SndToI}$ oracle.
– $\mathsf{SndToI}(i, M_{in})$ is used to engage in the $\mathsf{Join}$-$\mathsf{Issue}$ protocol with the honest, $\mathsf{Issue}$-executing $\mathsf{Issuer}$.
– $\mathsf{SndToS}(i, M_{in})$ models the scenario that the adversary has corrupted the $\mathsf{Issuer}$. The adversary uses this oracle to engage in the $\mathsf{Join}$-$\mathsf{Issue}$ protocol with an honest, $\mathsf{Join}$-executing $\mathsf{Signer}$.
– $\mathsf{ReadReg}(i)$ is used to obtain the content of entry $\mathbf{reg}[i]$.
– $\mathsf{ModifyReg}(i, val)$ is used to modify the content of entry $\mathbf{reg}[i]$ by setting $\mathbf{reg}[i] := val$.
– $\mathsf{SSK}(i)$ returns the personal secret key $\mathbf{ssk}[i]$ and the group signing key $\mathbf{gsk}[i]$ of group member $\mathsf{Signer}_i$.
– $\mathsf{OSign}(i)$ is an interactive oracle (i.e. the adversary must engage in an interaction with this oracle). If the interaction completes successfully, the adversary obtains a blind signature by member $\mathsf{Signer}_i$ on a message of its choice.
– $\mathsf{CH}_b(i_0, i_1)$ is a left-right oracle for defining anonymity and is only called once. The adversary sends a couple of identities $(i_0, i_1)$ and interacts with this oracle in order to produce a blind signature on a message of its choice by the group member $\mathsf{Signer}_{i_b}$ for $b \leftarrow \{0, 1\}$.
– $\mathsf{Open}(m, \Sigma)$ allows the adversary to ask for signatures to be opened.

We give the details of those oracles in Figure 1. We define the security requirements as follows where we use a set of experiments as in Figure 2.

**Correctness**. A dynamic $\mathsf{GBS}$ scheme is correct if: all correctly produced signatures are accepted by the $\mathsf{GVf}$ algorithm, the $\mathsf{Opener}$ is always able to identify the honest group member who produced a signature and the $\mathsf{Judge}$ algorithm always accepts the decision made by the $\mathsf{Opener}$.

Formally, a dynamic $\mathsf{GBS}$ scheme is correct if for all $\lambda \in \mathcal{N}$, all PPT adversaries $\mathcal{A}$ have a negligible advantage $\mathsf{Adv}^{Corr}_{\mathsf{GBS},\mathcal{A}}(\lambda) := \Pr[\mathsf{Exp}^{Corr}_{\mathsf{GBS},\mathcal{A}}(\lambda) = 1]$.

**Anonymity**. A dynamic $\mathsf{GBS}$ scheme is anonymous if given two members of its choice, the adversary is unable to tell which member produced a signature.

```
AddS(i):
  – If $i \in \mathsf{HSL} \cup \mathsf{CSL} \cup \mathsf{BSL}$ Then Return $\perp$.
  – $\mathsf{HSL} := \mathsf{HSL} \cup \{i\}$.
  – $(\mathbf{ssk}[i], \mathbf{spk}[i]) \leftarrow \mathsf{SKg}(1^\lambda)$.
  – $\mathsf{cert}_i := \perp$, $\mathsf{dec}^i_{\mathsf{Issue}} := \mathsf{cont}$.
  – $\mathsf{St}^i_{\mathsf{Join}} := (\mathsf{gpk}, i, \mathbf{ssk}[i])$, $\mathsf{St}^i_{\mathsf{Issue}} := (\mathsf{ik}, i, \mathbf{spk}[i])$.
  – $(\mathsf{St}^i_{\mathsf{Join}}, M_{\mathsf{Issue}}, \mathsf{dec}^i_{\mathsf{Join}}) \leftarrow \mathsf{Join}(\mathsf{St}^i_{\mathsf{Join}}, \perp)$.
  – While ($\mathsf{dec}^i_{\mathsf{Issue}} = \mathsf{cont}$ and $\mathsf{dec}^i_{\mathsf{Join}} = \mathsf{cont}$) Do
      • $(\mathsf{St}^i_{\mathsf{Issue}}, M_{\mathsf{Join}}, \mathsf{dec}^i_{\mathsf{Issue}}) \leftarrow \mathsf{Issue}(\mathsf{St}^i_{\mathsf{Issue}}, M_{\mathsf{Issue}})$.
      • $(\mathsf{St}^i_{\mathsf{Join}}, M_{\mathsf{Issue}}, \mathsf{dec}^i_{\mathsf{Join}}) \leftarrow \mathsf{Join}(\mathsf{St}^i_{\mathsf{Join}}, M_{\mathsf{Join}})$.
  – If $\mathsf{dec}^i_{\mathsf{Issue}} = \mathsf{accept}$ Then $\mathbf{reg}[i] := \mathsf{St}^i_{\mathsf{Issue}}$.
  – If $\mathsf{dec}^i_{\mathsf{Join}} = \mathsf{accept}$ Then $\mathbf{gsk}[i] := \mathsf{St}^i_{\mathsf{Join}}$.
  – Return $\mathbf{spk}[i]$.

SSK(i):
  – If $i \notin \mathsf{HSL} \setminus \{\mathsf{CSL} \cup \mathsf{BSL}\}$ Then Return $\perp$.
  – $\mathsf{BSL} := \mathsf{BSL} \cup \{i\}$.
  – Return $(\mathbf{gsk}[i], \mathbf{ssk}[i])$.

OSign(i):
  – If $i \in \mathsf{CSL} \cup \mathsf{BSL}$ Then Return $\perp$.
  – If $\mathbf{gsk}[i] = \epsilon$ Then Return $\perp$.
  – Call $\mathsf{Sign}(\mathbf{gsk}[i])$.

CH_b(i_0, i_1):
  – If $i_0$ or $i_1 \notin \mathsf{HSL} \cup \mathsf{BSL}$ Then Return $\perp$.
  – If $\mathbf{gsk}[i_0] = \epsilon$ or $\mathbf{gsk}[i_1] = \epsilon$ Then Return $\perp$.
  – $\mathsf{CLA} := \mathsf{CLA} \cup \{i_0, i_1\}$.
  – Call $\mathsf{Sign}(\mathbf{gsk}[i_b])$.

Open(m, Σ):
  – If $\mathsf{GVf}(\mathsf{gpk}, m, \Sigma) = 0$ Then Return $(\perp, \perp)$.
  – $(id, \tau) \leftarrow \mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathbf{reg}, m, \Sigma)$.
  – If Anonymity: Return $(\perp, \perp)$ if $id \in \mathsf{CLA}$.
  – If Blindness: Return $(\perp, \perp)$ if $(m, \Sigma) \in \mathsf{CLB}$.[1]
  – Return $(id, \tau)$.

CrptS(i, pk):
  – If $i \in \mathsf{HSL} \cup \mathsf{CSL} \cup \mathsf{BSL}$ Then Return $\perp$.
  – $\mathsf{CSL} := \mathsf{CSL} \cup \{i\}$.
  – $\mathbf{spk}[i] := \mathsf{pk}$.
  – $\mathsf{St}^i_{\mathsf{Issue}} := (\mathsf{ik}, i, \mathbf{spk}[i])$.
  – $\mathsf{dec}^i_{\mathsf{Issue}} := \mathsf{cont}$.
  – Return 1.

SndToI(i, M_in):
  – If $i \notin \mathsf{CSL} \cup \mathsf{BSL}$ Then Return $\perp$.
  – If $\mathsf{dec}^i_{\mathsf{Issue}} \neq \mathsf{cont}$ Then Return $\perp$.
  – If $\mathsf{St}^i_{\mathsf{Issue}}$ is undefined
      • $\mathsf{St}^i_{\mathsf{Issue}} := (\mathsf{ik}, i, \mathbf{spk}[i])$.
  – $(\mathsf{St}^i_{\mathsf{Issue}}, M_{\mathsf{out}}, \mathsf{dec}^i_{\mathsf{Issue}}) \leftarrow \mathsf{Issue}(\mathsf{St}^i_{\mathsf{Issue}}, M_{\mathsf{in}})$.
  – If $\mathsf{dec}^i_{\mathsf{Issue}} = \mathsf{accept}$ Then $\mathbf{reg}[i] := \mathsf{St}^i_{\mathsf{Issue}}$.
  – Return $(M_{\mathsf{out}}, \mathsf{dec}^i_{\mathsf{Issue}})$.

SndToS(i, M_in):
  – If $i \in \mathsf{CSL} \cup \mathsf{BSL}$ Then Return $\perp$.
  – If $i \notin \mathsf{HSL}$ Then
      • $\mathsf{HSL} := \mathsf{HSL} \cup \{i\}$.
      • $(\mathbf{ssk}[i], \mathbf{spk}[i]) \leftarrow \mathsf{SKg}(1^\lambda)$.
      • $\mathbf{gsk}[i] := \epsilon$, $M_{\mathsf{in}} := \epsilon$.
  – If $\mathsf{dec}^i_{\mathsf{Join}} \neq \mathsf{cont}$ Then Return $\perp$.
  – If $\mathsf{St}^i_{\mathsf{Join}}$ is undefined
      • $\mathsf{St}^i_{\mathsf{Join}} := (\mathsf{gpk}, i, \mathbf{ssk}[i])$.
  – $(\mathsf{St}^i_{\mathsf{Join}}, M_{\mathsf{out}}, \mathsf{dec}^i_{\mathsf{Join}}) \leftarrow \mathsf{Join}(\mathsf{St}^i_{\mathsf{Join}}, M_{\mathsf{in}})$
  – If $\mathsf{dec}^i_{\mathsf{Join}} = \mathsf{accept}$ Then $\mathbf{gsk}[i] := \mathsf{St}^i_{\mathsf{Join}}$.
  – Return $(M_{\mathsf{out}}, \mathsf{dec}^i_{\mathsf{Join}})$.

ReadReg(i):
  – Return $\mathbf{reg}[i]$.

ModifyReg(i, val):
  – $\mathbf{reg}[i] := val$.
```

**Fig. 1.** Oracles used in the security experiments for dynamic group blind signatures

We distinguish between CPA-Anonymity and Full Anonymity (FA). In the former the adversary is not granted access to an Open oracle, whereas in the latter it has access to such an oracle at any stage of the game with the exception that it may not be queried on the challenge signature. One can also consider a weaker non-adaptive variant of full anonymity which we refer to as Weak Full Anonymity (WFA) where the adversary can only ask Open queries before seeing the challenge signature.

The issue with defining full anonymity for group blind signatures is that the signing protocol is a blind one and unlike in group signatures, the challenge message-signature pair is only revealed by the adversary (playing the role of the user) at

[1] This assuming strong unforgeability. If weak unforgeability is required then this is replaced by the following check: Return $(\perp, \perp)$ if $(m, \cdot) \in \mathsf{CLB}$.

Experiment $\mathsf{Exp}^{Corr}_{\mathsf{GBS},\mathcal{A}}(\lambda)$

- $(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GKg}(1^\lambda)$; $\mathsf{HSL} := \emptyset$; $\mathsf{CSL} := \emptyset$; $\mathsf{BSL} := \emptyset$.
- $(i, m) \leftarrow \mathcal{A}(\mathsf{gpk} : \mathsf{AddS}(\cdot), \mathsf{ReadReg}(\cdot))$.
- If $i \notin \mathsf{HSL}$ or $\mathbf{gsk}[i] = \epsilon$ Then Return 0.
- $(\Sigma, \perp) \leftarrow \langle \mathsf{Obtain}(\mathsf{gpk}, m), \mathsf{Sign}(\mathbf{gsk}[i]) \rangle$.
- If $\mathsf{GVf}(\mathsf{gpk}, m, \Sigma) = 0$ Then Return 1.
- $(j, \tau) \leftarrow \mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathbf{reg}, m, \Sigma)$; If $i \neq j$ Then Return 1.
- If $\mathsf{Judge}(\mathsf{gpk}, i, \mathbf{spk}[i], m, \Sigma, \tau) \neq 1$ Then Return 1 Else Return 0.

Experiment $\mathsf{Exp}^{Anon\text{-}b}_{\mathsf{GBS},\mathcal{A}}(\lambda)$

- $(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GKg}(1^\lambda)$; $\mathsf{HSL} := \emptyset$; $\mathsf{CSL} := \emptyset$; $\mathsf{BSL} := \emptyset$; $\mathsf{CLA} := \emptyset$.
- $b^* \leftarrow \mathcal{A}^{\langle \cdot, \mathsf{CH}_b(\cdot,\cdot) \rangle^1}(\mathsf{gpk}, \mathsf{ik} : \mathsf{CrptS}(\cdot,\cdot), \mathsf{SndToS}(\cdot,\cdot), \mathsf{ModifyReg}(\cdot,\cdot), \mathsf{Open}(\cdot,\cdot), \mathsf{SSK}(\cdot))$.
- Return $b^*$.

Experiment $\mathsf{Exp}^{Trace}_{\mathsf{GBS},\mathcal{A}}(\lambda)$

- $(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GKg}(1^\lambda)$; $\mathsf{HSL} := \emptyset$; $\mathsf{CSL} := \emptyset$; $\mathsf{BSL} := \emptyset$.
- $(m, \Sigma) \leftarrow \mathcal{A}(\mathsf{gpk}, \mathsf{ok} : \mathsf{AddS}(\cdot), \mathsf{CrptS}(\cdot,\cdot), \mathsf{SndToI}(\cdot,\cdot), \mathsf{ReadReg}(\cdot), \mathsf{SSK}(\cdot))$.
- If $\mathsf{GVf}(\mathsf{gpk}, m, \Sigma) = 0$ Then Return 0.
- $(i, \tau) \leftarrow \mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathbf{reg}, m, \Sigma)$.
- If $i = 0$ or $\mathsf{Judge}(\mathsf{gpk}, i, \mathbf{spk}[i], m, \Sigma, \tau) = 0$ Then Return 1 Else Return 0.

Experiment $\mathsf{Exp}^{Non\text{-}Frame}_{\mathsf{GBS},\mathcal{A}}(\lambda)$

- $(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GKg}(1^\lambda)$; $\mathsf{HSL} := \emptyset$; $\mathsf{CSL} := \emptyset$; $\mathsf{BSL} := \emptyset$.
- $(id, (m_i, \Sigma_i)_{i=1}^{l+1}) \leftarrow \mathcal{A}^{\langle \cdot, \mathsf{OSign}(\cdot) \rangle^*}(\mathsf{gpk}, \mathsf{ok}, \mathsf{ik} : \mathsf{CrptS}(\cdot,\cdot), \mathsf{SndToS}(\cdot,\cdot), \mathsf{ModifyReg}(\cdot,\cdot), \mathsf{SSK}(\cdot))$.
- If $id \notin \mathsf{HSL} \setminus \mathsf{BSL}$ or $\mathbf{gsk}[id] = \epsilon$ Then Return 0.
- If all of the following conditions are satisfied Then Return 1 Else Return 0:
  - $\mathsf{GVf}(\mathsf{gpk}, m_j, \Sigma_j) = 1$ for all $j \in [1, l+1]$.
  - $(id_j, \tau_j) \leftarrow \mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathbf{reg}, m_j, \Sigma_j)$; $id = id_j$ for all $j \in [1, l+1]$.
  - $\mathsf{Judge}(\mathsf{gpk}, id, \mathbf{spk}[id], m_j, \Sigma_j, \tau_j) = 1$ for all $j \in [1, l+1]$.
  - $\mathcal{A}$ interacted with $\mathsf{OSign}(id)$ no more than $l$ times.
  - $\forall i, j$ where $1 \leq i, j \leq l+1$, we have that if $i \neq j$ then $m_i \neq m_j$.

Experiment $\mathsf{Exp}^{Blind\text{-}b}_{\mathsf{GBS},\mathcal{A}}(\lambda)$

- $(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GKg}(1^\lambda)$; $\mathsf{HSL} := \emptyset$; $\mathsf{CSL} := \emptyset$; $\mathsf{BSL} := \emptyset$; $\mathsf{CLB} := \emptyset$.
- $(m_0, m_1, \mathsf{state}_{\mathsf{find}}) \leftarrow \mathcal{A}_{\mathsf{find}}(\mathsf{gpk}, \mathsf{ik} : \mathsf{CrptS}(\cdot,\cdot), \mathsf{SndToS}(\cdot,\cdot), \mathsf{ReadReg}(\cdot), \mathsf{SSK}(\cdot), \mathsf{Open}(\cdot,\cdot))$.
- $\mathsf{state}_{\mathsf{sign}} \leftarrow \mathcal{A}_{\mathsf{sign}}^{\langle \mathsf{Obtain}(\mathsf{gpk}, m_b), \cdot \rangle^1, \langle \mathsf{Obtain}(\mathsf{gpk}, m_{1-b}), \cdot \rangle^1}(\mathsf{state}_{\mathsf{find}})$.
- Let $\Sigma_b$ and $\Sigma_{1-b}$ be the outputs of the above interactions on $m_b$ and $m_{1-b}$, respectively.
- If $\Sigma_0 = \perp$ or $\Sigma_1 = \perp$ or $\exists i \in \{b, 1-b\}$ s.t. $\mathsf{GVf}(\mathsf{gpk}, m_i, \Sigma_i) = 0$ Then $(\Sigma_0, \Sigma_1) := (\perp, \perp)$ Else $\mathsf{CLB} := \mathsf{CLB} \cup \{(m_b, \Sigma_b), (m_{1-b}, \Sigma_{1-b})\}$.
- $b^* \leftarrow \mathcal{A}_{\mathsf{guess}}(\mathsf{state}_{\mathsf{sign}}, \Sigma_0, \Sigma_1 : \mathsf{CrptS}(\cdot,\cdot), \mathsf{SndToS}(\cdot,\cdot), \mathsf{ReadReg}(\cdot), \mathsf{SSK}(\cdot), \mathsf{Open}(\cdot,\cdot))$.
- Return $b^*$.

**Fig. 2.** Security experiments for dynamic group blind signatures

the end of the signing interaction with the $\mathsf{CH}_b$ oracle. Therefore, it is problematic to identify the challenge signature the adversary obtained from interacting with the $\mathsf{CH}_b$ oracle. An adversary can trivially break anonymity by revealing a bogus message-signature pair different from the one it got from interacting with the $\mathsf{CH}_b$ oracle and then at a later stage queries the $\mathsf{Open}$ oracle on the original challenge signature.

In the definition we propose (shown in Figure 2), if the adversary queries the $\mathsf{Open}$ oracle on a signature that opens to a signer in the challenge list, the oracle returns a special symbol instead of returning the identity of the signer.

This restriction, which is similar to that used for IND-RCCA security [6] for encryption schemes, ensures that the $\mathsf{Open}$ oracle does not open a challenge signature. Our definition provides the adversary with strong capabilities, for instance, it can fully corrupt the $\mathsf{Issuer}$ and can ask for signers' personal secret keys/group signing keys to be revealed including the two signers it chooses for the challenge (and thus capturing full key exposure attacks). The definition is appropriate even for the case where the final signature is weakly unforgeable, i.e. given a signature on a message, anyone can generate a new signature on the same message without knowledge of the signing key. WLOG in order to simplify the resulting security proofs, we only allow the adversary a single invocation of the challenge oracle. We show in the full version [13] that this is sufficient by showing a reduction from an adversary that invokes the challenge oracle polynomially many times to one which is allowed a single invocation. We also provide discussions about alternative definitions in the full version [13].

Our definition of anonymity also captures unlinkability of signatures. If an adversary can link signatures by the same signer, it can break our notion of anonmity. Note that the adversary in our definition is allowed to learn the secret keys of any group member including the challenge signers it uses in calling the $\mathsf{CH}_b$ oracle. Thus, it can produce signatures on behalf of any group member. Therefore, unlike [19], we do not define unlinkability as a separate requirement.

Formally, a dynamic $\mathsf{GBS}$ scheme is anonymous if for all $\lambda \in \mathcal{N}$, all PPT adversaries $\mathcal{A}$ have a negligible advantage $\mathsf{Adv}^{Anon}_{\mathsf{GBS},\mathcal{A}}(\lambda)$ where $\mathsf{Adv}^{Anon}_{\mathsf{GBS},\mathcal{A}}(\lambda) := \left|\Pr[\mathsf{Exp}^{Anon-0}_{\mathsf{GBS},\mathcal{A}}(\lambda) = 1] - \Pr[\mathsf{Exp}^{Anon-1}_{\mathsf{GBS},\mathcal{A}}(\lambda) = 1]\right|$.

**Traceability**. A dynamic $\mathsf{GBS}$ scheme is traceable if the $\mathsf{Opener}$ is always able to identify the signer. Also, the honest $\mathsf{Opener}$ is able to produce a proof for his claim that will be accepted by the $\mathsf{Judge}$ algorithm. We require that the $\mathsf{Issuer}$ is honest because a dishonest $\mathsf{Issuer}$ will always be able to create dummy signers whose signatures cannot be traced. In addition, we require that the $\mathsf{Opener}$ is partially but not fully corrupt because a fully corrupt $\mathsf{Opener}$ can simply refuse to open signatures.

Formally, a dynamic $\mathsf{GBS}$ scheme is traceable if for all $\lambda \in \mathcal{N}$, all PPT adversaries $\mathcal{A}$ have a negligible advantage $\mathsf{Adv}^{Trace}_{\mathsf{GBS},\mathcal{A}}(\lambda) := \Pr[\mathsf{Exp}^{Trace}_{\mathsf{GBS},\mathcal{A}}(\lambda) = 1]$.

**Non-Frameability**. A dynamic $\mathsf{GBS}$ scheme is non-frameable if it is impossible to prove that a particular group member produced a signature unless the member has indeed produced the signature.

Note that since the signing is done blindly, the signer does not know what message he has signed. To capture this, we use a similar definition to that used for the unforgeability of blind signatures [17,23]. The adversary wins if it outputs $l+1$ signatures on $l+1$ distinct messages that all open to the same honest group member but the adversary only asked for $l$ signatures by that group member. This requirement should hold even if both the $\mathsf{Opener}$ and the $\mathsf{Issuer}$ are fully corrupt and that is the reason we give $\mathcal{A}$ access to both $\mathsf{ik}$ and $\mathsf{ok}$ keys. If strong unforgeability is required, we drop the condition that the messages have to be distinct from the definition.

Formally, a dynamic $\mathsf{GBS}$ scheme is non-frameable if for all $\lambda \in \mathcal{N}$, all PPT adversaries $\mathcal{A}$ have a negligible advantage $\mathsf{Adv}_{\mathsf{GBS},\mathcal{A}}^{Non-Frame}(\lambda)$ where the advantage is defined as $\mathsf{Adv}_{\mathsf{GBS},\mathcal{A}}^{Non-Frame}(\lambda) := \Pr[\mathsf{Exp}_{\mathsf{GBS},\mathcal{A}}^{Non-Frame}(\lambda) = 1]$.

**Blindness**. A dynamic $\mathsf{GBS}$ scheme is blind if the adversary is unable to tell which message it is signing. Also, the adversary cannot link a signature to its signing session. In traditional blind signatures [17], blindness is defined via a game in which the adversary freely chooses two messages, and then after interacting with an honest $\mathsf{Obtain}$ oracle that requests signatures on those two messages in an arbitrary order (unknown to the adversary), the adversary wins if it correctly guesses the order in which the two messages were signed.

To capture the case that group members (including the $\mathsf{Issuer}$) might collude to break blindness, the adversary is allowed to use different (possibly corrupt) keys in producing the challenge signatures. This definition would then also imply the anonymity requirement, i.e. if a malicious signer from the group can recognize a signature he has produced then he can trivially break blindness. Also, unlike [25] which necessitates that the group must be static, we only require that both challenge signatures verify w.r.t. the same group public key. Otherwise, the adversary can trivially break blindness.

In the definition (in Figure 2), we equip the adversary with strong capabilities such as corrupting the $\mathsf{Issuer}$ as well as corrupting and/or learning the personal secret key/group signing key of any group member. However, the adversary is denied access to the $\mathsf{Opener}$'s key. If either of the challenge interactions does not finish successfully (i.e. if either $\Sigma_0 = \perp$ or $\Sigma_1 = \perp$), the adversary is not informed about the other signature. It is sufficient to run the challenge phase only once because the adversary has access to all the keys and the messages and hence it can replicate the signing multiple times by itself.

As was the case with the anonymity definition, one can consider different variants depending on whether or not the adversary is given access to the $\mathsf{Open}$ oracle with Full Blindness (FB) being the strongest among those variants. Refer to the full version [13] for more discussion about the different variants.

In FB the adversary can query the $\mathsf{Open}$ oracle at any stage on any signature except the two challenge signatures. If security is w.r.t. weak unforgeability then the $\mathsf{Open}$ oracle returns a special symbol if the signature is on either challenge message. On the other hand, if security is w.r.t. strong unforgeability, it returns the special symbol if queried on either challenge message-signature pair.

Formally, a dynamic $\mathsf{GBS}$ scheme is blind if for all $\lambda \in \mathcal{N}$, all PPT adversaries $\mathcal{A}$ have a negligible advantage $\mathsf{Adv}_{\mathsf{GBS},\mathcal{A}}^{Blind}(\lambda)$ where the advantage is defined as $\mathsf{Adv}_{\mathsf{GBS},\mathcal{A}}^{Blind}(\lambda) := \left|\Pr[\mathsf{Exp}_{\mathsf{GBS},\mathcal{A}}^{Blind-0}(\lambda) = 1] - \Pr[\mathsf{Exp}_{\mathsf{GBS},\mathcal{A}}^{Blind-1}(\lambda) = 1]\right|$.

## 4 Building Blocks

In this section, we present the building blocks we use in our constructions.

### 4.1 Groth-Sahai (GS) Proofs

Groth-Sahai (GS) proofs [16] are non-interactive proofs in the Common Reference String (CRS) model. We use the SXDH-based instantiation of the proofs (which, as noted by [14], are the most efficient instantiation) and that prove knowledge of a satisfying assignment for a *Pairing-Product Equation* (PPE):

$$\mathcal{E} := \prod_{j=1}^{n} \hat{e}(A_j, \underline{Y_j}) \prod_{i=1}^{m} \hat{e}(\underline{X_i}, B_i) \prod_{i=1}^{m}\prod_{j=1}^{n} \hat{e}(\underline{X_i}, \underline{Y_j})^{\gamma_{i,j}} \;=\; t_T, \tag{1}$$

where $X_1, \ldots, X_m \in \mathbb{G}_1$ and $Y_1, \ldots, Y_n \in \mathbb{G}_2$ are the secret variables (hence underlined) and $A_i \in \mathbb{G}_1$, $B_i \in \mathbb{G}_2$, $\gamma_{i,j} \in \mathbb{Z}_p$ and $t_T \in \mathbb{G}_T$ are public constants.

The proof system is defined by the algorithms

$$(\mathsf{GSSetup}, \mathsf{GSProve}, \mathsf{GSVerify}, \mathsf{GSExtract}, \mathsf{GSSimSetup}, \mathsf{GSSimProve}).$$

GSSetup takes as input the description of bilinear groups $\mathcal{P}$ and outputs a *binding* reference string crs and a trapdoor key xk which allows for witness extraction. GSProve takes as input a CRS, a witness (which we underline to distinguish it from public constants) and a set of equations, and outputs a proof $\Psi$ for the satisfiability of the equations. GSVerify takes as input a CRS, a proof $\Psi$ and a set of equations, and outputs 1 if the proof is valid or 0 otherwise. In the rest of the paper we will omit the equations from the input to the GSVerify algorithm. GSExtract takes as input a binding CRS, a valid proof $\Psi$ and the extraction key xk, and outputs the witness used in the proof. GSSimSetup takes as input the description of bilinear groups $\mathcal{P}$ and outputs a *hiding* reference string, $\mathsf{crs}_{\mathsf{sim}}$, and a trapdoor key tr which allows for proof simulation. GSSimProve takes as input a hiding CRS, $\mathsf{crs}_{\mathsf{sim}}$, the simulation key tr and a set of equations, and outputs a simulated proof $\Psi_{\mathsf{sim}}$.

The distributions of strings crs and $\mathsf{crs}_{\mathsf{sim}}$ are computationally indistinguishable and simulated proofs are indistinguishable from proofs output by GSProve. The proof system has prefect completeness, perfect soundness, composable witness-indistinguishability or composable zero-knowledge. We refer to [16] for the formal definitions.

As formalized by [2], GS proofs can be re-randomized by re-randomizing the associated GS commitments and updating the proofs accordingly so that we obtain fresh proofs that are unlinkable to the original ones. This requires knowledge of neither the witness nor the associated randomness used in the original GS commitments. We define an algorithm GSRandomize which takes as input a CRS and a proof $\Psi$ and outputs a proof $\Psi'$ which is a randomized version of the proof $\Psi$.

### 4.2 Blind Signatures

Blind signatures [8] allow a user to obtain signatures on messages hidden from the signer. Their security [17,23] consists of two requirements: blindness and unforgeability. Blindness requires that an adversarial signer cannot learn the message being signed and that he is not able to match a signature to its signing session. On the other hand, unforgeability deals with an adversarial user whose goal is to obtain $l + 1$ distinct message-signature pairs given only $l$ interactions with the honest signer.

In [1], the authors presented a blind signature scheme whose unforgeability reduces to the AWFCDH and q-ADHSDH assumptions (cf. Section 2). The scheme uses GS proofs and is akin to the idea used in Fischlin's generic construction [11]. The signer signs a commitment to the message. However, unlike the generic construction, the user transforms the signature on the commitment to the message into a signature on the message itself instead of proving that he knows a signature on a commitment to the message. In addition, the user proves knowledge of the message and the randomness used in the commitment when requesting a signature. The message space of the scheme is $\mathcal{M} := \{(G_1^m, G_2^m) | m \in \mathbb{Z}_p\}$.

Exploiting some properties of this blind signature scheme, we will show later how this scheme can be used to hide the identity of the signer while signing hidden messages. We refer to [1,13] for details of the scheme.

### 4.3 A New Structure-Preserving Signature Scheme

We present here a new structure-preserving signature scheme [1], i.e. whose messages, signatures and verification keys are group elements, and whose verification is a conjunction of PPE equations. The scheme which we call $\mathsf{NCL}$ is a variant of the CL signature scheme [5]. It is unforgeable under the DH-LRSW assumption (cf. Section 2).

The scheme is given by the following triple of algorithms. Given the description of bilinear groups $\mathcal{P}$ output by $\mathsf{BGrpSetup}(1^\lambda)$.

- $\mathsf{NCLKeyGen}(\mathcal{P})$: Select $x, y \leftarrow \mathbb{Z}_p$, set $\mathsf{sk}_{\mathsf{NCL}} := (x, y) \in \mathbb{Z}_p^2$ and $\mathsf{pk}_{\mathsf{NCL}} := (X, Y) := (G_2^x, G_2^y) \in \mathbb{G}_2^2$.
- $\mathsf{NCLSign}(\mathsf{sk}_{\mathsf{NCL}}, (M_1, M_2))$: To sign a message $(M_1, M_2) \in \mathbb{G}_1 \times \mathbb{G}_2$, return $\perp$ if $\hat{e}(M_1, G_2) \neq \hat{e}(G_1, M_2)$. Otherwise, select $a \leftarrow \mathbb{Z}_p^*$, and set $A := G_1^a$, $B := A^y$, $C := M_1^{ay}$, $D := A^x \cdot M_1^{axy}$. Output $\sigma := (A, B, C, D) \in \mathbb{G}_1^4$.
- $\mathsf{NCLVerify}(\mathsf{pk}_{\mathsf{NCL}}, (M_1, M_2), \sigma)$: Output 1 iff $A \neq 1_{\mathbb{G}_1} \;\wedge\; \hat{e}(A, Y) = \hat{e}(B, G_2)$ $\wedge\; \hat{e}(B, M_2) = \hat{e}(C, G_2) \;\wedge\; \hat{e}(D, G_2) = \hat{e}(A \cdot C, X) \;\wedge\; \hat{e}(G_1, M_2) = \hat{e}(M_1, G_2)$.

The proof for the following theorem is straightforward.

**Theorem 1.** *Assuming the DH-LRSW assumption holds, the* $\mathsf{NCL}$ *signature scheme is existentially unforgeable against adaptive chosen-message attacks.*

## 5 Our Construction

In this section we present our constructions.

### 5.1 Techniques Used

We identify and utilize a number of observations and desired properties the building blocks we use have. We summarize those as follows:

1. Groth-Sahai proofs are independent of any public terms (i.e. public monomials) in the equations being proven.[2] Thus, given a witness-indistinguishable GS proof, $\varPsi$, (i.e. one for an equation with a non-trivial public right-hand side $\tau$), we can later (without knowing the original witnesses in the proof) transform $\varPsi$ into a related witness-indistinguishable/zero-knowledge proof $\varPsi'$ by splitting the public right-hand side $\tau$ into a set of witnesses and adding them to the witness of the original proof and then updating the proof accordingly. One can construct GS NIZK proofs for pairing-product equations (Equation 1) if either $t_T = 1$ or if one can factor $t_T$ by finding $P_i, Q_i$ such that $t_T = \prod_{i=1}^{n} \hat{e}(P_i, Q_i)$. It is the latter case that applies to our construction as the user knows how to open the commitment to the message and therefore can factor $t_T$. This will become clearer when we present our construction.
2. We prove in the full version [13] the following lemma (which is analogous to Lemma 4 [12] for PPE equations) for multi-scalar multiplication equations.

   **Lemma 1.** *Let $((\mathcal{C}_z, \mathcal{C}_A), \varPsi)$ be a GS proof for the equation $\mathcal{E} := \underline{A}^{\underline{z}} = Z$, where $\mathcal{C}_z, \mathcal{C}_A$ are the GS commitments to $z \in \mathbb{Z}_p$ and $A \in \mathbb{G}$ respectively and $Z \in \mathbb{G}$ then $(({\mathcal{C}_z}^{z'}, \mathcal{C}_A), \varPsi^{z'})$ is a GS proof for the equation $\mathcal{E}' := \underline{A}^{\underline{z' \cdot z}} = Z^{z'}$.*

   The same argument holds for quadratic equations over $\mathbb{Z}_p$ where exponentiation of the public right-hand side is replaced by multiplication.

### 5.2 Overview of the Construction

**The Broad Idea**. In Fischlin's generic construction for blind signatures [11], the user sends a commitment to the message to the signer, who in turn returns a signature on the commitment. The final blind signature is a NIZK proof of knowledge of the signature and the commitment s.t. the signature is valid on the commitment and the commitment is to the message in question.

Now assume in the above framework we want to hide the signer's identity from the user. So instead of sending the signature on the user's commitment (to the message) in the clear, the signer sends a proof of knowledge of such a signature and his verification key to the user. If the proofs used are re-randomizable and transformable in the sense of Observation 1 from Section 5.1, where in the equations used in the signer's proofs, the terms involving the commitment are public, then the user can re-randomize the proofs and transform them to hide the commitment to the message and compute the final blind signature. Unforgeability follows from that of Fischlin's framework, blindness also follows from that of the framework plus the re-randomizability of the proofs, and the anonymity of the signer is ensured by the hiding properties of the proofs. Thus, we obtain a signer-anonymous blind signature scheme.

[2] This observation was also independently noted by [12].

| $\underline{\mathsf{Join}(\mathsf{gpk}, i, \mathbf{ssk}[i])}$ | | $\underline{\mathsf{Issue}(\mathsf{ik}, i, \mathbf{spk}[i])}$ |
|---|---|---|
| $s \leftarrow \mathbb{Z}_p$, $\mathsf{sk}_i := s, \mathsf{pk}_i := ((S_1, S_2) := (G_1^s, G_2^s))$ | | |
| $\mathsf{sig}_i \leftarrow \mathsf{CERTSign}(\mathbf{ssk}[i], \mathsf{pk}_i)$ | $\xrightarrow{\mathsf{sig}_i, \mathsf{pk}_i}$ | Parse $\mathsf{pk}_i$ as $(S_1, S_2)$ |
| | | If $\mathsf{CERTVerify}(\mathbf{spk}[i], \mathsf{pk}_i, \mathsf{sig}_i) = 0$ |
| | | OR $\hat{e}(S_1, G_2) \neq \hat{e}(G_1, S_2)$ |
| | | OR $\mathsf{pk}_i = \mathsf{pk}_j$ for any $j$ Then Abort |
| Abort if $\mathsf{CERTVerify}(\mathsf{pk}_{\mathsf{CERT}}, \mathsf{pk}_i, \mathsf{cert}_i) = 0$ | $\xleftarrow{\mathsf{cert}_i}$ | $\mathsf{cert}_i \leftarrow \mathsf{CERTSign}(\mathsf{ik}, \mathsf{pk}_i)$ |
| $\mathbf{gsk}[i] := (\mathsf{sk}_i, \mathsf{pk}_i, \mathsf{cert}_i)$ | | $\mathbf{reg}[i] := (\mathsf{pk}_i, \mathsf{sig}_i)$ |

**Fig. 3.** The Join/Issue protocol for our construction

What remains is to extend the signer-anonymous blind signature to the group setting by requiring a compatible signature scheme to certify signer's keys when they join the group, and also providing a tracing mechanism for the Opener.

**The Construction**. We base our signing protocol on the blind signature scheme from [1], which uses GS proofs and has the required properties needed for our construction. The same methodology can similarly be applied to other GS based instantiations of Fischlin's construction satisfying the properties required for our paradigm, e.g. the instantiation by Abe et al. in [1].

To issue membership certificates (i.e. credentials) for new group members, we can use any structure-preserving signature scheme [1]. We will use the prefix CERT for this scheme. The Issuer gets the secret signing key $\mathsf{sk}_{\mathsf{CERT}}$ for the CERT signature scheme. Each potential group member $\mathsf{Signer}_i$ would have created his pair of personal secret/public keys $(\mathbf{ssk}[i], \mathbf{spk}[i])$ prior to joining the group. When requesting to join the group, $\mathsf{Signer}_i$ generates a pair of secret signing/public verification keys $(\mathsf{sk}_i, \mathsf{pk}_i)$ for the blind signature scheme. To stop a corrupt Issuer from framing group members, we ask that the group member signs his verification key $\mathsf{pk}_i$ with his personal secret key $\mathbf{ssk}[i]$, the resulting signature $\mathsf{sig}_i$ will be used as a proof when verifying the Opener's claim. We will use the CERT scheme for this purpose as well. Thus, WLOG we assume that the key pair $(\mathbf{ssk}[i], \mathbf{spk}[i])$ is a valid pair for the CERT scheme.

The Issuer first verifies the signature $\mathsf{sig}_i$ and if it is valid, he issues a signature on $\mathsf{pk}_i$ using the CERT signature scheme and his secret issuing key ik. After successfully joining the group, $\mathsf{Signer}_i$'s secret group signing key $\mathbf{gsk}[i]$ is $(\mathsf{sk}_i, \mathsf{pk}_i, \mathsf{cert}_i)$, whereas his registration information is set to $\mathbf{reg}[i] := (\mathsf{pk}_i, \mathsf{sig}_i)$.

The group public key gpk contains the public key of the CERT signature scheme, the public values used in the blind signature scheme and two GS reference strings $\mathsf{crs}_1$ and $\mathsf{crs}_2$, which are used in constructing GS proofs used in the first and second rounds of the signing protocol, respectively. We use separate GS reference strings because we believe this provides extra functionality such as preventing the opener from revoking the anonymity of the message in the signing phase or to allow for having a different opener for revoking anonymity of the message if needed. To open signatures, the Opener is given the extraction key for the GS proof system.

The signing protocol $\langle \mathsf{Obtain}, \mathsf{Sign} \rangle$ is a two-move protocol between a user with a message $(M_1, M_2) \in \mathbb{G}_1 \times \mathbb{G}_2$ and an anonymous group member $\mathsf{Signer}_i$

```
GKg(1^λ)
  – $\mathcal{P} \leftarrow \mathsf{BGrpSetup}(1^\lambda)$; $F, K, T \leftarrow \mathbb{G}_1$
  – $(\mathsf{crs}_i, \mathsf{xk}_i) \leftarrow \mathsf{GSSetup}(\mathcal{P})$ for i=1,2
  – $(\mathsf{sk}_{\mathsf{CERT}}, \mathsf{pk}_{\mathsf{CERT}}) \leftarrow \mathsf{CERTKeyGen}(\mathcal{P})$
  – $\mathsf{gpk} := (\mathcal{P}, \mathsf{crs}_1, \mathsf{crs}_2, F, K, T, \mathsf{pk}_{\mathsf{CERT}})$
  – $\mathsf{ik} := \mathsf{sk}_{\mathsf{CERT}}$; $\mathsf{ok} := \mathsf{xk}_2$
  – Return $(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok})$

SKg($\mathcal{P}$)
  – $(\mathbf{ssk}[i], \mathbf{spk}[i]) \leftarrow \mathsf{CERTKeyGen}(\mathcal{P})$
  – Return $(\mathbf{ssk}[i], \mathbf{spk}[i])$

GVf(gpk, $(M_1, M_2), \Sigma$)
  – Parse $\Sigma$ as $\boldsymbol{\Omega}$
  – Return $\mathsf{GSVerify}(\mathsf{crs}_2, \boldsymbol{\Omega}) = 1$

Open(gpk, ok, reg, $(M_1, M_2), \Sigma$)
  – Parse gpk as $(\mathcal{P}, \mathsf{crs}_1, \mathsf{crs}_2, F, K, T, \mathsf{pk}_{\mathsf{CERT}})$
  – Parse $\Sigma$ as $\boldsymbol{\Omega}$ and ok as $\mathsf{xk}_2$
  – $(\sigma, \mathsf{cert}, \mathsf{pk}) \leftarrow \mathsf{GSExtract}(\mathsf{crs}_2, \mathsf{xk}_2, \boldsymbol{\Omega})$
  – If $\exists i$ s.t. $\mathbf{reg}[i].\mathsf{pk} = \mathsf{pk}$ Then
        Return $(i, (\sigma, \mathsf{cert}, \mathsf{pk}, \mathbf{reg}[i].\mathsf{sig}))$
    Else Return $(0, (\sigma, \mathsf{cert}, \mathsf{pk}, \epsilon))$

Judge(gpk, $i$, $\mathbf{spk}[i]$, $(M_1, M_2), \Sigma, \tau$)
  – Parse gpk as $(\mathcal{P}, \mathsf{crs}_1, \mathsf{crs}_2, F, K, T, \mathsf{pk}_{\mathsf{CERT}})$
  – Parse $\tau$ as $(i, \sigma, \mathsf{cert}, \mathsf{pk}, \mathsf{sig})$
  – If $i > 0$ and $\mathsf{Verify}(\mathsf{pk}, (M_1, M_2), \sigma) = 1$
        and $\mathsf{CERTVerify}(\mathbf{spk}[i], \mathsf{pk}, \mathsf{sig}) = 1$
        and $\mathsf{CERTVerify}(\mathsf{pk}_{\mathsf{CERT}}, \mathsf{pk}, \mathsf{cert}) = 1$
    Then Return 1 Else Return 0

The signing protocol $\langle \mathsf{Obtain}(\mathsf{gpk}, (M_1, M_2)), \mathsf{Sign}(\mathbf{gsk}[i]) \rangle$

Obtain → Sign:
  – Choose $q \leftarrow \mathbb{Z}_p$ and set $Q_1 := G_1^q$, $Q_2 := G_2^q$ and $Co := M_1 \cdot T^q$.
  – $\boldsymbol{\Psi} \leftarrow \mathsf{GSProve}\big(\mathsf{crs}_1, \{M_1, M_2, Q_1, Q_2\}, \{\hat{e}(\underline{M_1}, G_2) = \hat{e}(G_1, \underline{M_2})$
        $\wedge\ \hat{e}(\underline{Q_1}, G_2) = \hat{e}(G_1, \underline{Q_2})\ \wedge\ \hat{e}(\underline{M_1}, G_2)\hat{e}(T, \underline{Q_2}) = \hat{e}(Co, G_2)\}\big)$.
  – Send $(Co, \boldsymbol{\Psi})$ to Sign.

Sign → Obtain:
  – If $\mathsf{GSVerify}(\mathsf{crs}_1, \boldsymbol{\Psi}) \neq 1$ Then Abort().
  – Choose $r, c \leftarrow \mathbb{Z}_p$ and set $H := (K \cdot T^r \cdot Co)^{\frac{1}{s+c}}$, $R_1' := G_1^r$, $R_2' := G_2^r$,
        $C_1 := F^c$ and $C_2 := G_2^c$.
  – Set $\sigma := (H, C_1, C_2, R_1', R_2')$ and parse $\mathsf{pk}_i$ as $(S_1, S_2)$.
  – Compute $\boldsymbol{\Omega'} \leftarrow \mathsf{GSProve}(\mathsf{crs}_2, \{\mathsf{cert}, S_1, S_2, H, C_1, C_2\},$
        $\{\mathsf{CERTVerify}(\mathsf{pk}_{\mathsf{CERT}}, (\underline{S_1}, \underline{S_2}), \underline{\mathsf{cert}}) = 1$
        $\wedge\ \hat{e}(\underline{S_1}, G_2) = \hat{e}(G_1, \underline{S_2})\ \wedge\ \hat{e}(\underline{C_1}, G_2) = \hat{e}(F, \underline{C_2})$
        $\wedge\ \hat{e}(\underline{H}, \underline{S_2} \cdot \underline{C_2}) = \hat{e}(K \cdot Co, G_2)\hat{e}(T, R_2')\})$.
  – Send $\big(R_1', R_2', \boldsymbol{\Omega'}\big)$ to Obtain.

Obtain:
  If $\mathsf{GSVerify}(\mathsf{crs}_2, \boldsymbol{\Omega'}) \neq 1$ or $\hat{e}(R_1', G_2) \neq \hat{e}(G_1, R_2')$ Then Abort().
  – $\boldsymbol{\Omega'} \leftarrow \mathsf{GSRandomize}(\mathsf{crs}_2, \boldsymbol{\Omega'})$ and set $R_i := R_i' \cdot Q_i$ for $i = 1, 2$
  – Modify² $\boldsymbol{\Omega'}$ to $\boldsymbol{\Omega} \leftarrow \mathsf{GSProve}(\mathsf{crs}_2, \{\mathsf{cert}, S_1, S_2, H, C_1, C_2, R_1, R_2\},$
        $\{\mathsf{CERTVerify}(\mathsf{pk}_{\mathsf{CERT}}, (\underline{S_1}, \underline{S_2}), \underline{\mathsf{cert}}) = 1$
        $\wedge\ \hat{e}(\underline{S_1}, G_2) = \hat{e}(G_1, \underline{S_2})\ \wedge\ \hat{e}(\underline{C_1}, G_2) = \hat{e}(F, \underline{C_2})$
        $\wedge\ \hat{e}(\underline{H}, \underline{S_2} \cdot \underline{C_2})\hat{e}(T^{-1}, \underline{R_2}) = \hat{e}(K \cdot M_1, G_2)$
        $\wedge\ \hat{e}(\underline{R_1}, G_2) = \hat{e}(G_1, \underline{R_2})\})$.
  – Output $\Sigma := \boldsymbol{\Omega}$.
```

³ The transformation is done without knowledge of the original witness of the proof.

**Fig. 4.** The Construction

with a secret group signing key $\mathbf{gsk}[i]$. The user commits to the message using Pedersen commitment $Co := M_1 \cdot T^q$ for some random $q \leftarrow \mathbb{Z}_p$ and computes $Q_i := G_i^q$ for i=1,2. He then sends the commitment $Co$ along with GS proofs of knowledge $\boldsymbol{\Psi}$ to prove that: the commitment $Co$ is indeed to the message $M_1$ and that the message and the randomness pairs are well-formed.

The signer first verifies the proofs and if they are valid, produces a signature $\sigma$ on the commitment $Co$ as in the blind signature scheme in [1], where $\sigma := (H, C_1, C_2, R_1', R_2')$. However, instead of sending the signature in the clear as in in [1], the signer sends a GS proof of knowledge $\boldsymbol{\Omega'}$ of his public verification key $\mathsf{pk}_i$, his membership certificate $\mathsf{cert}_i$ and the signature $\sigma$ such that the signature verifies w.r.t. his key, the certificate is valid on his key and the key is well-formed. Note that $R_1'$ and $R_2'$ are independent of the signing key and hence we can send them in the clear to the user. The signer's response is $(R_1', R_2', \boldsymbol{\Omega'})$.

The user first verifies the GS proofs $\boldsymbol{\Omega'}$ and that the pair $(R_1', R_2')$ is well-formed. If they are valid, the user re-randomizes those proofs using the algorithm $\mathsf{GSRandomize}$. The new proofs are unlinkable to the original ones. Note that the last equation proven in $\boldsymbol{\Omega'}$ is $\hat{e}(\underline{H}, \underline{S_2} \cdot \underline{C_2}) = \hat{e}(K \cdot Co, G_2)\hat{e}(T, R_2')$, where $Co$ and $R_2'$ are public at this stage and hence not part of the witness. The user now updates the components $R_1'$ and $R_2'$ to include the randomness used in the commitment (by computing $R_i := R_i' \cdot Q_i$ for $i = 1, 2$) and then (by applying Observation (1) from Section 5.1) decomposes the commitment and transforms the proof for the last equation in $\boldsymbol{\Omega'}$ into a proof based on the message $M_1$ instead of the commitment $Co$ by adding the value $R_2$ to the witness. The transformation is done without knowledge of $H, C_2$ or $S_2$. The proof is now for the equation $\hat{e}(\underline{H}, \underline{S_2} \cdot \underline{C_2})\hat{e}(T^{-1}, \underline{R_2}) = \hat{e}(K \cdot M_1, G_2)$, where $H, S_2, C_2$ and $R_2$ are parts of the witness. In addition, the user adds to $\boldsymbol{\Omega}$ a new GS proof to prove that $R_1$ and $R_2$ hide the same exponent.

The final signature $\Sigma$ is a set of GS proofs of knowledge $\boldsymbol{\Omega}$ to prove that: the group member has a valid credential from the $\mathsf{Issuer}$, his public key is well-formed and the blind signature verifies w.r.t. his key.

To open a signature, the $\mathsf{Opener}$ uses his secret extraction key to extract the verification key $\mathsf{pk}$, the signature $\sigma$ and the membership certificate $\mathsf{cert}$ from the proofs. Besides those, the $\mathsf{Opener}$ returns the index $i$ of the group member and the signature $\mathsf{sig}$ in support of his claim. The $\mathsf{Judge}$ algorithm can verify the correctness of the $\mathsf{Opener}$'s decision by verifying those components and checking that the group member has indeed signed the key $\mathsf{pk}$ with his secret key $\mathbf{ssk}[i]$.

The construction is illustrated in Figure 4, whereas the joining protocol is in Figure 3.

We provide a proof for the following Theorem in the full version [13].

**Theorem 2.** *The construction in Figures 3 and 4 is a secure group blind signature scheme (with CPA-anonymity and CPA-blindness) if the* $\mathsf{CERT}$ *scheme is unforgeable, the GS proof system is sound, hiding (i.e. witness-indistinguishable/zero-knowledge) and re-randomizable, and the blind signature scheme is secure.*

Next we present two example instantiations of the construction.

**Instantiation I**. Here we instantiate the $\mathsf{CERT}$ scheme using the asymmetric automorphic signature scheme ($\mathsf{AFPV}$) from [1]. Thus, we get an instatiation whose security is solely based on non-interactive complexity assumptions. If GS proofs are instantiated in the SXDH setting, the final signature size is $42 \cdot |\mathbb{G}_1| + 38 \cdot |\mathbb{G}_2|$. Refer to the full version [13] for details.

**Instantiation II**. To get better efficiency, we instantiate the $\mathsf{CERT}$ scheme using our new $\mathsf{NCL}$ signature scheme (cf. Section 4.3). Since in our construction, the final group blind signature hides the components of the certificate and hence one cannot directly verify that the certificate is non-trivial, in the instantiation the signer additionally needs to prove that the certificate is non-trivial (i.e. that $A \neq 1_{\mathbb{G}_1}$). Otherwise, the adversary can create untraceable signatures by faking trivial certificates. We suggest two re-randomizable proofs for this statement in the full version [13]. Despite the need for this extra proof, this instantiation is more efficient than instantiation I and yields signatures of size $38 \cdot |\mathbb{G}_1| + 36 \cdot |\mathbb{G}_2|$. Refer to the full version [13] for details.

## 6 Achieving Full Anonymity

Groth-Sahai proofs do not provide simulation and extractability simultaneously and hence when simulating, the Opener can no longer answer Open queries.

One way to get full anonymity is to combine GS proofs with an IND-CCA secure encryption scheme to encrypt the witness used in the proofs and add an extra GS proof to prove that the encrypted information is the same as that used as a witness in the other proofs. We give the Opener the decryption key for the encryption scheme which allows him to recover the information from the ciphertext if it cannot be extracted from GS proofs.

It appears that the encryption scheme we require has to be re-randomizable but yet the IND-CCA security contradicts re-randomizability of ciphertexts which is in some sense a form of malleability. However, there are a number of schemes with properties that seem to suffice for this purpose, e.g. [15].

**Acknowledgements.** This work was supported by ERC Advanced Grant ERC-2010-AdG-267188-CRIPTO and EPSRC via grant EP/H043454/1. We thank Georg Fuchsbauer, Nigel Smart and Bogdan Warinschi for many useful discussions. We also thank anonymous reviewers for valuable comments.

## References

1. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-preserving signatures and commitments to group elements. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 209–236. Springer, Heidelberg (2010)
2. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable Proofs and Delegatable Anonymous Credentials. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 108–125. Springer, Heidelberg (2009)
3. Bellare, M., Rogaway, P.: Random oracles are practical: A Paradigm for Designing Efficient Protocols. In: ACM-CCS 1993, pp. 62–73. ACM (1993)
4. Bellare, M., Shi, H., Zhang, C.: Foundations of group signatures: The case of dynamic groups. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 136–153. Springer, Heidelberg (2005)
5. Camenisch, J., Lysyanskaya, A.: Signature schemes and anonymous credentials from bilinear maps. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 56–72. Springer, Heidelberg (2004)
6. Canetti, R., Krawczyk, H., Nielsen, J.B.: Relaxing Chosen-Ciphertext Security. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 565–582. Springer, Heidelberg (2003)

7. Camenisch, J., Stadler, M.: Efficient group signature schemes for large groups. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 410–424. Springer, Heidelberg (1997)
8. Chaum, D.: Blind signatures for untraceable payments. In: CRYPTO 1982, pp. 199–203. Plenum Press (1983)
9. Chaum, D., van Heyst, E.: Group signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991)
10. 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)
11. Fischlin, M.: Round-optimal composable blind signatures in the common reference string model. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 60–77. Springer, Heidelberg (2006)
12. Fuchsbauer, G.: Commuting Signatures and Verifiable Encryption and an Application to Non-Interactively Delegatable Credentials. In Cryptology ePrint Archive, Report 2010/233 (2010), http://eprint.iacr.org/2010/233.pdf
13. Ghadafi, E.: Formalizing group blind signatures and practical constructions without random oracles. In Cryptology ePrint Archive, Report 2011/402 (2011), http://eprint.iacr.org/2011/402.pdf
14. Ghadafi, E., Smart, N.P., Warinschi, B.: Groth-Sahai proofs revisited. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 177–192. Springer, Heidelberg (2010)
15. Groth, J.: Rerandomizable and Replayable Adaptive Chosen Ciphertext Attack Secure Cryptosystems. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 152–170. Springer, Heidelberg (2004)
16. 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)
17. Juels, A., Luby, M., Ostrovsky, R.: Security of blind digital signatures. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 150–164. Springer, Heidelberg (1997)
18. Lysyanskaya, A., Rivest, R.L., Sahai, A., Wolf, S.: Pseudonym systems (Extended abstract). In: Heys, H.M., Adams, C.M. (eds.) SAC 1999. LNCS, vol. 1758, pp. 184–199. Springer, Heidelberg (2000)
19. Lysyanskaya, A., Ramzan, Z.: Group blind digital signatures: A scalable solution to electronic cash. In: Hirschfeld, R. (ed.) FC 1998. LNCS, vol. 1465, pp. 184–197. Springer, Heidelberg (1998)
20. Naor, M.: On cryptographic assumptions and challenges. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 96–109. Springer, Heidelberg (2003)
21. Nguyen, K.Q., Mu, Y., Varadharajan, V.: Divertible Zero-Knowledge Proof of Polynomial Relations and Blind Group Signature. In: Pieprzyk, J.P., Safavi-Naini, R., Seberry, J. (eds.) ACISP 1999. LNCS, vol. 1587, pp. 117–128. Springer, Heidelberg (1999)
22. Okamoto, T., Ohta, K.: Divertible Zero Knowledge Interactive Proofs and Commutative Random Self-Reducibility. In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 134–149. Springer, Heidelberg (1990)
23. Pointcheval, D., Stern, J.: Security arguments for digital signatures and blind signatures. Journal of Cryptology 13(3), 361–396 (2000)
24. 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)
25. Wu, Q., Zhang, F., Susilo, W., Mu, Y.: An Efficient Static Blind Ring Signature Scheme. In: Won, D.H., Kim, S. (eds.) ICISC 2005. LNCS, vol. 3935, pp. 410–423. Springer, Heidelberg (2006)

# Analysis of the Non-perfect Table Fuzzy Rainbow Tradeoff[*]

Byoung-Il Kim and Jin Hong[**]

Department of Mathematical Sciences and ISaC,
Seoul National University, Seoul 151-747, Korea
{samaria2,jinhong}@snu.ac.kr

**Abstract.** Time memory tradeoff algorithms are tools for inverting one-way functions, and they are often used to recover passwords from unsalted password hashes. There are many publicly known tradeoff algorithms, and the rainbow tradeoff is widely believed to be the best algorithm. This work provides an accurate complexity analysis of the non-perfect table version of the fuzzy rainbow tradeoff algorithm, which has not yet received much attention. It is shown that, when the pre-computation cost and the online efficiency are both taken into consideration, the non-perfect fuzzy rainbow tradeoff is preferable to the original rainbow tradeoff in many situations.

**Keywords:** time memory tradeoff, rainbow table, fuzzy rainbow, distinguished point.

## 1 Introduction

Cryptanalytic time memory tradeoff algorithms are tools for quickly inverting one-way functions with the help of pre-computed data. They are used by law enforcement agencies and hackers to recover passwords from unsalted password hashes, and the multi-target variants of the algorithms have been used [7, 10, 11, 15] to show that the GSM mobile phones are insecure.

There are a multitude of publicly known tradeoff algorithms. However, a quick search for password recovery tools on the Web reveals [1–4] that the rainbow tradeoff [23] is by far the most popular algorithm, and this seems to indicate that the rainbow tradeoff is widely believed among implementers to be the best tradeoff algorithm.

Although it is difficult to clearly specify what is meant by one tradeoff algorithm to be better than another, the recent work [18] has given a plausible method for comparing the performances of different tradeoff algorithms. Unlike previous comparison attempts that had focused mainly on the optimal online

[*] This work was supported by the Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2012R1A1B4003379).

[**] JH is the corresponding author.

phase behavior of the algorithms, the recently suggested method takes both the pre-computation cost and the online efficiency into account. Hence, the new method reflects our intuition concerning the practicality, usefulness, or value of the algorithms more closely. This approach was used in [18, 20] to show that the perfect and non-perfect rainbow tradeoffs perform better than the classical Hellman [16], perfect distinguished point [12–14], and non-perfect distinguished point tradeoff algorithms, under typical environments, thus supporting the aforementioned beliefs.

In this work, we analyze the execution behavior of the non-perfect table fuzzy rainbow tradeoff [6, 8], which has not yet received much attention, and compare its performance against the original perfect and non-perfect rainbow tradeoffs. It is confirmed that the degree of online phase efficiency made possible by the original rainbow tradeoff through appropriate parameter choices is higher than that reachable with the fuzzy rainbow tradeoff. However, we find that, for online efficiency levels that can be reached by both tradeoff algorithms, the fuzzy rainbow tradeoff requires less pre-computation effort than the original rainbow tradeoff. In other words, up to a certain point, for the same pre-computation investment, higher online efficiency is returned by the fuzzy rainbow tradeoff. Since the massive pre-computation requirement stands as a significant barrier to any large scale deployment of the tradeoff technique, the fuzzy rainbow tradeoff will often be preferable to the original rainbow tradeoff.

The comparison given in this work follows the framework set by [18], and our main contribution is in providing an accurate execution behavior analysis of the non-perfect table fuzzy rainbow tradeoff. However, even the comparison task does require us to overcome new difficulties arising from the extra parameter introduced by the fuzzy rainbow tradeoff.

The fuzzy rainbow tradeoff is a combination of the distinguished point tradeoff and the rainbow tradeoff, and both of these tradeoffs can be seen as special cases of the fuzzy rainbow tradeoff corresponding to extreme choices of parameters. Hence, one might expect the performance of the fuzzy rainbow tradeoff to come somewhere in between those of the two tradeoffs. The findings of this paper, which indicate otherwise, could be seen as unexpected.

Arguments supporting the efficiency of the fuzzy rainbow tradeoff were given in the original publications [6, 8]. However, they were based on the concept of hidden states, which totally disregarded pre-computation cost, and the computations made there were not tight enough to be accurate up to small constant factors. Since the performances of tradeoff algorithms often differ only by small constant factors, which nevertheless have heavy consequences in practice, it was not possible to provide an appropriate comparison of algorithms based on their claims.

There are only two works concerning the fuzzy rainbow tradeoff, other than [6, 8], that we are aware of. The multi-target version of the fuzzy rainbow tradeoff is described as having been used during the presentation [21, 22] of a fully implemented attack on GSM phones, but no theoretical analysis was given there. The work [25] provides an entry-level analysis of the same algorithm, but incorrect assumptions were made concerning the parameters. Neither work mentions that the algorithm being dealt with appeared previously in [6, 8].

The rest of this paper is organized as follows. In the next section, we review the fuzzy rainbow tradeoff algorithm. The execution behavior of the fuzzy rainbow tradeoff is analyzed in Section 3 with accuracy that does not ignore any small constant factors. The obtained results are used in Section 4 to present a fair comparison between the fuzzy rainbow tradeoff and the original rainbow tradeoffs. Concluding remarks are made in Section 5.

Much of the proofs in this article have been removed or condensed to adhere to the space requirement. Readers are referred to [19] for a presentation with arguments that are easier to follow and some extra material that are not mentioned here, such as experimental verification of our theoretical findings.

## 2 Preliminaries

The reader is assumed to be familiar with the basic tradeoff techniques and terminology. Throughout this paper, the one-way function $f$ is taken to act on a search space of size $\mathsf{N}$. The composition of the one-way function and the reduction function of $i$-th color will be written as $f_i$. The standard notation for the number of chains $m$ per table, the chain length $t$, and the number of tables $\ell$ will be used. When dealing with DPs (distinguished points), the distinguishing property will always be assumed to be of probability $\frac{1}{t}$, so that the expected length of a random chain is $t$. The collection of all chains associated with one pre-computation table is referred to as a pre-computation matrix.

The fuzzy rainbow tradeoff [6, 8] is a combination of the rainbow tradeoff and the DP tradeoff. Recall that each pre-computation matrix for the rainbow tradeoff contains $m$ pre-computation chains, each of length $t$, that take the form

$$\mathrm{SP} \xrightarrow{f_1} \circ \xrightarrow{f_2} \circ \cdots \circ \xrightarrow{f_t} \mathrm{EP}, \tag{1}$$

where SP and EP denote the starting and ending points, respectively. The color of the one-way function is changed at each iteration of the chain generation for a total of $t$ different colors, for each matrix.

Also recall that each DP matrix contains $m$ chains of variable lengths that take the form

$$\mathrm{SP} \xrightarrow{f_i} \circ \xrightarrow{f_i} \circ \cdots \circ \xrightarrow{f_i} \mathrm{DP} = \mathrm{EP}. \tag{2}$$

That is, each chain is generated using a single color until the appearance of a DP and this point is taken to be the ending point. One-way function colors are changed only when the pre-computation starts on a new DP matrix.

In the case of the fuzzy rainbow tradeoff, a distinguishing property of probability $\frac{1}{t}$ and a positive integer $s$ are fixed, and chains of the form

$$\begin{aligned} \mathrm{SP} \xrightarrow{f_1} \circ \cdots \circ \xrightarrow{f_1} \mathrm{DP} \xrightarrow{f_2} \circ \cdots \circ \xrightarrow{f_2} \mathrm{DP} \xrightarrow{f_3} \circ \cdots \\ \cdots \xrightarrow{f_{s-1}} \mathrm{DP} \xrightarrow{f_s} \circ \cdots \circ \xrightarrow{f_s} \mathrm{DP} = \mathrm{EP} \end{aligned} \tag{3}$$

are used. The colored one-way function iterations are continued under a fixed color until a DP is reached, after which the iterations are continued with a different color. A total of $s$ colors are used for each pre-computation chain. As with other tradeoff algorithms, $m$ chains are generated for each of the $\ell$ pre-computation tables, and only the starting point and ending point pairs, sorted according to the ending points, are stored in the pre-computation tables. The DP ($s = 1$) and rainbow ($t = 1$) tradeoffs are degenerate forms of the fuzzy rainbow tradeoff.

The algorithm we have just described is the non-perfect table version of the fuzzy rainbow tradeoff. One can also consider the perfect tables version of this algorithm, obtained by retaining just one chain from every set of merging chains. However, only the non-perfect table version of the fuzzy rainbow tradeoff will be studied in this work. The perfect table version is likely to have better online efficiency, but will require larger pre-computation, and its study is deferred to a future work.

The fuzzy rainbow tradeoff analogue of the matrix stopping rules is $mt^2s \approx \mathsf{N}$. That is, we always assume that the fuzzy rainbow tradeoff parameters $m$, $t$, and $s$ are chosen in such a way that the matrix stopping constant $\mathtt{F}_{\mathrm{msc}} = \frac{mt^2s}{\mathsf{N}}$ is neither too large nor very close to zero. We shall express such a situation simply as $\mathtt{F}_{\mathrm{msc}} = \Theta(1)$. The appropriateness of this matrix stopping rule can be found in [6]. The theoretical arguments of this paper will be easier to comprehend when $s$ is assumed to be much smaller than $m$ or $t$, even though no such assumption appears in [6, 8]. It will later become clear that the $s$ values of interest will mostly be in the range $10 \sim 300$.

To complete the description of the fuzzy rainbow tradeoff algorithm, the order of online chain creation needs to be clarified. In short, the multiple tables are processed in parallel, in a manner analogous to the online phase of the rainbow tradeoff. At the initial pass, the online chains that start from the final $s$-th colors are generated for all the pre-computation tables. If the treatment of all alarms generated by these online chains did not succeed in recovering the correct answer, one moves to the second pass of the online phase. At the second pass, the online chains that start from the $(s-1)$-th colors and extend into the $s$-th colors are generated for all the pre-computation tables. This process continues until either the correct answer is found or all the $s$ passes are complete. As with the original rainbow tradeoff, generation of the shorter online chains before the longer ones reduces the expected time complexity.

The fuzzy rainbow tradeoff algorithm as introduced by [6, 8] was a time memory *data* tradeoff algorithm. The intension of [6, 8] was to create a variant of the rainbow tradeoff that has a multi-target tradeoff curve of the $TM^2D^2 \approx \mathsf{N}^2$ form, as with the multi-target versions of Hellman or DP tradeoffs, since the original rainbow tradeoff was shown to be inferior [9] in this respect. However, in this work, we shall restrict to the $D = 1$ case and treat the fuzzy rainbow tradeoff as a single target inversion algorithm. The multi-target version of the fuzzy rainbow tradeoff must be compared against the multi-target versions of the classical Hellman and DP tradeoffs, but nothing similar to [18, 20] has yet appeared for the multi-target versions. Furthermore, rudimentary arguments seem to indicate that the analyses done for the single target inversion algorithms will carry over to the multi-target algorithm setting with minimal changes.

A single fuzzy rainbow matrix may be understood as a concatenation of $s$ DP sub-matrices. Throughout this work, the $i$-th ($i = 1, \dots, s$) DP sub-matrix will be denoted by $\mathtt{DM}_i$. The only difference between a $\mathtt{DM}_i$ and a normal non-perfect DP matrix is that $\mathtt{DM}_i$ may contain duplicate starting points, that bring about fully identical chains.

Any implementation of a tradeoff algorithm that relies on DPs will place a chain length bound to detect chains falling into loops. In this work, we assume that a sufficiently large chain length bound is used. This is not exactly equivalent to sending the chain length bound to infinity during analysis, and a discussion of the precise meaning of this assumption may be found in [18, 20].

There will be many approximations made throughout this paper. Many of these will depend on the observation $(1 - \frac{1}{b})^a \approx e^{-\frac{a}{b}}$, which is accurate when $a = O(b)$ [18]. Another class of approximations will involve interpreting a finite sum as a definite integral. Under any reasonable choice of tradeoff parameters, these approximations will be very accurate, whenever we use them, and they will be written as equalities rather than as approximations.

## 3 Analysis of the Fuzzy Rainbow Tradeoff

The online behavior of the fuzzy rainbow tradeoff is analyzed in this section. The analysis will focus on the average online time complexity, rather than the worst case time complexity, and take the effects of false alarms fully into account.

### 3.1 Probability of Success

Comparisons of different tradeoff algorithms must surely be done under identical requirements on the success probability of inversion. This subsection is devoted to obtaining an expression for the success rate of the fuzzy rainbow tradeoff.

Let us define the *pre-computation coefficient* of a fuzzy rainbow tradeoff to be

$$\mathtt{F}_{\mathrm{pc}} = \frac{mts\ell}{\mathsf{N}}, \tag{4}$$

so that $\mathtt{F}_{\mathrm{pc}}\mathsf{N}$ is the cost of pre-computation. We also define the *coverage rate* of a fuzzy rainbow matrix as

$$\mathtt{F}_{\mathrm{cr}} = \frac{1}{mts}\big(|\mathtt{DM}_1| + |\mathtt{DM}_2| + \cdots + |\mathtt{DM}_s|\big), \tag{5}$$

where $|\mathtt{DM}_i|$ is the number of distinct points expected in the $i$-th DP sub-matrix. The following proposition, which is easy to prove, shows that this is a natural definition.

**Proposition 1.** *The success probability of the fuzzy rainbow tradeoff is* $\mathtt{F}_{\mathrm{ps}} = 1 - e^{-\mathtt{F}_{\mathrm{cr}}\mathtt{F}_{\mathrm{pc}}}$.

To utilize this proposition, we need a way to express the coverage rate in terms of the tradeoff algorithm parameters. Recall from [17, 20] that the number of distinct entries $|\mathtt{DM}|$ contained in a non-perfect DP matrix satisfies

$$|\mathtt{DM}| = m_{\mathrm{ep}} t, \tag{6}$$

where $m_{\mathrm{ep}}$ denotes the number of distinct ending points of the DP matrix. The work [20] uses this fact in conjunction with another expression for $|\mathtt{DM}|$, found in [18], to obtain the formula

$$m_{\mathrm{ep}} = m_{\mathrm{sp}} \frac{2}{1 + \sqrt{1 + \frac{2 m_{\mathrm{sp}} t^2}{\mathsf{N}}}}, \tag{7}$$

that relates the number of distinct starting points $m_{\mathrm{sp}}$ to the number of distinct ending points $m_{\mathrm{ep}}$ of a non-perfect DP matrix.

Returning to the fuzzy rainbow matrices, let us use $m_{i-1}$ and $m_i$ to denote the number of distinct starting points and ending points, respectively, expected in each DP sub-matrix $\mathtt{DM}_i$. In particular, $m_0 = m$ and $m_s$ are the numbers of distinct starting and ending points, respectively, of the full fuzzy rainbow matrix. We shall refer to each collection of $m_i$ points as the *i-th color boundary points* of a fuzzy rainbow matrix, where $i = 0, 1, \ldots, s$. Adopting the above two facts to our situation, we can state that

$$|\mathtt{DM}_i| = m_i t \tag{8}$$

and

$$m_{i+1} = m_i \frac{2}{1 + \sqrt{1 + \frac{2 m_i t^2}{\mathsf{N}}}} \quad \text{with} \quad m_0 = m \tag{9}$$

are to be expected at each color index $i$. The following closed-form formula for $m_i$ is easier to handle than the iterative formula (9).

**Lemma 1.** *When the number of colors $s$ used in each pre-computation table is large, the number of i-th color boundary points in a fuzzy rainbow matrix is expected to be $m_i = \frac{2m}{2 + \mathtt{F}_{\mathrm{msc}} \frac{i}{s}}$, for $i = 0, 1, \ldots, s$.*

*Proof.* Since the assumption is that $\frac{m_i t^2}{\mathsf{N}} = O\big(\frac{1}{s}\big)$ is small, one can use series expansion to approximate the iterative formula (9) by $m_{i+1} = m_i\big(1 - \frac{1}{2}\frac{m_i t^2}{\mathsf{N}}\big)$. After rewriting this as a certain difference equation, one can apply the Euler method to view it as the differential equation $y' = -\frac{\mathtt{F}_{\mathrm{msc}}}{2s} y^2$, with the initial condition $y(0) = \frac{m_0}{m} = 1$, and solve for $y$ to obtain $\frac{m_i}{m} = \frac{2}{2 + \mathtt{F}_{\mathrm{msc}} \frac{i}{s}}$. □

The coverage rate given below follows from a combination of Lemma 1 and (8).

**Proposition 2.** *When the number of colors $s$ is large, we have*

$$\frac{1}{mts} \sum_{k=i+1}^{s} |\mathtt{DM}_k| = \frac{2}{\mathtt{F}_{\mathrm{msc}}} \ln\left(\frac{2 + \mathtt{F}_{\mathrm{msc}}}{2 + \mathtt{F}_{\mathrm{msc}} \frac{i}{s}}\right).$$

*In particular, the coverage rate is* $\mathtt{F}_{\mathrm{cr}} = \frac{2}{\mathtt{F}_{\mathrm{msc}}} \ln\left(1 + \frac{\mathtt{F}_{\mathrm{msc}}}{2}\right)$*, for a single fuzzy rainbow matrix.*

*Remark 1.* The statement of Proposition 2 assumes $s$ to be large, but when explicit values of $\frac{m_i}{m}$ are iteratively computed through (9) and compared against the values given by Lemma 1, the two are seen to agree accurately, even for small positive integer values of $s$. Hence, we shall treat Lemma 1 and Proposition 2 as being valid for all $s$ values of interest.

*Remark 2.* Proposition 1 implies that any set of parameters $m$, $t$, $s$, and $\ell$ that achieves the success rate $\mathtt{F}_{\mathrm{ps}}$ must satisfy the relation

$$\frac{\ell}{t} = \frac{\mathtt{F}_{\mathrm{pc}}}{\mathtt{F}_{\mathrm{msc}}} = \frac{\{-\ln(1-\mathtt{F}_{\mathrm{ps}})\}}{\mathtt{F}_{\mathrm{msc}}\mathtt{F}_{\mathrm{cr}}}.$$

Recall that we are working with parameters for which $\mathtt{F}_{\mathrm{msc}}$ is of $\Theta(1)$ order, so that, according to Proposition 2, the coverage rate $\mathtt{F}_{\mathrm{cr}}$ is also of $\Theta(1)$ order. Hence, unless the success rate requirement $\mathtt{F}_{\mathrm{ps}}$ is unrealistically close to 100%, parameters $\ell$ and $t$ must be of similar order.

## 3.2 Online Complexity

Having secured full knowledge concerning the success rate of the fuzzy rainbow tradeoff, we next discuss the online execution complexities. We start with the cost of generating the online chains. Note that our interest is in the average case complexity, rather than the worst case complexity.

We first need to obtain the probability for each pass of the online phase to be executed.

**Lemma 2.** *The probability for the online chains that start from the i-th color of each fuzzy rainbow matrix to be generated is*

$$\left(\frac{2+\mathtt{F}_{\mathrm{msc}}\frac{i}{s}}{2+\mathtt{F}_{\mathrm{msc}}}\right)^{2\frac{\ell}{t}}.$$

*Proof.* The online chains that start from the $i$-th color of each fuzzy rainbow matrix will be generated if and only if the correct answer to the inversion target does not belong to the DP sub-matrices $\mathtt{DM}_{i+1}, \ldots, \mathtt{DM}_s$ contained in the $\ell$ fuzzy rainbow matrices. Hence, the probability under consideration is

$$\prod_{j=i+1}^{s}\left(1-\frac{|\mathtt{DM}_j|}{\mathsf{N}}\right)^{\ell} = \exp\left(-\frac{\ell}{t}\mathtt{F}_{\mathrm{msc}}\sum_{j=i+1}^{s}\frac{|\mathtt{DM}_j|}{mts}\right) = \left(\frac{2+\mathtt{F}_{\mathrm{msc}}\frac{i}{s}}{2+\mathtt{F}_{\mathrm{msc}}}\right)^{2\frac{\ell}{t}},$$

where the second equality follows from Proposition 2 and Remark 1. □

Since each of the $\ell$ online chains that start from the $i$-th color is expected to require $(s-i+1)t$ iterations of the one-way function, the following statement is a direct consequence of Lemma 2.

**Proposition 3.** *The generation of the online chains during the online phase of a fuzzy rainbow tradeoff is expected to require*

$$t\ell \sum_{i=1}^{s} (s-i+1)\Big(\frac{2+\mathrm{F}_{\mathrm{msc}}\frac{i}{s}}{2+\mathrm{F}_{\mathrm{msc}}}\Big)^{2\frac{\ell}{t}}$$

*iterations of the one-way function.*

The remaining component of the online time complexity is the cost of dealing with alarms. The proof of the following claim is quite technical and can be found in [19].

**Proposition 4.** *The resolving of alarms during the online phase of a fuzzy rainbow tradeoff is expected to require*

$$t\ell\,\frac{\mathrm{F}_{\mathrm{msc}}}{s} \sum_{i=1}^{s} \big\{i(s-i+1)+1\big\}\Big(\frac{2+\mathrm{F}_{\mathrm{msc}}\frac{i}{s}}{2+\mathrm{F}_{\mathrm{msc}}}\Big)^{2\frac{\ell}{t}}$$

*iterations of the one-way function.*

We can now combine the two components of the online time complexity for the fuzzy rainbow tradeoff and state its tradeoff coefficient.

**Theorem 1.** *The time memory tradeoff curve for the non-perfect table fuzzy rainbow tradeoff that uses $s$ colors per pre-computation matrix is $TM^2 = \mathrm{F}_{\mathrm{tc},s}\mathsf{N}^2$, where the tradeoff coefficient is*

$$\mathrm{F}_{\mathrm{tc},s} = \mathrm{F}_{\mathrm{msc}}^2\Big(\frac{\ell}{t}\Big)^3 \sum_{i=1}^{s} \bigg[\Big(1-\frac{i-1}{s}\Big)\Big(1+\mathrm{F}_{\mathrm{msc}}\frac{i}{s}\Big)+\frac{\mathrm{F}_{\mathrm{msc}}}{s^2}\bigg]\Big(\frac{2+\mathrm{F}_{\mathrm{msc}}\frac{i}{s}}{2+\mathrm{F}_{\mathrm{msc}}}\Big)^{2\frac{\ell}{t}}\frac{1}{s}.$$

*Proof.* The storage complexity of the fuzzy rainbow tradeoff is $M = m\ell$, and the time complexity is the sum

$$T = t\ell \sum_{i=1}^{s} \Big\{(s-i+1)\Big(\mathrm{F}_{\mathrm{msc}}\frac{i}{s}+1\Big)+\frac{\mathrm{F}_{\mathrm{msc}}}{s}\Big\}\Big(\frac{2+\mathrm{F}_{\mathrm{msc}}\frac{i}{s}}{2+\mathrm{F}_{\mathrm{msc}}}\Big)^{2\frac{\ell}{t}}$$

of the two terms given by Proposition 3 and Proposition 4. The tradeoff curve is obtained by appropriately combining the two complexities $M$ and $T$. □

In passing, we state that Lemma 2 allows the number of table lookups expected during the online phase to be written as $\ell\sum_{i=1}^{s}\big(\frac{2+\mathrm{F}_{\mathrm{msc}}\frac{i}{s}}{2+\mathrm{F}_{\mathrm{msc}}}\big)^{2\frac{\ell}{t}}$, which is of $\Theta(ts)$ order.

## 3.3 Storage Optimization

The storage complexity $M$ appearing in the tradeoff curve of Theorem 1 refers to the total number of entries, i.e., starting and ending point pairs, in the pre-computation tables. In practice, the physical storage size, which depends not only on the number of table entries, but also on how many bits of storage must be allocated to each table entry, will be more important.

The use of sequentially generated starting points [5, 11, 12] allows each starting point to be recorded in $\log m$ bits. The issue of storing the ending points effectively is more complicated. The known techniques are the removal of ending point portions that can be recovered from the definition of the distinguishing property [11, 24], truncation of ending points[8, 11], and the index file method [11]. A brief explanation of these techniques can be found in [18, 20].

The first and third techniques are rather well known and will not be discussed here. The second technique is to truncate the ending points to a certain length before recording them to storage. During the online phase, the terminating DP of an online chain is likewise truncated before being searched for in the pre-computation table. Truncation reduces the physical storage requirement, but causes occasional alarms to be generated even when the online chain did not merge into any pre-computation chain. Thus, we need to find the degree of truncation that restricts the side effects to a negligible fraction of the online time complexity.

Let us assume a fixed truncation method for ending points with a truncated match probability of $\frac{1}{r}$. That is, we assume that the truncated outcome of two random ending points, which are a priori DPs, will be identical with probability $\frac{1}{r}$. For example, one could retain $\log r$ bits of each ending point, that are unrelated to the distinguishing property, to obtain this effect.

An explicit expression for the expected extra cost of dealing with truncation-related alarms is given in [19]. We will not go into the details, but one can deduce from this result that the extra cost is of $\Theta\big(t^2 s^2 \frac{m}{r}\big)$ order. On the other hand, similar treatment of the time complexity $T$, appearing in the proof of Theorem 1, shows that $T$ is of $\Theta(t^2 s^2)$ order. Hence, if $\frac{m}{r}$ is a sufficiently small fraction, the added cost of treating truncation-related alarms will be insignificant in comparison to the time complexity $T$ of the algorithm that does not use truncation of ending points. In other words, the side effects of ending point truncation will be negligible if the truncated ending points contain slightly more than $\log m$ bits of information. Of course, the ending point portion that can be recovered from the DP definition contain no information, and almost $\log m$ bits of the ending point can be further removed without any information loss through the index table method.

In summary, storage of each starting point of the fuzzy rainbow tradeoff requires $\log m$ bits and the storage of each ending point requires a very small number $\varepsilon$ of bits. Each entry of the fuzzy rainbow tradeoff can be recorded in $\log m + \varepsilon$ bits.

## 4 Comparison of Tradeoff Algorithms

To compare the performances tradeoff algorithms, we follow the framework of [18] and present the pre-computation coefficient versus tradeoff coefficient curves for the non-perfect fuzzy rainbow, perfect rainbow, and non-perfect rainbow tradeoffs, at several fixed success rate requirements. The curves present the range of options made available by each algorithm concerning the degree of online efficiency that can be achieved after the investment of certain amount of pre-computation effort. The perfect and non-perfect rainbow tradeoffs were chosen as the comparison targets, because the two were shown by the recent works [18, 20] to be the most competitive algorithms among the five major tradeoff algorithms, under typical conditions.

The information required to draw the curves for the comparison target algorithms can be found in [18, 20]. The fuzzy rainbow tradeoff similarly allows the $\mathtt{F}_{\mathrm{pc}}$ versus $\mathtt{F}_{\mathrm{tc},s}$ curve, under fixed $\mathtt{F}_{\mathrm{ps}}$ and $s$, to be plotted as a curve parameterized by the single variable $\mathtt{F}_{\mathrm{msc}}$.

## 4.1 Optimal $s$

Before comparing the fuzzy rainbow tradeoff with the original rainbow tradeoff, let us first discuss the effects of the parameter $s$ on the performance of the fuzzy rainbow tradeoff.

Consider a fuzzy rainbow tradeoff implementer that is working with a certain fixed $s$. Suppose that a set of parameters $m$, $t$, and $\ell$ achieving all desired properties have been found. These parameters need not be optimal in any particular sense, and we only assume that the desired success rate is met, while the balance between resource requirements, such as pre-computation cost, physical storage size, and expected online time, is suitable for the intended purpose and environment. The implementer is willing to further tweak the parameters slightly, but large changes to $\log m$, $\log t$, and $\log \ell$, that would destroy the favorable balance between pre-computation cost, storage size, and online time will not be made.

The implementer still wishes to explore the possibility of using a different $s$ values, as long as the three resource requirements remain largely unchanged. Consider another $s$-value $s'$ and the associated parameter set $m' = \frac{s'}{s}m$, $t' = \frac{s}{s'}t$, and $\ell' = \frac{s}{s'}\ell$. It is clear that the pre-computation coefficient $\mathtt{F}_{\mathrm{pc}} = \frac{mts\ell}{\mathsf{N}} = \frac{m't's'\ell'}{\mathsf{N}} = \mathtt{F}'_{\mathrm{pc}}$ and number of table entries $M = m\ell = m'\ell' = M'$, for the new set of parameters, are identical to those of the original parameters. Furthermore, $\mathtt{F}_{\mathrm{msc}} = \frac{mt^2s}{\mathsf{N}} = \frac{m't'^2s'}{\mathsf{N}} = \mathtt{F}'_{\mathrm{msc}}$ implies that the coverage rate $\mathtt{F}_{\mathrm{cr}}$ (Proposition 2) and success rate $\mathtt{F}_{\mathrm{ps}}$ (Proposition 1) also remain unchanged.

Since the algorithm behavior corresponding to the two sets of parameters seems to differ only in their time complexities $T'$ and $T$, it may be tempting to simply compare $\mathtt{F}_{\mathrm{tc},s'} = \frac{T'M'^2}{\mathsf{N}^2} = \frac{T'M^2}{\mathsf{N}^2}$ against $\mathtt{F}_{\mathrm{tc},s} = \frac{TM^2}{\mathsf{N}}$ to determine which of the two are better. However, one must keep in mind that $M$ does not represent the physical amount of storage. Recall from Section 3.3 that $\log m + \varepsilon$ bits of storage are required per table entry, where $\varepsilon$ corresponds to the bits used to record the partial ending points that remain after applications of the ending point truncation and the index table techniques. Hence, to be fair, one must compare

$$(\log m+\varepsilon)^2\,\mathtt{F}_{\mathrm{tc},s} \quad \text{and} \quad (\log m'+\varepsilon)^2\,\mathtt{F}_{\mathrm{tc},s'} = \Big\{\log m+\varepsilon+\log\Big(\frac{s'}{s}\Big)\Big\}^2\,\mathtt{F}_{\mathrm{tc},s'} \quad (10)$$

against each other.

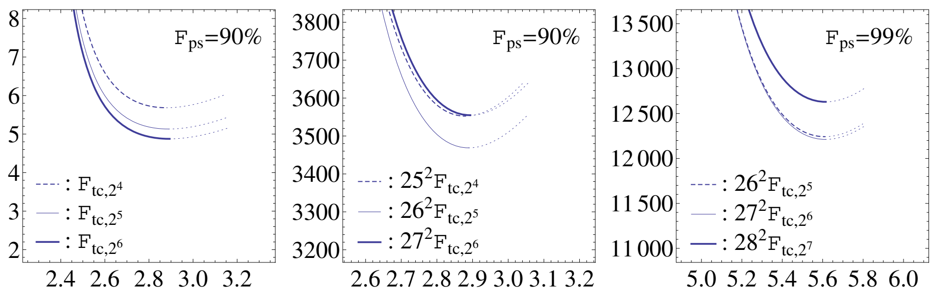


**Fig. 1.** The tradeoff coefficient $F_{tc}$ in relation to their respective pre-computation costs, at small $s$ ($x$-axis: pre-computation coefficient; $y$-axis: tradeoff coefficients)

Let us briefly work with explicit example figures. The first box of Figure 1 presents the $F_{pc}$ versus $F_{tc,s}$ curves at 90% success rate, for the $s = 2^4$, $s = 2^5$, and $s = 2^6$ cases. The $x$-axis gives the pre-computation coefficient $F_{pc}$ and the $y$-axis gives the tradeoff coefficients $F_{tc,s}$. The lower parts in each box correspond to better online efficiency and parts closer to the left edge correspond to smaller pre-computation requirements. The parameters corresponding to dotted parts of the curves should not be used, since they correspond to worse online efficiency at more pre-computation cost than the lowest point of each curve. It is evident that increasing the $s$ value brings the $F_{pc}$ versus $F_{tc,s}$ curve closer to the bottom left corner. However, as previously noted, this observation alone should not be used to draw any premature conclusions.

Suppose that a certain parameter set appropriate for the resources available to the tradeoff implementer is such that $\log m + \varepsilon = 25$, when $s = 2^4$ is used. Under this situation, the discussion above shows that, in order to compare the performances of the fuzzy rainbow tradeoffs running with different $s$, we should be focusing on the second box of Figure 1 that present curves for the adjusted tradeoff coefficients. The change from $s = 2^4$ to $s = 2^5$ follows the trend seen in the first box, but the transition from $s = 2^5$ to $s = 2^6$ does not, worsening the performance of the fuzzy rainbow tradeoff. The choice of $s = 2^5$ is seen to be optimal, at least among the powers of 2, for this situation.

The optimal choice of $s$ certainly would have been different if we had started from a different pairing of $\log m + \varepsilon$ and $s$. Comparison of the second and third boxes of Figure 1 shows that the optimal choice of $s$ also depends on the success rate requirement.

In summary, using a larger $s$ reduces the tradeoff coefficient $F_{tc,s}$, but increases the number of bits required to store each table entry. In order to find the value of $s$ that is optimal for the specific situation in hand, it suffices to draw curves, similar to the second and third boxes of Figure 1, corresponding to various $s$ options, with appropriate adjustment factors multiplied to $F_{tc,s}$. A table of optimal $s$ values for each situation is given in the full version [19] of this article.

## 4.2 Fuzzy Rainbow Tradeoff versus Rainbow Tradeoff

To compare the fuzzy rainbow tradeoff directly with the perfect and non-perfect rainbow tradeoffs, we need to find the appropriate adjustment factors to be multiplied to the tradeoff coefficients, and this starts with a discussion of the fuzzy rainbow tradeoff parameters $m_{\mathrm{F}}$, $t_{\mathrm{F}}$, $\ell_{\mathrm{F}}$, and $s$ that would make the resource requirements of the algorithm comparable to those of the perfect or non-perfect rainbow tradeoffs running under parameters $m_{\mathrm{R}}$, $t_{\mathrm{R}}$, and $\ell_{\mathrm{R}}$.

We had mentioned in Section 3.3 that the time complexity for the fuzzy rainbow tradeoff satisfies $T_{\mathrm{F}} \approx t_{\mathrm{F}}^2 s^2$ and we know that the time complexity for the two usual rainbow tradeoffs satisfies $T_{\mathrm{R}} \approx t_{\mathrm{R}}^2$. Equating the very rough time and storage complexities of the two algorithms and using the facts $\ell_{\mathrm{F}} \approx t_{\mathrm{F}}$ (Remark 2) and $\ell_{\mathrm{R}} \approx 1$, we see that one must require

$$t_{\mathrm{F}}^2 s^2 \approx t_{\mathrm{R}}^2 \quad \text{and} \quad m_{\mathrm{F}} t_{\mathrm{F}} \approx m_{\mathrm{F}} \ell_{\mathrm{F}} \approx m_{\mathrm{R}} \ell_{\mathrm{R}} \approx m_{\mathrm{R}}. \tag{11}$$

These should be taken as extremely rough requirements, but the relations

$$\log t_{\mathrm{F}} + \log s \approx \log t_{\mathrm{R}} \quad \text{and} \quad \log m_{\mathrm{F}} + \log t_{\mathrm{F}} \approx \log m_{\mathrm{R}} \tag{12}$$

are somewhat reasonably accurate requirements one should adhere to, if the two algorithms are to be using similar resources.

Recall from [18] and [20] that each pre-computation table entry of both the perfect and non-perfect rainbow tradeoffs consumes $\log m_{\mathrm{R}} + \varepsilon_{\mathrm{R}}$ bits of storage, where $\varepsilon_{\mathrm{R}}$ is a small positive integer. We have already seen that the fuzzy rainbow tradeoff similarly consumes $\log m_{\mathrm{F}} + \varepsilon_{\mathrm{F}}$ bits of storage per table entry.

Our interest lies in the ratio of bits required per table entry, and the use of (12) implies

$$\frac{\log m_{\mathrm{F}} + \varepsilon_{\mathrm{F}}}{\log m_{\mathrm{R}} + \varepsilon_{\mathrm{R}}} \approx \frac{\log m_{\mathrm{R}} - \log t_{\mathrm{R}} + \log s + \varepsilon_{\mathrm{F}}}{\log m_{\mathrm{R}} + \varepsilon_{\mathrm{R}}} \approx \frac{\frac{1}{3}\log \mathsf{N} + \log s + \varepsilon_{\mathrm{F}}}{\frac{2}{3}\log \mathsf{N} + \varepsilon_{\mathrm{R}}}, \tag{13}$$

where the second approximation is for the parameter set $m_{\mathrm{R}} = \mathsf{N}^{\frac{2}{3}}$ and $t_{\mathrm{R}} = \mathsf{N}^{\frac{1}{3}}$ that is typically considered during theoretical analyses of tradeoff algorithms. In the extreme, by ignoring the small integers $\varepsilon_{\mathrm{F}}$ and $\varepsilon_{\mathrm{R}}$, and also assuming $s$ to be small, one might argue that this ratio could be as small as $\frac{1}{2}$, at the theoretically typical parameters. However, this is a rather optimistic figure that is biased in favor of the fuzzy rainbow tradeoff.

Consider the very large example of $\log \mathsf{N} \approx 75$ and the corresponding theoretically typical parameter set

$$\log m_{\mathrm{R}} \approx 50 \quad \text{and} \quad \log t_{\mathrm{R}} \approx 25, \tag{14}$$

for the rainbow tradeoff. According to (12), the parameter set for the fuzzy rainbow tradeoff of $s = 2^5$ that calls for comparable resources would satisfy

$$\log m_{\mathrm{F}} \approx 30 \quad \text{and} \quad \log t_{\mathrm{F}} \approx 20. \tag{15}$$

When $\varepsilon_{\mathtt{R}} \approx \varepsilon_{\mathtt{F}} \approx 7$, the ratio of bits per table entry is $\frac{\log m_{\mathtt{F}}+\varepsilon_{\mathtt{F}}}{\log m_{\mathtt{R}}+\varepsilon_{\mathtt{R}}} \approx \frac{37}{57}$. Hence, if one's favored balance of resources corresponds to rainbow tradeoff parameters of (14), one must compare the adjusted tradeoff coefficient $\left(\frac{37}{57}\right)^2 \mathtt{F}_{\mathrm{tc},2^5} = 0.42\,\mathtt{F}_{\mathrm{tc},2^5}$ against $\mathtt{R}_{\mathrm{tc}}$ (non-perfect rainbow) and $\bar{\mathtt{R}}_{\mathrm{tc}}$ (perfect rainbow).

Similarly, for the rather small example of $\log \mathsf{N} \approx 40$, $\log m_{\mathtt{R}} \approx 27$, $\log t_{\mathtt{R}} \approx 13$, $\varepsilon_{\mathtt{R}} \approx 7$, and $\varepsilon_{\mathtt{F}} \approx 7$, with $\log s \approx 5$, the ratio would be $\frac{\log m_{\mathtt{F}}+\varepsilon_{\mathtt{F}}}{\log m_{\mathtt{R}}+\varepsilon_{\mathtt{R}}} \approx \frac{13}{17}$, and a fair comparison would let $\left(\frac{13}{17}\right)^2 \mathtt{F}_{\mathrm{tc},2^5} = 0.58\,\mathtt{F}_{\mathrm{tc},2^5}$ compete against $\mathtt{R}_{\mathrm{tc}}$ and $\bar{\mathtt{R}}_{\mathrm{tc}}$.

Different $\mathtt{F}_{\mathrm{tc},s}$ adjustment factors will need to be applied, depending on $s$ and the rough range of online resources that are appropriate for the situation in hand. However, taking the experience obtained from Figure 1 and the above two examples into account, we will somewhat arbitrarily choose to compare the two explicit choices $0.5\,\mathtt{F}_{\mathrm{tc},2^4}$ and $0.5\,\mathtt{F}_{\mathrm{tc},2^6}$ against $\mathtt{R}_{\mathrm{tc}}$ and $\bar{\mathtt{R}}_{\mathrm{tc}}$. The full version [19] of this paper treats the adjustment factors and $s$ values more carefully.

We clearly state that there could be situations that call for a tradeoff coefficient adjustment factor that is much larger than the 0.5 we will be using. The appropriate adjustment factor depends not only on the externally given implementation environment, but also on the taste of the implementer concerning the balance between online efficiency and pre-computation cost, so that the adjustment factor cannot be fixed in an objective manner. In any case, the discussion given below can easily be adjusted to work for any specific situation.

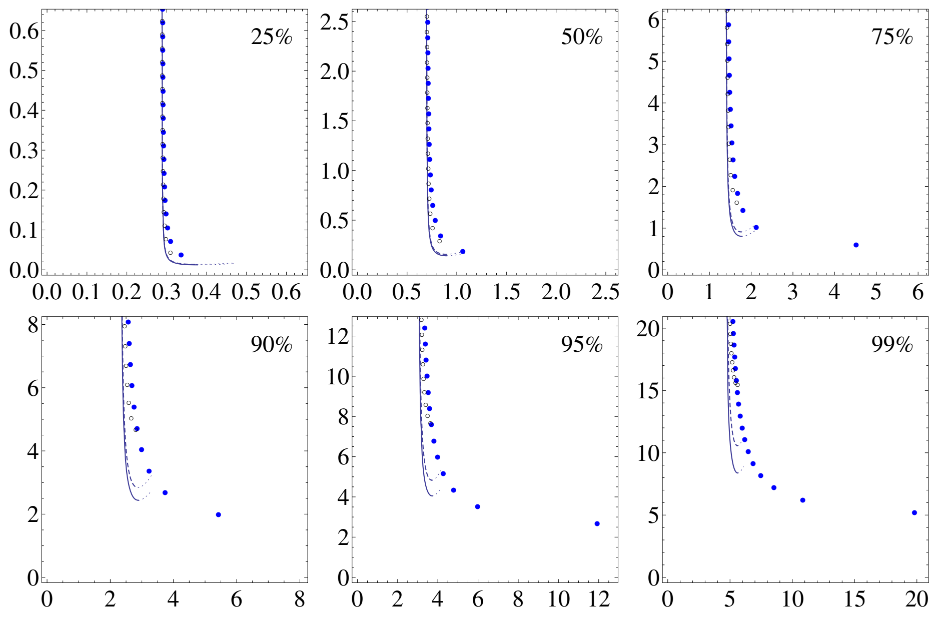


**Fig. 2.** The tradeoff coefficients $\mathtt{R}_{\mathrm{tc}}$ (empty circles), $\bar{\mathtt{R}}_{\mathrm{tc}}$ (filled dots), $0.5\,\mathtt{F}_{\mathrm{tc},2^4}$ (dashed), and $0.5\,\mathtt{F}_{\mathrm{tc},2^6}$ (line), in relation to their respective pre-computation costs, at various success rates ($x$-axis: pre-computation coefficients; $y$-axis: tradeoff coefficients).

The pre-computation coefficient versus tradeoff coefficient curves for the various tradeoff algorithms are given in Figure 2. Each box presents data corresponding to the success rate requirement indicated at its upper right corner. The curves for the fuzzy rainbow tradeoffs are given as the dashed line ($s = 2^4$ case) and the thin line ($s = 2^6$ case). As mentioned before, the dotted parts of the curves should be ignored. The data for the perfect rainbow tradeoff (filled dots) and non-perfect rainbow tradeoff (empty circles) appear as discrete set of points, due to their small number of tables.

In all the boxes the two curves for the fuzzy rainbow tradeoff are situated closer to the lower left corner than the data points for the two rainbow tradeoffs. The fuzzy rainbow tradeoff provides better online efficiency for the same pre-computation investment. Thus a very rough conclusion would be that the fuzzy rainbow tradeoff is better in performance than the perfect and non-perfect rainbow tradeoffs.

At the 75% and higher success rates, the lowest dot for the perfect rainbow curve is situated lower than the lowest point of the two fuzzy rainbow tradeoff curves. This shows that the perfect rainbow tradeoff is able to provide better online efficiency than the fuzzy rainbow tradeoff at high success rates. That is, no choice of fuzzy rainbow tradeoff parameters will make its online efficiency better than the optimal efficiency reachable with the perfect rainbow tradeoff. Hence, the perfect rainbow tradeoff can be advantageous over the fuzzy rainbow tradeoff when the success rate requirement is high and the online efficiency is important.

However, it must be understood that the higher online efficiency option can be utilized only if it is paid for with higher pre-computation cost. Since the pre-computation cost is the largest barrier in any large scale deployment of the tradeoff technique, the higher cost cannot be ignored, even though it could be amortized through multiple uses of the online phase. At the high success rates, the decision as to whether the fuzzy rainbow or the perfect rainbow tradeoff is better will be different depending on how costly the additional pre-computation will be, relative to the value of better online efficiency, with its multiple uses taken into account, to the implementer.

At the low success rates 25% and 50%, the lowest point of the fuzzy rainbow curve is lower than the lowest point of the two original rainbow tradeoffs. For these low success rates, fuzzy rainbow tradeoff is always advantageous over the two original rainbow tradeoffs in terms of both the online efficiency and pre-computation cost.

Finally, a second pass through all six boxes, with focus on the empty circles, reveals that the performance of the non-perfect rainbow tradeoff is always inferior to that of the fuzzy rainbow tradeoff. The degree of online efficiency that can be provided by the non-perfect rainbow tradeoff can always be obtained with the fuzzy rainbow tradeoff at a lower pre-computation cost.

## 5 Conclusion

The online execution behavior of the non-perfect table fuzzy rainbow tradeoff was analyzed in this work. The success rate, the accurate average case online execution time that accounts for false alarms, and the physical storage size required to hold the pre-computation tables have all been obtained.

The information obtained through our analyses was used to compare the fuzzy rainbow tradeoff against the original rainbow tradeoff algorithm, which is widely taken to be the best tradeoff algorithm. The tradeoff coefficient adjustment factor and the color count $s$ per table had to be fixed to somewhat arbitrary values for the comparison. However, our choices were based on figures obtained from reasonable examples, and the ensuing conclusions should be valid for the most part of the practical parameter range. Furthermore, the process can be repeated easily for any other choices, should there be the need to work with a drastically different range of parameters.

We discovered that the fuzzy rainbow tradeoff is always advantageous over the non-perfect rainbow tradeoff. The fuzzy rainbow tradeoff also outperforms the perfect rainbow tradeoff at low success rate requirements. For high success rate requirements, the situation is less conclusive. It is possible for the perfect rainbow tradeoff to provide online efficiency that cannot be reached by the fuzzy rainbow tradeoff, but the advantage must be paid for with higher pre-computation cost. For efficiency levels that are reachable by both algorithms, the fuzzy rainbow tradeoff required less pre-computation.

It remains to analyze the perfect table version of the fuzzy rainbow tradeoff. The good performance of the non-perfect table fuzzy rainbow tradeoff witnessed through this work is an optimistic sign. On the other hand, the relatively poor performance of the perfect table DP tradeoff [20] could be interpreted as a negative indication.

## References

1. Cryptohaze, GPU Rainbow Cracker, `https://www.cryptohaze.com/`
2. L0phtCrack, L0phtCrack 6, `http://www.l0phtcrack.com/`
3. Objectif Sécurité, Ophcrack, `http://ophcrack.sourceforge.net/`
4. RainbowCrack Project, RainbowCrack and RainbowCrack for GPU, `http://project-rainbowcrack.com/`
5. Avoine, G., Junod, P., Oechslin, P.: Characterization and improvement of time-memory trade-off based on perfect tables. ACM Trans. Inform. Syst. Secur. 11(4), 17:1–17:22 (2008); Preliminary version presented at INDOCRYPT 2005
6. Barkan, E.P.: Cryptanalysis of Ciphers and Protocols. Ph.D. Thesis, Technion—Israel Institute of Technology (March 2006)
7. Barkan, E., Biham, E., Keller, N.: Instant ciphertext-only cryptanalysis of GSM encrypted communication. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 600–616. Springer, Heidelberg (2003)
8. Barkan, E., Biham, E., Shamir, A.: Rigorous bounds on cryptanalytic time/Memory tradeoffs. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 1–21. Springer, Heidelberg (2006)
9. Biryukov, A., Mukhopadhyay, S., Sarkar, P.: Improved time-memory trade-offs with multiple data. In: Preneel, B., Tavares, S. (eds.) SAC 2005. LNCS, vol. 3897, pp. 110–127. Springer, Heidelberg (2006)

10. Biryukov, A., Shamir, A.: Cryptanalytic time/memory/data tradeoffs for stream ciphers. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 1–13. Springer, Heidelberg (2000)
11. Biryukov, A., Shamir, A., Wagner, D.: Real time cryptanalysis of A5/1 on a PC. In: Schneier, B. (ed.) FSE 2000. LNCS, vol. 1978, pp. 1–18. Springer, Heidelberg (2001)
12. Borst, J.: Block Ciphers: Design, Analysis, and Side-Channel Analysis. Ph.D. Thesis, Katholieke Universiteit Leuven (September 2001)
13. Borst, J., Preneel, B., Vandewalle, J.: On the time-memory tradeoff betweeen exhaustive key search and table precomputation. In: Proceedings of the 19th Symposium on Information Theory in the Benelux, WIC (1998)
14. Denning, D.E.: Cryptography and Data Security[1]. Addison-Wesley (1982)
15. Golić, J.D.: Cryptanalysis of alleged A5 stream cipher. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 239–255. Springer, Heidelberg (1997)
16. Hellman, M.E.: A cryptanalytic time-memory trade-off. IEEE Trans. on Infor. Theory 26, 401–406 (1980)
17. Hong, J., Lee, G.W., Ma, D.: Analysis of the parallel distinguished point tradeoff. In: Bernstein, D.J., Chatterjee, S. (eds.) INDOCRYPT 2011. LNCS, vol. 7107, pp. 161–180. Springer, Heidelberg (2011)
18. Hong, J., Moon, S.: A comparison of cryptanalytic tradeoff algorithms. To appear in J. Cryptology, `http://dx.doi.org/10.1007/s00145-012-9128-3`
19. Kim, B.-I., Hong, J.: Analysis of the non-perfect table fuzzy rainbow tradeoff. IACR Cryptology ePrint Archive, Report 2012/612, version 20121116:123317 (2012)
20. Lee, G.W., Hong, J.: A comparison of perfect table cryptanalytic tradeoff algorithms. IACR Cryptology ePrint Archive, Report 2012/540 (2012)
21. Nohl, K.: Attacking phone privacy. Presented at Black Hat USA 2010, Las Vegas (July 2010)
22. Nohl, K., Paget, C.: GSM-SRSLY? Presented at 26th Chaos Communication Congress (26C3), Berlin (December 2009)
23. Oechslin, P.: Making a faster cryptanalytic time-memory trade-off. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 617–630. Springer, Heidelberg (2003)
24. Standaert, F.-X., Rouvroy, G., Quisquater, J.-J., Legat, J.-D.: A time-memory tradeoff using distinguished points: New analysis & FPGA results. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 593–609. Springer, Heidelberg (2003)
25. van den Broek, F., Poll, E.: A comparison of time-memory trade-off attacks on stream ciphers. In: Youssef, A., Nitaj, A., Hassanien, A.E. (eds.) AFRICACRYPT 2013. LNCS, vol. 7918, pp. 406–423. Springer, Heidelberg (2013)

[1] On p.100, credit is given to Rivest for suggesting to apply the notion of distinguished points to the classical Hellman tradeoff.

# Complexity of Increasing the Secure Connectivity in Wireless Ad Hoc Networks

Seyit A. Camtepe

Queensland University of Technology

**Abstract.** We consider the problem of maximizing the secure connectivity in wireless ad hoc networks, and analyze complexity of the post-deployment key establishment process constrained by physical layer properties such as connectivity, energy consumption and interference. Two approaches, based on graph augmentation problems with nonlinear edge costs, are formulated. The first one is based on establishing a secret key using only the links that are already secured by shared keys. This problem is in NP-hard and does not accept polynomial time approximation scheme PTAS since minimum cutsets to be augmented do not admit constant costs. The second one extends the first problem by increasing the power level between a pair of nodes that has a secret key to enable them physically connect. This problem can be formulated as the optimal key establishment problem with interference constraints with bi-objectives: (i) maximizing the concurrent key establishment flow, (ii) minimizing the cost. We prove that both problems are NP-hard and MAX-SNP (i.e., it is NP-hard to approximate them within a factor of $1 + \epsilon$ for $\epsilon > 0$) with a reduction to MAX3SAT problem.

## 1 Introduction

Efficient key management schemes are essential to ensure the integrity and confidentiality in wireless ad hoc networks. An example of such networks are wireless sensor networks operating in adversarial conditions. Many different key management schemes are proposed for the wireless sensor networks. Some solutions assign (a.k.a. pre-distribute) each node a key-chain, a set of symmetric keys or keying materials (e.g., ID, master keys, hash functions, pseudo random functions, shared polynomials, key matrices and location information), to be shared with *some* of its neighbors after deployment with high probability. Others are based on trusted entities (e.g., base stations, trusted nodes and certificate authorities) to establish symmetric or asymmetric keys between sensor nodes. The unique key-chain assigned to each node creates a binding between the identity of a node and its set of keys; thus, provides authentication which is limited by the resilience of the underlying key distribution scheme. A detailed comparative survey on wide range of key management schemes can be found in [1, 2].

We consider the problem of how to maximize the number of secure links in a wireless sensor network in order to increase its secure connectivity after deployment. In most deployment schemes, sensor nodes are randomly scattered over a large application area which might be inaccessible or infeasible to access after the deployment. Even with the controlled placement of sensor nodes, due to environmental challenges and deployment errors, the post-deployment network configuration might be unknown a priori. After the deployment, each node discovers its neighbors and tries to find a key to secure its wireless links in key discovery phase. Key management schemes are mostly blind to after deployment properties [1, 2]; therefore, many physical links may be left unprotected (i.e., without a key on them) which may result in a suboptimal secure routing, or even worse: secured links may not induce a *connected* network. What is needed is to optimally increase the secure connectivity after deployment (Figure 1). In the key establishment phase, each pair of neighboring nodes, which do not have common keys, establish one or more keys. Key establishment between two nodes can be achieved by exchanging messages directly over their insecure wireless link or over one or more secure paths on which each link is secured with a symmetric key as illustrated in Figure 1. Focus of this work is to understand complexity of the key establishment process in distributed wireless sensor networks subject to the physical layer properties such as connectivity, energy consumption and interference. In the broader sense, we would like to understand feasibility of existing key management schemes which trust on post deployment key establishment processes for secure connectivity.

Utilization of multi-hop wireless networks is investigated as the wireless scheduling problem which assigns transmission power levels to the network nodes and tries to schedule all the links in an arbitrary network topology. Scheduling complexity of arbitrary topologies in wireless networks in the context of physical Signal-to-Interference-plus-Noise-Ratio (SINR) has been investigated in [3–5] and shown to be NP-complete in various formulations. Secure capacity of a randomly deployed network is analyzed in [6] where each node receives a key-chain due to the *random key pre-distribution scheme* [7]. In [8], a framework is proposed to improve existing key pre-distribution schemes by assuming that sensors are deployed in groups and group members are located close to each other after deployment. Hence, more research is required on analyzing the complexity of increasing the secure connectivity and secure capacity in wireless ad hoc networks.

**Contribution:** Our contribution is theoretical as we formulate the different variants of the problem and analyze their complexity. In particular, we present two approaches: (i) establish new symmetric keys for the existing physical links (problem **P1**), and (ii) establish new physical links by increasing transmission power to connect the nodes that they do share a key (problem **P2**). Both of the problems are variants of the graph augmentation problem which are in general NP-hard for fixed cost functions and accept polynomial time constant approximation schemes (PTAS) [9].

Problem **P1** is a variant of the optimal graph (edge) augmentation problem on *key* graph $G_K$ (Figure 1). However, instead of a fixed cost assignment, it defines a nonlinear cost function on the links since the order of augmentation changes the cost assignment. In problem **P2**, new physical links can be created by increasing the power levels to reach a node with a shared secret key.

Although this problem can also be formulated as an optimal graph augmentation problem on the *physical* graph $G_P$ (Figure 1), it has two main differences. First, increasing power levels induce interference on the nodes and may have an adverse effect on the overall network capacity. Thus, there are interference constraints on the nodes in **P2** to ensure an acceptable signal to interference plus noise ratio (SINR). Second, the cost of each link has two parameters: (i) energy cost for establishing this link, and (ii) amount of interference this link induces on the other nodes. We prove that neither **P1** nor **P2** accepts PTAS.
**Organization:** Rest of the paper is organized as follows: in Section 2, we describe the network model and basic notations. We break the problem of optimally increasing secure connectivity into three optimization problems. In Section 3, we formulate the first problem **P1** as an instance of edge augmentation problem on the key graph. In Section 4, we formulate the second problem **P2** as a constrained optimization problem with interference constraints on the physical graph. Finally, in Section 5, we conclude.

## 2 Network Model and Problem Definition

### 2.1 Network Model and Notations

We model a wireless sensor network as a set of nodes $WN = \{\, n_1, n_2, \ldots, n_N\}$ distributed over an Euclidean plane. The Euclidean distance between two nodes $n_s$ (sender) and $n_r$ (receiver, $1 \leq s, r \leq N$) is represented by $d(n_s, n_r)$. In this work, we assume that each node $n_s$ has discrete power levels $(1, 2, 3, \ldots, l^i_{max}$ where $1 \leq i \leq N)$. Each node may have different maximum power level $l_{max}$ due to its battery condition. By changing their power levels ($P^l_s$: node $n_s$ transmitting at power level $l$), nodes can control the received signal strength $\frac{P^l_s}{d(n_s,n_r)^\alpha}$ ($\alpha$ is a constant that depends on the medium) on the intended recipient $r$. We use the Signal-to-Interference-plus-Noise-Ratio (SINR) model because the graph-theoretic modeling of interference ignores the fact that interference coming from different transmitters accumulate and can not be limited to a specific border. SINR model considers that a message is successfully received by a receiver if the ratio between received signal strength and noise plus interference from other nodes exceeds a threshold $\beta$ (Equation 1) which is defined by the hardware.

$$\frac{\frac{P^l_s}{d(n_s,n_r)^\alpha}}{Noise + \sum_{n_k \in WN \setminus n_s} \frac{P^l_k}{d(n_k,n_r)^\alpha}} \geq \beta \tag{1}$$

Wireless networks are generally represented with undirected graphs where the uniform transmission range and symmetric links are assumed. **Physical Graph** $G_P = (V, E_P)$ represents a network where each node is represented with a vertex, and there is an edge between two vertices if the corresponding nodes are within each others transmission range. For the same vertex set $V$, **Key Graph** $G_K = (V, E_K)$ represents the key connectivity where there is an edge in between two vertices if the corresponding nodes share or can establish one or more symmetric keys to secure their communication. In **Secure Graph** $G_S = (V, E_S)$, there is an edge in between two vertices if they have an edge both in $G_P$ and $G_K$. In other words, $E_S = E_P \bigcap E_K$ as illustrated in Figure 1.

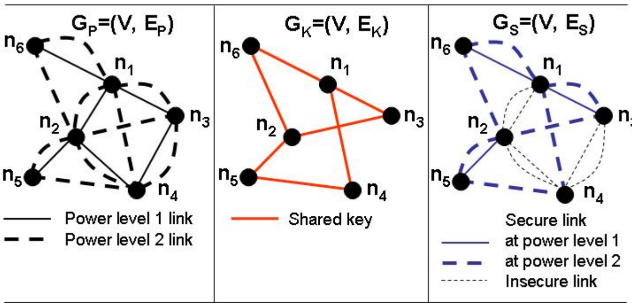


**Fig. 1.** Physical graph $G_P = (V, E_P)$, Key graph $G_K = (V, E_K)$ and Secure graph $G_S = (V, E_S)$ where $E_S = E_P \bigcap E_K$

**Notations:** Nodes which are within each other's radio range are called **neighboring nodes**. A wireless link between two neighboring nodes is called a **physical link**. A physical link between two neighboring nodes that share a key is called a **secure link**. If the nodes don't share a key, then it is an **insecure link**. A **secure path** is a path on which each physical link is a secure link. A **key path** is a secure path which is used to exchange a shared-key (i.e. with a mechanism similar to Diffie-Hellman). Table 1 lists notations used throughout this paper.

**Table 1.** Abbreviations

| | | | |
|---|---|---|---|
| $WN$ | Network with N nodes $\{n_1, \ldots, n_N\}$ | $\mathcal{F}$ | Set of flows $(s, t)$ |
| $T$, | Set of transmitters $(n_{i,l})$ | $R$ | Set of receivers $(n_i)$ |
| $T(i)$ | Transmitters of node $n_i$ | $R(j)$ | Receiver of the transmitter $j \in T$ |
| $P_i^l$ | Transmission power of $n_{i,l}$ at level $l$ | $f^{s,t}$ | Flow $(s, t)$, $f^{s,t} \in \{0, 1\}$ |
| $f_{i,j}^{s,t}$ | Flow on edge $(i, j)$ due to flow $f^{s,t}$ | $n_{i,l}$ | $i^{th}$ node transmitting at level $l$ |
| $l_{max}^i$ | Maximum power level of node $n_i$ | $K_{i,j}$ | Shared key between nodes $n_i$ and $n_j$ |
| $KC_i$ | Key-chain of node $n_i$ | $E^R$ | Receive cost of a unit flow |
| $E^T$ | Transmission cost of a unit flow | $G_P$ | Physical graph $G_P(V, E_P)$ |
| $G_K$ | Key graph $G_K(V, E_K)$ | $G_S$ | Secure graph $G_S(V, E_S)$ |
| $G_A$ | Auxiliary graph $G_A(V_A, E_A)$ | | |

## 2.2 Problem Definition

Upon deployment of a wireless sensor network, the induced secure graph may be under-utilized because, although $G_S = G_K \cap G_P$ is connected many physical links may not be secured by a key resulting in inefficient routing as shown in Figure 2-C. It may be even disconnected as depicted in Figure 2-E.

In this paper we consider the problem of optimally increasing secure connectivity either by establishing new keys using the secure paths (we rule out executing Diffie-Hellman (DH) or similar techniques over an *insecure* wireless link due to lack of authentication that makes man-in-the-middle attacks possible), or by adding new physical links (i.e., increasing the transmission power) between nodes that share a key. We consider two optimization problems:

- **P1** ($G_K \to G_S$): Find order of shared key establishment for unsecured physical links. Find optimal secure paths to establish shared keys (Approach: graph augmentation on $G_K$).
- **P2** ($G_P \to G_S$): Find optimal set of new physical links to be established between the nodes with shared keys (Approach: graph augmentation on $G_P$).

## 3 Problem P1: Augmenting the Key Graph $G_K$

Problem **P1** assumes both key graph $G_K$ and physical graph $G_P$ are connected and it adds edges to $G_K$ to increase key connectivity of an *under-utilized wireless sensor network* to obtain $\kappa - connected$ secure graph where $\kappa \geq 2$

In problem **P1**, adding an edge between the nodes $n_i$ and $n_j$ in $G_K$ means establishing keys between nodes $n_i$ and $n_j$ through a secure path by using Diffie-Hellman (DH) or similar key establishment algorithms. Recall that DH itself does not provide authentication, thus it should be applied through a secure path where each pair of neighboring nodes on the path shares a key.

Consider Figure 2-(A,B) as an example where secure graph is connected. Although each node pairs $(n_1, n_2)$, $(n_3, n_4)$ and $(n_2, n_7)$ has a physical link, they do not share a key to secure their links. These node pairs have to communicate through secure paths, rather than using their direct link, yielding an under-utilized network. In this problem, our challenge is three-fold. First, a pair of nodes should be identified to establish a key between them. Second, a minimum cost (e.g., shortest hop count) secure path for each node pair should be found through which DH key establishment can be executed. Third, the order in which DH key establishment is executed should be identified. In the secure graph of Figure 2-(A,B), establishing a key first for $(n_1, n_2)$ results in a shorter secure path for the nodes $(n_3, n_4)$.

Problem **P1** is a variant of graph augmentation problem on the keying graph $G_K$. Given a graph $G = (V, E)$ with $n$ nodes and $m$ edges where each edge $(u, v)$ has an arbitrary non-negative weight $c_{(u,v)}$, let $G' = (V, E')$ be its subgraph where $E' \subseteq E$. The edge augmentation problem is to find minimum-weight set of edges from the edge set $E \setminus E'$ whose addition makes $G'$ $\kappa - edge - connected.$ The node connectivity augmentation version is slightly different. Given a graph $G = (V, E)$ and a set of vertices $V' \subseteq V$, problem is to find a set of edges with minimum-weight whose addition provides connectivity between every pair of vertices in $V'$.

The augmentation problem is NP-Hard when $\kappa - edge$ or $\kappa - vertex$ disjoint paths are required between every pair of nodes in $V'$ for $\kappa \geq 2$. However, for fixed

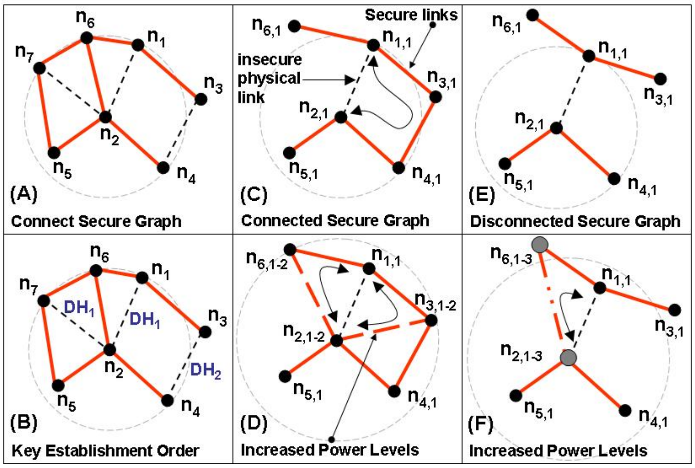


**Fig. 2.** (A) Under-utilized secure graph $G_S = (V, E_S)$. (B) Order of Diffie-Hellman key establishment for the minimized cost (e.g., establishing key for $(n_1, n_2)$ first results in shorter secure path for $(n_3, n_4)$). (C) Secure graph is connected. Nodes $n_{1,1}$ and $n_{2,1}$ have a physical link but don't share a key. They can communicate through a secure path of 3 hops to establish a key. (D) Nodes $n_2$, $n_3$ or $n_6$ can establish new secure links at the power level 2 to provide shorter secure paths for nodes $n_1$ and $n_2$. (E) Secure graph is disconnected. Nodes $n_1$ and $n_2$ have a link but they do not share a key, and they can not find a secure path to establish key. (F) Nodes $n_2$ and $n_6$ share a key, and they can establish a new link at the power level 3 to provide the secure connectivity.

cost assignment on the edges it has an approximation (PTAS) which achieves a factor of 2 for $\kappa = 2$ [9]. There is a rich literature of previous work for such tractable variants of **P1** that offer both deterministic [10, 11] and randomized [12] approaches.

However, the cost function to be minimized in **P1** is different from classical graph augmentation since the cost of each edge-to-be-inserted (call this a new-edge) to $G_S$ may change as the new edges are added to $G_K$. For example, suppose the cost or weight of a new-edge $(i, j)$ is the length of the shortest path between $i$ and $j$ in $G_S$, then this cost will change depending on the order of insertion. This dependency presents a non-linear cost function on the links and makes the order of augmentation important. Thus, optimality depends upon the ordering of the set of node pairs ($E_W \subseteq E_P \setminus E_K$) as illustrated in Figure 2-(A,B). This problem is not only NP-Hard but also it does not admit a PTAS since minimum cutsets to be augmented do not admit constant costs.

## 4 Problem P2: Augmenting the Physical Graph $G_P$

In this problem, we consider adjusting power levels to create a (new) physical link between a pair of nodes that share a symmetric key. The optimization problem here is to determine which nodes should increase their power levels to provide the secure connectivity at a minimum cost (Figures 2-E,F). Increasing power levels decreases the number of hops in a secure path as illustrated in Figures 2-(C,D). However, increasing transmission power generates more interference on surrounding nodes. Enforcing a bound on *instantaneous interference* to ensure acceptable SINR for wireless communications yields to a mixed integer non-linear optimization problem [13]. Thus, problem **P2** has two parts: (i) identification of optimal number of edges to augment $G_P$, and (ii) interference constrained power selection for materializing these edges. We use an *auxiliary graph representation* similar to [13] for representing the power levels and formulating the interference constraints.

We note that problem **P2** can be formulated also as an instance of the *edge augmentation* problem. However, there are two complications: (i) interference constraints on the nodes, and (ii) a complex cost function on the edge set that must capture not only the energy cost but also the interference induced on the other nodes. Thus, **P2** is optimal augmentation of $G_P$ subject to interference constraints with a nontrivial cost function.

We formulate *edge augmentation* problem with the interference constraints on nodes and transmission costs on edges as a flow problem using an auxiliary graph $G_A = (V_A, E_A)$ similar to [13].

## 4.1 Auxiliary Graph Representation

In this representation, for each node $n_i$, auxiliary $G_A$ includes a receiver vertex $n_i$ and $l^i_{max}$ transmitter vertices $(n_{i,1}, n_{i,2}, \ldots, n_{i,l^i_{max}})$ corresponding to the each discrete power level. Receivers from all nodes form the receiver set $R = \{n_1, n_2, \ldots, n_i\}$, and transmitters form the transmitter set $T = \{\, n_{1,1}, \ldots, n_{1,l^1_{max}}, n_{2,1}, \ldots, n_{2,l^2_{max}}, \ \ldots, \ n_{i,1}, \ldots, n_{i,l^i_{max}}\}$ where $V_A = R \bigcup T$. $T(i)$ represents all transmitters $\{n_{i,1}, n_{i,2}, \ldots, n_{i,l^i_{max}}\}$ of the receiver $n_i$, and $R(j)$ represents receiver $n_j$ of the transmitter $n_{j,l}$. Edge set $E_A$ includes edges $(i, j)$ of types: (1) $i \in R$ *and* $j \in T(i)$, and (2) $i \in T$ *and* $j \in R$ where there is a shared-key between nodes $n_i$ and $n_j$ (i.e. $(i, j) \in E_K$). First rule states that there are edges from the receiver of each node to all of its transmitters (dashed edges in Figure 3-A). Second rule states that there is an edge from each transmitter to each receiver located within the transmission range required that both nodes share a key (solid edges in Figure 3-A). These edges have cost associated with them as the amount of energy consumed to transfer one unit of flow. For simplicity, all edges considered to have unlimited capacities but the network is capacitated due to interference. There is a limit on the amount of interference a receiver can handle meaning that not all transmitters can transmit at the same time.

We force a limit on the amount of interference-plus-noise that a node can tolerate as the *Reception Quality* constraint. This constraint requires that a message is received by a receiver if the ratio between the received signal strength and the interference-plus-noise due to surrounding transmitters do not exceed a threshold as specified in Equation 1. Then, our optimization problem becomes finding a minimum cost set of edges on the auxiliary graph subject to the interference

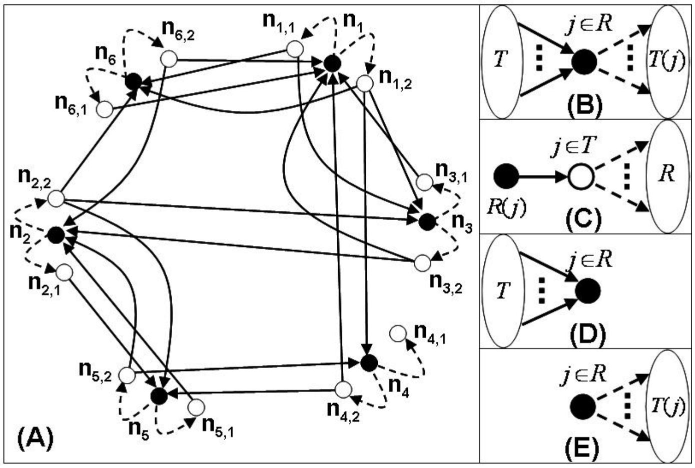


**Fig. 3.** (A) Auxiliary graph $G_A = (V_A, E_A)$ corresponding to the secure graph $G_S = (V, E_S)$ of Figure 1. Black vertices are receivers $R = \{n_1, n_2, n_3, n_4, n_5, n_6\}$. Each node has two transmit power levels which are the white transmitter vertices $T = \{n_{1,1}, n_{1,2}, n_{2,1}, n_{2,2}, \ldots, n_{6,1}, n_{6,2}\}$. Each solid edge has a cost associated which is the total energy used by the system to pass one unit of flow and/or the energy consumption due to interference created on the surrounding receivers. Dashed edges have no cost. All edges have unlimited capacities but the network is capacitated due to the interference because there is a limit on the amount of interference a receiver can handle due to SINR model. (B) Receiver flow conservation for Equation 2, (C) Transmitter flow conservation for Equation 3, (D) Receiver utilization for Equation 5, and (E) Transmitter utilization for Equation 6.

constraint where cost of an edge is $E = E^T + E^R$ so that resulting secure graph is $\kappa -$ *connected.*

The optimization problem **P2** has bi-objectives: (1) maximizing the number of concurrent flows -this is the augmentation part, and (2) minimizing the cost which is defined w.r.t. power consumption (since we handle the interference in constraints). Thus, we break the problem into two subproblems and formulate two *integer programs*. In *maximum key establishment flow* problem **P2.1**, we seek for the maximum amount of flow $\mathcal{F}_{Max} \subseteq \mathcal{F}$ that we can grant subject to interference constraints. In *minimum cost key establishment flow* problem **P2.2**, we seek for minimum cost flow assignment on the auxiliary graph edges while keeping $|\mathcal{F}_{Max}|$ and the interference as constraints.

Having formulated the problem as an auxiliary graph, it can be shown that both problems **P2.1** and **P2.2** are $NP-Hard$ and $MAX-SNP-Hard$ based on a reduction from MAX3SAT (see the appendix for formal proofs). Thus, they are intractable and it is NP-Hard to approximate them within a factor $1+\epsilon$ for some fixed $\epsilon > 0$.

## 4.2 Problem P2.1: Mathematical Programming Formulation

We formulate **P2.1** as a constrained optimization problem. The optimization problem aims to maximize the number of source-destination pairs $(s, t) \in \mathcal{F}$ be granted on the auxiliary graph concurrently subject to interference thresholds on each vertex.

**Definition 1 (MaxKeyEstabFlow Problem P2.1).** *Given $G_A = (V_A, E_A)$ the auxiliary graph representation of a deployment, euclidian distances $d(n_i, n_j)$ between nodes for all node pairs $(n_i, n_j)$, SINR constants $\beta$ and $\alpha$, power levels $(1, 2, 3, \ldots, l^i_{max})$ for all nodes $n_i$ and set of flows $\mathcal{F}$ for the key establishment traffic, **P2.1** is the problem of maximizing the number $\mathcal{X}$ ($\mathcal{X} = |\mathcal{F}'|$ where $\mathcal{F}' \subseteq \mathcal{F}$) of source-destination pairs that can exchange key establishment messages concurrently on the auxiliary graph $G_A$ subject to interference constraints. Solution to the problem is the subset $\mathcal{F}'$ of source-destination pairs, and flows of source-destination pairs $(s, t) \in \mathcal{F}'$ assigned to a subset of edges $E'_A \subseteq E_A$.*

Problem is similar to *integer multiflow* optimization problem [14] because flows belonging to multiple source-destination pairs $(s, t) \in \mathcal{F}$ is assigned to edges of the auxiliary graph. Vertices of the edges having non-zero flow in the auxiliary graph will correspond to power level of the corresponding pairwise communication.

Let $G_A$ be the auxiliary graph corresponding to a deployment with N nodes. Also, $\mathcal{F}$ is the set of node pairs $(s, t)$ representing neighboring nodes which don't share a key, and which need to exchange key establishment messages. We assume that the key establishment is done by exchanging two units of messages between $s$ and $t$, thus the demand for $(s, t)$ and $(t, s)$ are both one. Then, the problem is to find largest routable subset of $\mathcal{F}$ in $G_A$ subject to: (i) flow conservation (receiver and transmitter), (ii) flow symmetry, (iii) utilization (receiver and transmitter), and (iv) reception quality:

**(i.a) Receiver flow conservation** constraint requires that the difference between flows coming and leaving a receiver (as in Figure 3-B) due to a flow between $(s, t)$ should be: (i) zero if the node is not the source or the destination, (ii) $f^{s,t} \in \{0, 1\}$ if the node is destination, and (iii) $(-f^{s,t}) \in \{-1, 0\}$ if the node is source. Thus, for each $j \in R$ and $\forall (s, t) \in \mathcal{F}$:

$$\sum_{i \in T} f^{s,t}_{i,j} - \sum_{i \in T(j)} f^{s,t}_{j,i} = x \quad s.\ t. \quad \begin{cases} x = f^{s,t}, & \text{j=t;} \\ x = -f^{s,t}, & \text{j=s;} \\ x = 0, & o/w. \end{cases} \tag{2}$$

**(i.b) Transmitter flow conservation** constraint requires that all flows coming and leaving a transmitter (as in Figure 3-C) due to a flow between $(s, t)$ should be equivalent. Thus, for each $j \in T$ and $\forall (s, t) \in \mathcal{F}$:

$$\sum_{i \in R(j)} f^{s,t}_{i,j} - \sum_{i \in R} f^{s,t}_{j,i} = 0. \tag{3}$$

**(ii) Flow symmetry** constraint requires that when there is a flow on link $(n_{i,l}, n_j)$ $(1 \leq l \leq l^i_{max})$ due to the flow between $(s, t) \in \mathcal{F}$, there should be a flow on link $(n_{j,l'}, n_i)$ $(1 \leq l' \leq l^j_{max})$ due to the flow between $(t, s) \in \mathcal{F}$. In other words, key exchange request and response messages between two nodes use the same path in the secure graph. This assumption helps in that whenever one of the transmitters $n_{i,l}$ or $n_{j,l'}$ can not be activated due to the interference, the other one should not be. Thus, for each node pair $n_i$ and $n_j$, and $\forall(s, t) \in \mathcal{F}$:

$$\sum_{l=1}^{l^i_{max}} f^{s,t}_{n_{i,l},n_j} - \sum_{l'=1}^{l^j_{max}} f^{t,s}_{n_{j,l'},n_i} = 0. \tag{4}$$

**(iii.a) Receiver utilization** constraint requires that the receiver utilization (as in Figure 3-D) due to a flow should not exceed the unity. Thus, for each $j \in R$ and $\forall(s, t) \in \mathcal{F}$:

$$\sum_{i \in T} f^{s,t}_{i,j} \in \{0, 1\}. \tag{5}$$

**(iii.b) Transmitter utilization** constraint requires that the transmitter utilization (as in Figure 3-E) due to a flow should not exceed unity. Thus, for each $j \in R$ and $\forall(s, t) \in \mathcal{F}$:

$$\sum_{i \in T(j)} f^{s,t}_{j,i} \in \{0, 1\}. \tag{6}$$

**(iv) Reception Quality** constraint states that flow $f^{s,t}_{i,j}$ (flow on edge $(i, j)$ due to flow $f^{s,t}$) exists (non-zero) if the ratio between the received signal strength and the interference-plus-noise, due to surrounding transmitters, do not exceed a threshold as specified in Equation 1. This threshold is applicable to a receiver if there exists a flow on this receiver. Thus, given $\delta^k$ and $f^{s,t}_{i,j}$ which are the indicator of a flow on each transmitter $k$ and on the receiver $j$ respectively:

$$\forall k \in T,\ \delta^k = \begin{cases} 1, \sum_{(s,t)\in\mathcal{F}} \sum_{m \in R} f^{s,t}_{k,m} > 0; \\ 0, \text{o/w}. \end{cases}$$

For each $j \in R$:

$$\frac{\frac{P^l_i}{d(n_i,n_j)^\alpha}}{Noise + \sum_{k \in T \setminus \{i\}} \frac{P^l_k \times \delta^k}{d(n_k,n_j)^\alpha}} \geq \beta \times f^{s,t}_{i,j} \tag{7}$$

Our mathematical program then becomes:

$$Maximize \quad \mathcal{X} = \sum_{(s,t)\in\mathcal{F}} f^{s,t} \quad Subject\ to\ (2), (3), (4), (5), (6), (7).$$

**Proof sketch:** We prove that MaxKeyEstabFlow is NP-hard by using a reduction from MAX3SAT problem, which is a truth assignment to the variables, to find maximum number of clauses that can be satisfied in a boolean formula in the $3CNF$ form. We define a reduction from MAX3SAT to MaxKeyEstabFlow in two steps. First, given a boolean formula in the $3CNF$ form with $n$ variables and $m$

clauses, we create a WSN deployment in an Euclidian plane. We create sensor nodes $C_i$ and $D_i$ for $i^{th}$ clause, and sensor nodes $x_j$ and $\overline{x}_j$ for $j^{th}$ variable where only the sensor nodes $x_j$ and $\overline{x}_j$ create interference on each other (a.k.a. both variables can not be set as TRUE). We define set of flows $\mathcal{F} = \{(C_i, D_i), (D_i, C_i) | 1 \leq i \leq m\}$. Second, using this WSN deployment we create an auxiliary graph representation as described in Section 4.1. Thus, the objective of finding a truth assignment to the variables so that maximum number of clauses that can be satisfied becomes finding maximum number of source-destination pairs in $\mathcal{F}$ which can be granted concurrently both on the WSN and on the auxiliary graph $G_A$ subject to interference constraints. Inapproximability results for **P2.1** comes from the interference created by the links and the interference threshold constraint. We show that for every $\epsilon > 0$, there is a gap preserving reduction from the MAX3SAT to MaxKeyEstabFlow that has parameters (c, 1+$\epsilon$, c$|\mathcal{F}|/2$, 1+$\epsilon$) where $\mathcal{F}$ is the set of flows. We show that the MAX3SAT ($\varphi$) = c $\Leftrightarrow$ MaxKeyEstabFlow ($\tau(\varphi)$) = c.m. (see the appendix for formal proofs).

## 4.3 Problem P2.2 Mathematical Programming Formulation

**Definition 2 (MinCostKeyEstabFlow Problem P2.2).** *Given the auxiliary graph representation $G_A = (V_A, E_A)$ of a deployment, euclidian distances $d(n_i, n_j)$ between nodes for all node pairs $(n_i, n_j)$, SINR constants $\beta$ and $\alpha$, power levels $(1, 2, 3, \ldots, l^i_{max})$ for all nodes $n_i$, set of flows $\mathcal{F}$ for the key establishment traffic and the maximum number $\mathcal{X}$ of concurrent key establishment flow, it is the problem of finding at least $\mathcal{X}$source-destination pairs which can exchange key establishment messages on the auxiliary graph $G_A$ at a minimum cost subject to interference constraints. Solution to the problem is the subset $\mathcal{F}'$ of source-destination pairs, flows of source-destination pairs $(s,t) \in \mathcal{F}'$ assigned to a subset of edges $E'_A \subseteq E_A$ and the overall cost.*

Our objective is to grant at least $\mathcal{X}$ flows through the auxiliary graph $G_A$ with a minimum cost. Result of the program is the flow assigned to each link on the auxiliary graph $G_A$. This result will also imply the power level assignment to each sensor node so to grant at least $\mathcal{X}$ flows between source-destination pairs. Our formulation has the same constraints as the maximization problem: (i) flow conservation (receiver and transmitter), (ii) flow symmetry, (iii) utilization (receiver and transmitter), and (iv) reception quality. *Flow bound* is additional constraint which requires total flow granted by the flow assignment should be at least $\mathcal{X}$. Thus:

$$\sum_{(s,t)\in\mathcal{F}} f^{s,t} \geq \mathcal{X}. \tag{8}$$

Our mathematical program becomes:

$$Minimize \quad \sum_{(s,t)\in\mathcal{F}} \sum_{i\in T,\ j\in R} f^{s,t}_{i,j}\ C_{i,j} \quad Subject\ to\ (2),(3),(4),(5),(6),(7),(8).$$

$C_{i,j} = E^T + E^R$ is the energy cost of a unit flow on the edge $(i, j)$ where $i \in T$ and $j \in R$. All other edges have zero costs.

**Proof Sketch:** We prove that MinCostKeyEstabFlow is NP-hard by using a reduction from the *Weighted MAX3SAT* problem where each clause has a weight, and the problem is to maximize the sum of the weights of satisfied clauses. The *Weighted MAX3SAT* is both NP-hard and MAX-SNP [15] problem. We use similar approach as in MaxKeyEstabFlow to show that MinCostKeyEstabFlow problem is both NP-hard and MAX-SNP (see the appendix for formal proofs).

## 5 Conclusion and Discussions

Focus of this work is first to formulate the key establishment problem in wireless sensor networks together with the physical layer properties, then to analyze its complexity. We present mathematical programming formulations *maximum key establishment flow* and *minimum cost key establishment flow* as variants of graph augmentation problems. We prove that finding optimum solutions and finding polynomial time approximations are both NP-hard. We place these problems in inapproximability Class I [9] which is the richest class of all. Our results show that post-deployment key establishment in distributed wireless sensor networks is a hard problem. Most key management schemes trusting on post deployment key establishment for secure connectivity may not be feasible and applicable to practical solutions. Research should focus on making efficient use of deployment knowledge, or on developing deterministic key management schemes (such as [16–18]) which can ensure that any pair of nodes secure their communication using symmetric or asymmetric keys without explicit key establishment flows.

## References

1. Zhang, J., Varadharajan, V.: Wireles ssensor network key management survey and taxonomy. J. Netw. and Com. App. 33 (2010)
2. Camtepe, S.A., Yener, B.: Key Management. In: Wireless Sensor Network Security. Cryptology and Information Security Series. IOS Press (2008)
3. Santi, P., Maheshwari, R., Resta, G., Das, S., Blough, D.M.: Wireless link scheduling under a graded sinr interference model. In: ACM FOWANC (2009)
4. Goussevskaia, O., Oswald, Y.A., Wattenhofer, R.: Complexity in geometric sinr. In: ACM MobiHoc (2007)
5. Moscibroda, T., Wattenhofer, R., Zollinger, A.: Topology control meets sinr: the scheduling complexity of arbitrary topologies. In: ACM MobiHoc (2006)
6. Bhandari, V., Vaidya, N.H.: Secure capacity of multi-hop wireless networks with random key pre-distribution. In: IEEE MCN (2008)
7. Eschenauer, L., Gligor, V.D.: A key-management scheme for distributed sensor networks. In: ACM CCS (2002)
8. Liu, D., Ning, P., Du, W.: Group-based key predistribution for wireless sensor networks. ACM TOSN 4(2) (2008)
9. Hochbaum, D.S.: Approximation Algorithms for NP-Hard Problems. PWS Publishing Company (1997)

10. Naor, D., Gusfield, D., Martel, C.: A fast algorithm for optimally increasing the edge connectivity. SIAM J. of Comp. 26(4) (1997)
11. Nagamochi, H., Ibaraki, T.: Augmenting edge-connectivity over the entire range in o(nm) time. J. Alg. 30(2) (1999)
12. Benczúr, A.A., Karger, D.R.: Augmenting undirected edge connectivity in $(n^2)$ time. In: ACM-SIAM SODA (1998)
13. Savas, O., Alanyali, M., Yener, B.: Joint route and power assignment in asynchronous multi-hop wireless networks. In: MedHocNet (2004)
14. Costa, M.C., Létocart, L., Roupin, F.: Minimal multicut and maximal integer multiflow: a survey. Elsevier J. of Op. Res. 162 (2005)
15. Papadimitriou, C.H., Yannakakis, M.: Optimization, approximation, and complexity classes. J. Comp. and Sys. Sci. 43(3) (1991)
16. Camtepe, S.A., Yener, B.: Combinatorial design of key distribution mechanisms for wireless sensor networks. IEEE/ACM TON 15(2) (2007)
17. Blom, R.: An optimal class of symmetric key generation systems. In: Beth, T., Cot, N., Ingemarsson, I. (eds.) EUROCRYPT 1984. LNCS, vol. 209, pp. 335–338. Springer, Heidelberg (1985)
18. Blundo, C., De Santis, A., Herzberg, A., Kutten, S., Vaccaro, U., Yung, M.: Perfectly-secure key distribution for dynamic conferences. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 471–486. Springer, Heidelberg (1993)

## A Proofs

*Proof.* (P2.1 MaxKeyEstabFlow is in NP-hard) We prove that MaxKeyEstabFlow is NP-Hard using a reduction from the MAX3SAT problem which is a truth assignment to the variables $\{x_1, x_2, \ldots, x_n\}$ to find maximum number of clauses that can be satisfied in a boolean formula $\varphi$ in the $3CNF$ form with clauses $\{C_1, C_2, \ldots, C_m\}$. We define a reduction $\tau$ from MAX3SAT to MaxKeyEstabFlow in two steps. The first step reduces a MAX3SAT problem instance into a WSN problem instance, and the second step derives an auxiliary graph formulation.
*Step 1:* Given a boolean formula $\varphi$ in the $3CNF$ form with $n$ variables and $m$ clauses, create a WSN deployment in an Euclidian plane (for $1 \leq i \leq m$ and $1 \leq j \leq n$):

1. Create sets of sensor nodes: $C = \{C_i | 1 \leq i \leq m\}$, $D = \{D_i | 1 \leq i \leq m\}$, $X = \{x_j | 1 \leq j \leq n\}$ and $\overline{X} = \{\overline{x_j} | 1 \leq j \leq n\}$. Namely, create sensor nodes $C_i$ and $D_i$ for $i^{th}$ clause, and sensor nodes $x_j$ and $\overline{x}_j$ for $j^{th}$ variable.
2. Sensor nodes $x_j$ and $\overline{x}_j$ have a maximum power level of $l^j_{max} = 1$. Sensor nodes $C_i$ and $D_i$ have a maximum power level of $l^i_{max} = L_{max}$ which covers whole WSN and can use the RTS/CTS signalling to check availability of the channel at receivers.
3. Only the sensor nodes $x_j$ and $\overline{x}_j$ create interference on each other (a.k.a. boolean variables $x_j$ and $\overline{x}_j$ can not be *true* at the same time).
4. Distribute a key-chain $KC$ to each sensor node. $KC_{C_i}$ and $KC_{x_j}$ (a.k.a., $KC_{\overline{x}_j}$) should share a key if variable $x_j$ (a.k.a., $\overline{x}_j$) appears in $i^{th}$ clause. Similarly, $KC_{D_i}$ and $KC_{x_j}$ (a.k.a., $KC_{\overline{x}_j}$) should share a key if variable $x_j$ (a.k.a., $\overline{x}_j$) appears in $i^{th}$ clause. All other pairs of key-chains should not share a key.

5. Create set of flows $\mathcal{F} = \{(C_i, D_i), (D_i, C_i) | 1 \leq i \leq m\}$. These are the pairs of nodes which have physical links but do not share keys to secure their communication.
6. Place the sensor nodes on a unit disk area: (a) Draw $v \times v$ ($v = \lceil\sqrt{n}\rceil$) grid for $n$ variables. (b) Each grid location should be a square of size $2\alpha I \times 2\alpha I$ where $I$ is the distance below which SINR on receiving node due to other nodes is less than the threshold based on Formula 1. $\alpha I$ (for a constant $\alpha$) is the distance over which interference is negligible. (c) For each sensor node $x_j$, select a random empty grid coordinate and locate the node at the center of the grid location. (d) Place each sensor node $\overline{x}_j$ at a random location where Euclidian distance between $d(x_j, \overline{x}_j) < I$. Thus, SINR on $x_j$ as receiver can be less than the threshold only due to $\overline{x}_j$.

*Step 2:* Given a WSN deployment which is reduced from a boolean formula $\varphi$ in the $3CNF$ form with $n$ variables and $m$ clauses, develop an auxiliary graph formulation as described in Section 4.1 and as illustrated in Figure 4:

1. Create auxiliary graph $G_A = (V_A, E_A)$ (for $1 \leq i \leq m$, $1 \leq j \leq n$ and $1 \leq g \leq L_{max}$):
   (a) Receiver nodes are $R = C \bigcup D \bigcup X \bigcup \overline{X}$.
   (b) Add transmitter nodes $C_i^{T_g}$, $D_i^{T_g}$, $x_j^T$ and $\overline{x}_j^T$.
   (c) Add directed edges ($C_i^R$, $C_i^{T_g}$) and ($D_i^R$, $D_i^{T_g}$), ($x_j^R$, $x_j^T$) and ($\overline{x}_j^R$, $\overline{x}_j^T$).
   (d) Add directed edges ($C_i^{T_g}$, $x_j^R$) (a.k.a., $\overline{x}_j^R$) and ($x_j^T$, $C_i^R$) (a.k.a, $\overline{x}_j^T$) if $x_j$ (a.k.a, $\overline{x}_j$) shares a key with $C_i$.
   (e) Add directed edges ($D_i^{T_g}$, $x_j^R$) (a.k.a., $\overline{x}_j^R$) and ($x_j^T$, $D_i^R$) (a.k.a, $\overline{x}_j^T$) if $x_j$ (a.k.a, $\overline{x}_j$) shares a key with $D_i$.
2. Set edge capacities as unlimited.
3. Create set of flows $\mathcal{F} = \{(C_i, D_i), (D_i, C_i) | 1 \leq i \leq m\}$.

This algorithm transforms a boolean formula $\varphi$ in $3CNF$ form with $n$ variables and $m$ clauses first into a WSN deployment with $2(m + n)$ nodes, and then formulates it as an auxiliary graph $G_A$ with $(2m(L_{max} + 1) + 4n)$ nodes and $O(m+2n)$ edges where $|\mathcal{F}| = 2m$. Objective of finding a truth assignment to the variables so that number of clauses that can be satisfied is maximized becomes finding maximum number of flows in $\mathcal{F}$ which can be granted concurrently both on the WSN and on the auxiliary graph $G_A$ subject to interference constraints. Thus, the transformation from MAX3SAT to MaxKeyEstabFlow can be carried out in polynomial time.

A solution to the problem instance $\tau(\xi)$ of MaxKeyEstabFlow in auxiliary graph representation can be converted to a solution of problem instance $\xi$ of MAX3SAT in two easy steps in linear time. First, if total flow on the transmitter $x_j^T \geq 1$ (a.k.a. $\overline{x}_j^T \geq 1$) then set boolean variables $x_j = True$ (a.k.a. $\overline{x}_j = True$) and $\overline{x}_j = False$ (a.k.a. $x_j = False$) for $1 \leq j \leq n$. Note that the interference constraint does not permit both flows $x_j^T \geq 1$ and $\overline{x}_j^T \geq 1$. Second, if total flow on both transmitters are $x_j^T = 0$ and $\overline{x}_j^T = 0$, then set either ($x_j = True$ and $\overline{x}_j = False$) or ($x_j = False$ and $\overline{x}_j = True$) for $1 \leq j \leq n$. This assignment

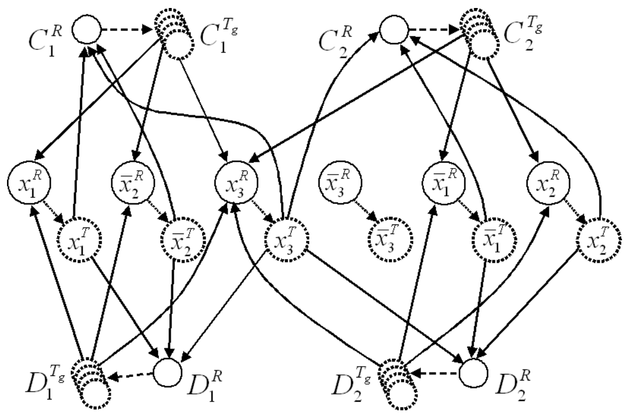


**Fig. 4.** Auxiliary graph $G_A = (V_A, E_A)$ reduced from sample boolean formula $\varphi = ((x_1 \vee \overline{x}_2 \vee x_3) \wedge (\overline{x}_1 \vee x_2 \vee x_3))$. There is only one transmit power level for the nodes corresponding to the boolean variables. Nodes $C_1$, $C_2$, $D_1$, $D_2$ have $L_{max}$ transmit power levels. Set of receivers are $R = \{x_1^R, x_2^R, x_3^R, \overline{x}_1^R, \overline{x}_2^R, \overline{x}_3^R, C_1^R, C_2^R, D_1^R, D_2^R\}$, and set of transmitters are $T = \{x_1^T, x_2^T, x_3^T, \overline{x}_1^T, \overline{x}_2^T, \overline{x}_3^T, C_1^{T_g}, C_2^{T_g}, D_1^{T_g}, D_2^{T_g}\}$ for $1 \leq g \leq L_{max}$ where $V_A = R \bigcup T$. All edges have unlimited capacities. Finally set of flow is $\mathcal{F} = \{(C_1, D_1), (C_2, D_2), (D_1, C_1), (D_2, C_2)\}$.

does not change the number of the satisfied clauses in $\xi$, but some satisfied clauses may have more than one variable set to $True$. Very similar steps apply for converting the solution to the problem instance $\tau(\xi)$ of MaxKeyEstabFlow in WSN to solution to the problem instance $\xi$ of MAX3SAT in linear time. The flows on sensor nodes $x_j$ and $\overline{x}_j$ should be considered instead of the flows on transmitters $x_j^T$ and $\overline{x}_j^T$.

Optimal solution to the instance $\xi$ of MAX3SAT has $c$ satisfied clauses if and only if the optimal solution to the instance $\tau(\xi)$ of MaxKeyEstabFlow on WSN and auxiliary graph representations has $c$ flows $(C, D)$ (i.e. flows $(C, D), (D, C) \in \mathcal{F}$) which are granted. (1)$MAX3SAT \rightarrow MaxKeyEstabFlow$: assume that $\tau(\xi)$ has an optimal solution $d > c$. Then it would be possible to satisfy more than $c$ clauses by simply setting $True$ value for the respective variables. This contradicts the fact that $\xi$ has an optimal solution $c$. (2) $MaxKeyEstabFlow \rightarrow MAX3SAT$: assume that $\xi$ has an optimal solution $d > c$. Then it would be possible to grant flow for $d$ source-destination pairs without contradicting the interference constraint. This contradicts the fact that $\tau(\xi)$ has an optimal solution $c$. □

**Definition 3.** *[9, Definition 10.4] A maximization problem $\Pi$ is MAX-SNP-Hard if for every MAX-SNP problem $\Gamma$ and every two constants $c \leq 1$, $\rho > 1$, there are two constants $c' \leq 1$, $\rho' > 1$ such that there is a gap preserving reduction from $\Gamma$ to $\Pi$ with parameters $(c, \rho, c', \rho')$.*

*Proof.* (P2.1 MaxKeyEstabFlow is in MAX-SNP) MAX3SAT is a MAX-SNP problem [15] where its optimum $c$ is a fraction equivalent to the maximum number of satisfiable clauses divided by the total number of clauses. It is NP-Hard to approximate MAX3SAT within a fixed ratio $\rho = 1 + \epsilon$ for $\epsilon > 0$. For proving inapproximability results, we use *gap preserving reduction* as described in Definition 3. For every $\epsilon > 0$, there is a gap preserving reduction from MAX3SAT to MaxKeyEstabFlow that has parameters (c, 1+$\epsilon$, c$|\mathcal{F}|/2$, 1+$\epsilon$) where $\mathcal{F}$ is the set of flows. We use the polynomial time reduction $\tau$ from MAX3SAT to MaxKeyEstabFlow described in the NP-hard proof of MaxKeyEstabFlow. Let $\varphi$ be a boolean formula in $3CNF$ form with $n$ variables and $m$ clauses. MAX3SAT($\varphi$) represents the maximum number of satisfiable clauses divided by the total number of clauses, and MaxKeyEstabFlow ($\tau(\varphi)$) represents the maximum number of flows that can be granted. We will show that MAX3SAT ($\varphi$) = c $\Leftrightarrow$ MaxKeyEstabFlow ($\tau(\varphi)$) = c.m. First, assume that MAX3SAT($\varphi$)=c. There must be *c.m* satisfied clauses. Each satisfied clause $C_i$ must have at least one satisfied variable where each of the corresponding transmitter nodes has one unit of flow, meaning that corresponding flow ($C_i$, $D_i$) can be granted. Thus, *MaxKeyEstabFlow* ($\tau(\varphi)$) $\geq$ *c.m.* Second, assume that MaxKeyEstabFlow ($\tau(\varphi)$) = c.m. There must be *c.m* flows granted. Each granted flow ($C_i$, $D_i$) means one satisfied clause $C_i$ so that MAX3SAT($\varphi$) $\geq$ *c*. Thus:

– $MAX3SAT(\varphi) = c \;\Rightarrow\; MaxKeyEstabFlow(\tau(\varphi)) = c.m$
– $MAX3SAT(\varphi) < \frac{c}{1+\epsilon} \Rightarrow MaxKeyEstabFlow(\tau(\varphi)) < \frac{c.m}{1+\epsilon}$.

This gap-preserving reduction from MAX3SAT shows that it is NP-Hard to approximate MaxKeyEstabFlow within factor $1 + \epsilon$. Thus, MaxKeyEstabFlow is MAX-SNP-Hard, meaning also that MaxKeyEstabFlow doesn't have a polynomial time approximation scheme (PTAS) unless $P = NP$. □

*Proof.* (P2.2 MinCostKeyEstabFlow is in both NP-hard and MAX-SNP-hard) We use the *Weighted MAX3SAT* problem where each clause has a weight, and the problem is to maximize the sum of the weights of satisfied clauses. *Weighted MAX3SAT* is a both NP-Hard and MAX-SNP-Hard [15] problem. We can show that MinCostKeyEstabFlow problem is both NP-Hard and MAX-SNP-Hard by using a polynomial time reduction from *Weighted MAX3SAT* to MinCostKeyEstabFlow which is obtained by adding two simple steps to reduction algorithm $\tau$ of NP-hard proof of MaxKeyEstabFlow. Consider a boolean formula $\varphi$ in $3CNF$ form with $n$ variables and $m$ clauses with weights (i.e. weight $w_i$ for the clause $C_i$). First, for $1 \leq i \leq m$ and $1 \leq j \leq n$, set cost $(-w_i/2)$ for the edge $(C_i, x_j)$ (a.k.a. $(C_i, \overline{x}_j)$) of WSN deployment where $x_j$ (a.k.a $\overline{x}_j$) appears in clause $C_i$ (set cost $(-w_i/2)$ for the edge $(C_i^{T_g}, x_j^R)$ (a.k.a. $(C_i^{T_g}, \overline{x}_j^R)$) of auxiliary graph representation where $1 \leq g \leq L_{max}$. All other edges have zero costs. Second, set $\mathcal{X} = 1$. Problem of maximizing the sum of the weights of the satisfied clauses becomes problem of minimizing the cost of granting one or more flows subject to interference constraint. The rest of the proof follows the discussions in NP-hard and MAX-SNP proofs of MaxKeyEstabFlow. We conclude that MinCostKeyEstabFlow problem is both NP-Hard and MAX-SNP-Hard, meaning also that MinCostKeyEstabFlow doesn't have a polynomial time approximation scheme (PTAS) unless $P = NP$. □

# Towards Privacy Preserving Mobile Internet Communications – How Close Can We Get?*

Kristian Gjøsteen, George Petrides, and Asgeir Steine

NTNU, Trondheim, Norway

**Abstract.** In today's 3rd Generation mobile communications, some privacy concerns of mobile phone users are still not dealt with. Most prominent of these is the fact that mobile network operators learn both the identity and location of each device at any given time in order to be able to provide seamless services like telephony and internet access. In addition, the same information can leak to independent eavesdroppers using special equipment. The purpose of this paper is to investigate the possibility of a privacy preserving alternative. Our findings show that we can achieve anonymous internet access for mobile devices, and can build conditional privacy preserving persistent connections to service providers on top of it. As an alternative to mobile telephony, user-to-user mobile internet telephony providing improved but not optimal levels of privacy can be achieved under realistic assumptions, the limitations being due to traffic analysis attacks.

**Keywords:** Privacy, Anonymity, Mobile communications.

## 1 Introduction

A functional requirement in mobile communications is that users of mobile phones give frequent updates of their location to the *mobile network operator* (MNO) they are connected to in order to be continuously able to receive calls and enjoy seamless communication. In today's 3rd Generation systems (UMTS [1]) this updating is done by initially sending the *International Mobile Subscriber Identity* (IMSI) that is embedded on their smart card and uniquely identifies them, and subsequently a *Temporary Mobile Subscriber Identity* (TMSI) generated by the MNO, to the MNO.

Despite the fact that IMSIs are transmitted as rarely as possible and TMSIs are changed frequently by the MNO in order to prevent user tracing by eavesdroppers on the radio link, no protection is currently offered against active attackers forcing the transmission of IMSIs instead of TMSIs[1]. Even if this issue gets resolved by modifications to the current architecture [1], more serious privacy concerns are raised by the MNO's ability to effortlessly learn each

* Funded by the Norwegian Research Council's VERDIKT programme project 183195.

[1] Such an attack is possible since an MNO can request a user's IMSI (sent in clear) before it authenticates itself to him, and can be mounted using an IMSI-catcher [6].

user's location at any given time, which can lead to abuse similar to that in the Deutsche Telekom scandal [9].

A related privacy issue is that when a mobile phone user wants to make a call, he has to inform his MNO of the callee's identity in order to get connected. In addition, the contents of telephone conversations are available to the MNO of both the caller and the callee. To sum it all up, MNOs learn everything there is to know: *who, where,* and *what.*

*Our contribution.* In this article we investigate the extent to which the above concerns can be addressed. The natural approach is to eliminate the source of the problem, namely the MNO's knowledge of the connecting user's identity, and see what can be achieved.

Although public key cryptography can offer solutions (anonymous authentication schemes such as [7,8] or the more general anonymous credential schemes of [2] for instance), and modern mobile devices often have adequate computing power to handle the associated expensive computations, it can also open up for *denial of service* (DoS) attacks against MNOs by anonymous attackers: since it is much easier to create a fake ciphertext than to validate one, a malicious user can anonymously send large amounts of fake ciphertexts to the MNO, thus forcing it to exhaust its resources in trying to check their validity. Preferably, we would like to reserve a player's ability to mount denial of service attacks until after that player has been authenticated and thus become identifiable.

For this reason we propose a solution under a modified setting, similar to what is realised today with *virtual mobile network operators* (VMNOs): MNOs own and maintain the network infrastructure but do not have direct relationship with customers. Instead, distinct intermediaries, the *service providers* (SPs), deal with user subscriptions for network access.

Assuming such a setting, mobile phone users (henceforth simply referred to as users) can anonymously establish encrypted communication channels with MNOs with indirect authentication via their subscribed SP every time they change their location. As a result, MNOs and active attackers cannot follow users as they move around network locations (except by a specific DoS attack or ordinary traffic analysis), while at the same time important features like user and network authentication and radio link encryption are maintained (Sect. 3).

The next step is to provide equivalents to today's services that preserve the acquired anonymity. First, we argue that it is straightforward for users to anonymously access the internet using ephemeral pseudonyms provided by MNOs (Sect. 3) and then sketch how through this anonymous internet connection users can establish persistent connections with various *content SPs* (CSPs) (Sect. 4).

Finally, we outline how seamless privacy preserving mobile telephony is possible with the aid of a CSP, the *telephony provider* (TP), but unfortunately only if both communicating parties are stationary. If at least one of them changes location during a call then traffic analysis can correlate new and old pseudonyms. Nonetheless, it is an improvement on the current state of affairs as both caller and callee are anonymous and the contents of their conversation private for the call's duration (Sect. 5).

At a first glance, the solutions we have proposed might seem not to comply with the laws of several countries, such as the European Union's Data Retention Directive [5], which require that records of information related to the activity of users, like network entry point location for instance, are kept for some specified amount of time. However, as will be seen in Sect. 3, anonymity and location hiding are made possible by replacing TMSIs by temporary user-generated session identifiers (SIDs) shared between MNOs and SPs. Therefore, if MNOs keep records of the locations of SIDs, and SPs records of users' identities associated with SIDs, a judicially invoked MNO–SP collaboration can disclose all the required information about specific users. Similarly, MNOs, SPs and TPs can together reveal all information regarding internet telephony since pseudonyms are associated with SIDs. Moreover, dividing the stored information into parts that on their own leak almost none of the private data of users, might have the effect of lessening the security impact of such controversial laws.

Crucial for the viability of our proposal is billing. Possible solutions that do not violate any of the anonymity requirements include fixed term contracts between users and SPs and SPs and MNOs (similar to current agreements between MNOs and VMNOs), or having MNOs charge SPs according to the number of their (anonymous) subscribers that connect to the network, without revealing any location information. This should be possible as MNOs and SPs will be responsible only for connecting users to the network. Additional services will be delivered by TPs and other CSPs, which for example can charge fixed subscription fees. In general, devising a suitable business plan can vary from entity to entity.

*Related work.* Anonymity has been considered in many fields of cryptography. Modern anonymising networks, which are based on the mix networks of [3] and onion routers (most notably the Tor project [11]), enhance user privacy by mixing up the network traffic of different users to make tracing and traffic analysis more difficult. Although they are interesting tools for obtaining unlinkability, in the mobile devices case they can only be used once the device has established connection to an MNO, a connection we require to be anonymous.

Other work on anonymity in mobile communications concerns a user's privacy with respect to various CSPs and eavesdroppers, whereas the MNO is assumed to know both his identity and location (e.g. [12]).

## 2 Model and Privacy Goals

Users are assumed to have direct communication only with MNOs over a *radio link* (RL). They enter and leave network positions (corresponding to base stations) and once in one, they receive all messages sent to it over the RL. In order to identify messages intended for them, they need to prepend every message they send to an MNO with a session identifier SID (like the IMSI or TMSI in the present-day setting), with which the MNO will also prepend the reply. An eavesdropper may choose to listen to the RL at certain positions, in which we assume he gains full control (i.e. the ability to intercept and inject messages).

We also assume the existence of a *secure communication channel* (SEC) between all MNOs and SPs to which the adversary has no access, and that the adversary is in full control of the internet over which all entities can have direct communication, apart from users which have to go through an MNO over RL.

Our privacy requirement is that each entity should only ever learn one piece of information about a user. MNOs and potential eavesdroppers always learn a user's position (i.e. the base station of connection) hence should never learn his identity or be able to correlate his various positions. As will be seen, in the schemes we develop SPs and CSPs learn the user's identity from the user himself, so they should never learn his position.

## 3 Secure and Anonymous Network Connection

A first step in achieving secure and anonymous connection between users and MNOs is establishing a common secret. The kind of key establishment suitable for the privacy issues we are trying to address is authenticated with one-sided anonymity, in the sense that only one of the parties (the user) is anonymous with respect to the other (the MNO). To achieve such anonymous authentication, a third party (the SP) needs to vouch that the user is indeed a subscriber without disclosing his identity and without learning his location. Note that vouching is not an uncommon technique but in this case implies that an MNO–SP collusion reveals everything. Figure 1 sketches a suitable protocol.

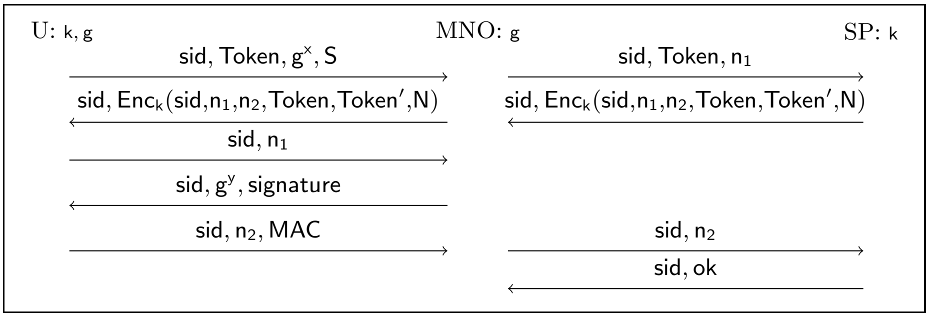


**Fig. 1.** Outline of an anonymous key establishment protocol. The user $U$ anonymously authenticates to the local MNO $N$ with the aid of his subscribed SP $S$. Communication between users and MNOs is over RL and between MNOs and SPs over SEC, both using a user-generated session identifier *sid*. $\mathsf{Enc}_k$ denotes encryption using the symmetric key $k$, and the signature and MAC (keyed with a function of the agreed secret $g^{xy}$, where $g$ is a generator of a multiplicative cyclic group of suitable prime order) are on the common user-MNO view at the time. A user should not run concurrent sessions.

The main idea is that instead of having a unique IMSI and a subscriber key (both shared with the MNO) embedded on their smart cards, users have a subscriber key shared with their subscribed SP, and an identity token created by the SP. As a result, instead of identifying themselves directly to the MNO using the IMSI, they can do so to the SP using the token which is indecipherable to the MNO. The SP can then confirm to the MNO that users are subscribers without disclosing their identity. Users and MNOs establish authenticated secrets using essentially a Diffie–Hellman key agreement [4].

*Authentication:* First note the requirement that the user never runs two instances of the protocol in parallel. If the ciphertext is decipherable, he knows it came from his SP, and if it contains the identifier sid it is not a replayed message. If it contains the original token sent, it means no one is trying to link this session to a previous one. When he successfully verifies the MNO's signature, he is convinced about the MNO's identity and hence the origin of the Diffie–Hellman partial key.

When the MNO receives its nonce $n_1$ from the user, it concludes that the user is a subscriber of SP since he was able to decrypt the ciphertext. On receiving the MAC from the user, the MNO is convinced that the user accepted the signature and agrees on all previous messages. At this point there are two possibilities for the origin of the Diffie–Hellman partial key: either an honest subscriber of the SP or a corrupted SP eavesdropping the radio link at this position, masquerading as a user. The latter case, though possible, is improbable as it lacks motivation.

Finally, when the SP receives back its nonce $n_2$, it is convinced of the user's identity and acknowledges this with the final ok message to the MNO.

*Anonymity and location privacy:* The SP should construct tokens as independent encryptions of the user's identity, and since no other part of the protocol contains information about the user's identity, users remain anonymous to anyone else.

The SP, unless listening from beforehand at the particular position and sees the token, learns nothing about the user's location.

*Linkability issues:* To avoid linkability, fresh secrets should be established using distinct tokens every time a user enters a new location. For this reason, tokens are independent encryptions of a user's identity and fresh tokens are sent encrypted during a protocol run. However, in case the protocol fails before a user receives a new token, reuse of the old token can make linking the new session with previous ones possible both for the MNO and any eavesdroppers. This situation is insignificant if the user remains in the same location, but can otherwise allow the tracking of anonymous users as they move around the network.

As an alternative that could prevent linkability, one can always choose to modify the protocol to use public key cryptography for the user's identification to the SP rather than tokens. We, however, prefer not to do so for the reason that token verification uses symmetric encryption which allows for fast and inexpensive ciphertext validity checks by the SP, whereas if using public key methods this can lead to troublesome DoS attacks, just as we mentioned in Sect. 1. In comparison, UMTS employs neither tokens nor public key cryptography: as already pointed out, IMSIs are sent in cleartext with consequence the possibility to track specific users.

After successfully establishing a secret, the user and the MNO can extract from it a temporary session identifier for the radio link, and two keys for encrypting the radio link communication, separated in upstream and downstream (a standard tool), thus achieving the required anonymous and secure connection. If secrets are authenticated and uncorrelated, then so are the communication channels, and if transmitted ciphertexts are authenticated then the adversary cannot create valid ones and inject them, but can only replay observed ones.

MNOs can then provide on-demand ephemeral pseudonyms to connected users for use as identifiers in internet communication. On providing them, a fresh nonce should be used to prevent replays that would result in users re-obtaining and reusing the same ones and hence in user tracking. Since internet communication for users is via MNOs which forward messages from senders to recipients (as assumed in Sect. 2), if pseudonyms are session-tied and contain identifying information on the issuing MNO, communication will be possible without leaking any user related information. Thus, internet access will be anonymous.

## 4 Persistent Connection to Content Service Providers

Users can utilise their internet pseudonyms in establishing location hiding persistent connections to various *content SPs* (CSPs) as sketched in Fig. 2.

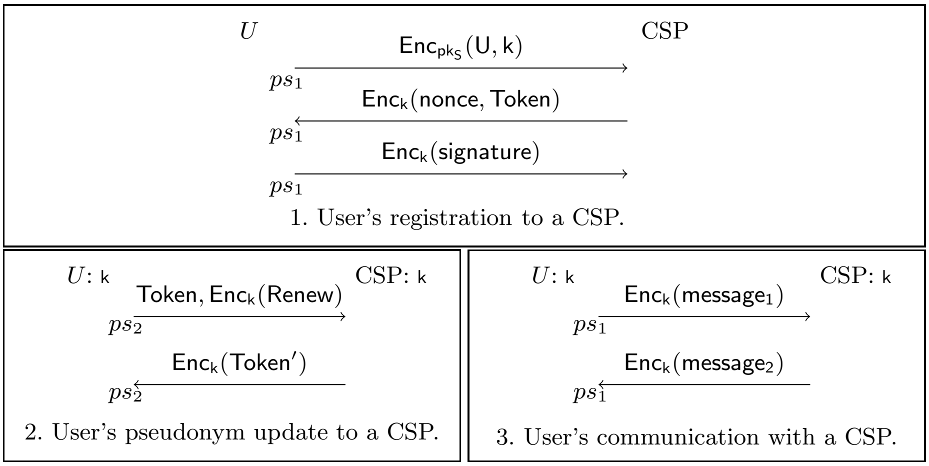


**Fig. 2.** Outline of a persistent connection protocol. Communication is over the internet using pseudonyms $ps_i$ via the issuing MNO. $\mathsf{Enc}_{pk_S}$ and $\mathsf{Enc}_k$ respectively denote encryption using CSP's public key and symmetric key $k$. The signature is on the common user-CSP view at the time.

*Authentication:* A user authenticates to a CSP during registration using a signature that includes the fresh nonce sent by the CSP. The CSP is authenticated by using the key sent by the user, as he is the sole owner of his decryption key. Use of the shared key authenticates any further communication between the two.

A user re-authenticates to a CSP from a new position by using the token received from the CSP encrypted under the shared symmetric key and receives a new token for further re-authentication.

Note that using public key encryption in the first communication between a user and a CSP can allow for a DoS attack as we have described earlier. However, a user mounting such an attack can be tracked down via his pseudonym. Therefore, the only real threat is when the attacker is someone with a radio transmitter that uses bogus pseudonyms, which is very difficult to track down.

*Anonymity and location privacy:* As in Sect. 3, if the CSP constructs tokens independently, users remain anonymous to anyone else.

By using pseudonyms, users do not leak any information on their location to the CSP they connect to.

*Linkability issues:* As a user moves around the network, neither the MNO nor external attackers should be able to correlate the user's various pseudonyms. Receiving a fresh token every time the user authenticates to the CSP from a new position makes subsequent re-authentication messages unlinkable. In case this new token is not received after a specified amount of time, the user re-registers with the CSP from scratch, thus retaining unlinkability. However, in special cases of identifiable traffic flow patterns (e.g. heavy flow), traffic analysis is enough to make the linking between pseudonyms possible, both by the MNO and external attackers.

## 5 Seamless Internet Telephony Services

By subscribing and establishing a persistent connection to a special CSP, the *telephony provider* (TP), a user can be called by other users as sketched in Fig. 3.

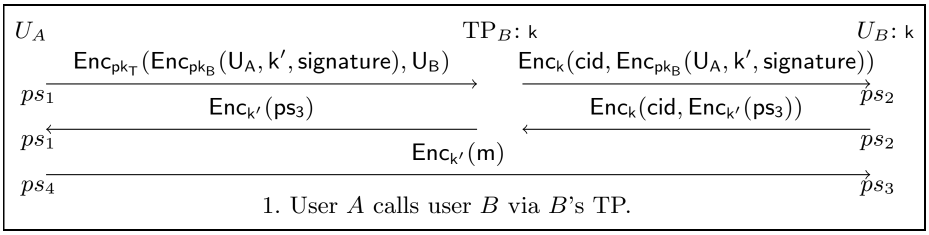

$U_A$
$\mathrm{TP}_B$: k
$U_B$: k
$\mathsf{Enc}_{\mathsf{pk_T}}(\mathsf{Enc}_{\mathsf{pk_B}}(\mathsf{U_A}, \mathsf{k}', \mathsf{signature}), \mathsf{U_B})$
$\mathsf{Enc}_{\mathsf{k}}(\mathsf{cid}, \mathsf{Enc}_{\mathsf{pk_B}}(\mathsf{U_A}, \mathsf{k}', \mathsf{signature}))$
$ps_1$
$ps_2$
$\mathsf{Enc}_{\mathsf{k}'}(\mathsf{ps_3})$
$\mathsf{Enc}_{\mathsf{k}}(\mathsf{cid}, \mathsf{Enc}_{\mathsf{k}'}(\mathsf{ps_3}))$
$ps_1$
$ps_2$
$\mathsf{Enc}_{\mathsf{k}'}(\mathsf{m})$
$ps_4$
$ps_3$
1. User $A$ calls user $B$ via $B$'s TP.

**Fig. 3.** Outline of call initialisation in a calling protocol. Communication is over the internet using pseudonyms $ps_i$ via the issuing MNO. $\mathsf{Enc}_{pk_T}$ and $\mathsf{Enc}_{pk_B}$ denote encryption using TP's and $U_B$'s public keys, $\mathsf{Enc}_k$ and $\mathsf{Enc}_{k'}$ encryption using symmetric keys $k$ and $k'$, and *cid* is a temporary call identifier.

The idea is that the caller requests, via the callee's TP (found in a publicly available phone directory), a pseudonym at which he can reach the callee directly. Once the callee accepts a call by sending the requested pseudonym via his TP, further communication with the caller will be private with respect to the TP via the direct sending of a symmetric encryption key and use of pseudonyms unknown to the TP.

*Authentication:* The caller need not authenticate to the TP, and authenticates to the callee using a signature. The callee is authenticated to the caller by using the symmetric key the caller sent him, as he is the only one who could decrypt the message containing it. Use of the shared symmetric key authenticates any further communication between the two. Regarding DoS attacks, the same applies as in Sect. 4.

*Anonymity and location privacy:* The use of public key cryptography and pseudonyms ensures that the caller's identity is revealed only to the callee and no one but the TP learns that someone is calling the callee.

Once again, due to the use of pseudonyms, no part of the protocol contains information on the user's location and hence TPs learn nothing about it.

*Linkability issues:* If a user moves while engaged in a conversation, changing his pseudonym cannot protect him from traffic analysis by eavesdroppers as the person he is calling with will still be using the same pseudonym (or a different one that the MNO can associate to the previous) and the pseudonyms can be correlated.

## 6 Conclusions

In this paper we have sketched how a privacy-preserving alternative to the current state of affairs in mobile communications can be achieved, though with certain limitations and without considering access to location based services. The added computational overhead due to the required signature verification and key establishment exponentiations should not have a noticeable impact on the batteries of mobile phones, unless users move around base stations too frequently. What we have presented is quite abstract, and the interested reader is referred to [10] where universally composable protocols capturing most of these ideas are described and analysed.

## References

1. 3GPP TS 33.102: Security Architecture, Ver. 11.5.0 (2013), `www.3gpp.org/ftp/Specs/html-info/33102.htm`
2. Camenisch, J., Lysyanskaya, A.: An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer, Heidelberg (2001)

3. Chaum, D.: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. Communications of ACM 24(2), 84–88 (1981)
4. Diffie, W., Hellman, M.: New Directions in Cryptography. IEEE Transactions on Information Theory 22(6), 644–654 (1976)
5. The European Parliament and Council: Directive 2006/24/EC. L 105, pp. 54–63 (2006)
6. Fox, D.: Der IMSI-Catcher. Datenschutz und Datensicherheit 26(4) (2002)
7. Lindell, Y.: Anonymous Authentication. Journal of Privacy and Confidentiality 2(2), 35–63 (2010), `repository.cmu.edu/jpc/vol2/iss2/4`
8. Nguyen, L., Safavi-Naini, R.: Dynamic $k$-Times Anonymous Authentication. In: Ioannidis, J., Keromytis, A., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 318–333. Springer, Heidelberg (2005)
9. Spiegel Online: Did Deutsche Telekom Spy on Journalists and Board Members? (2008), `www.spiegel.de/international/business/0,1518,555363,00.html`
10. Steine, A.: Privacy-Preserving Cryptographic Protocols. PhD Thesis, Norwegian University of Science and Technology (2012)
11. The Tor Project, `www.torproject.org/index.html.en`
12. Wachsmann, C., Chen, L., Dietrich, K., Löhr, H., Sadeghi, A.-R., Winter, J.: Lightweight Anonymous Authentication with TLS and DAA for Embedded Mobile Devices. In: Burmester, M., Tsudik, G., Magliveras, S., Ilić, I. (eds.) ISC 2010. LNCS, vol. 6531, pp. 84–98. Springer, Heidelberg (2011)

# Count-Min Sketches for Estimating Password Frequency within Hamming Distance Two

Leah South and Douglas Stebila

School of Mathematical Sciences, Queensland University of Technology,
Brisbane, Queensland, Australia
leah.south@connect.qut.edu.au, stebila@qut.edu.au

**Abstract.** The *count-min sketch* is a useful data structure for recording and estimating the frequency of string occurrences, such as passwords, in sub-linear space with high accuracy. However, it cannot be used to draw conclusions on groups of strings that are similar, for example close in Hamming distance. This paper introduces a variant of the count-min sketch which allows for estimating counts within a specified Hamming distance of the queried string. This variant can be used to prevent users from choosing popular passwords, like the original sketch, but it also allows for a more efficient method of analysing password statistics.

**Keywords:** count-min sketch, Bloom filter, password frequency, approximate string matching.

## 1 Introduction

The use of passwords for identity verification is widespread. There is a long line of research on analyzing the security and guessability of password [8,3]. In large online systems that we see on the Internet today, an important characteristic that affects the overall security of the system is that passwords within the system should not be too *popular*. In an ideal setting, of course, users would create a unique password that is hard to guess, and not popular, so that only that user and no one else could access their account. However, users tend to choose passwords that are easy to remember and familiar to them, such as dictionary words, or perhaps strings associated with the system in question. This tendency means that certain passwords are used with higher frequency, making them popular.

If an attacker knew the distribution of passwords, they could use its statistics and guess the most popular passwords first. This is known as the *statistical guessing technique* [9]. When there is a high percentage of popular passwords, the attacker can compromise a high percentage of accounts. For example, the 2009 breach of RockYou.com's 32 million account password database showed that the most popular password (`123456`) was used by 0.9% of all accounts, and the next 4 most popular passwords (`12345`, `123456789`, `password`, and `iloveyou`) were used by another 0.8% of all accounts. Clearly, the system was not screening passwords for popularity. As a result, a statistical guessing attack would lead to millions of accounts being compromised. In order to prevent a successful statistical guessing attack such as this, it is common to limit the number of guesses; more recently, it has been proposed [9] to limit the popularity of passwords: when users try to set their password to a string that is used in a percentage of accounts above some threshold, it is rejected and the user is required to choose another password.

In order to keep track of password popularity, some sort of system which counts passwords must be used. Online sites with a large number of users are best suited for systems which restrict popular passwords, as such sites are at high risk of *trawling attacks*, in which attackers aim to guess the passwords to many accounts without targeting any single account.

How can we store password information in a way that allows us to calculate frequency when users attempt to register a password? The simplest technique for calculating frequency during password registration would be to store a separate table of passwords along with their frequency. This is undesirable both for efficiency reasons (since the size of the table grows linearly in the number of distinct passwords) and for security reasons (since it immediately provides an attacker with the full distribution of passwords). Best-practice recommendations for storing passwords for login involve storing salted password hashes for each account; the set of such passwords does not admit statistical analysis since, by design, the hash of the same password under different salts yields different, seemingly independent, outputs, thus yielding no information about the frequency with which a password is used.

## 1.1 Bloom Filters and Count-Min Sketches

The *Bloom filter* [2] can be used [10] to store in sub-linear space a table representing a dictionary of prohibited passwords. The system is setup as follows. A $w \times h$ table $T$ of bits is used, along with $h$ independent hash functions $\mathsf{hash}_1, \ldots, \mathsf{hash}_h : \{\mathtt{A}-\mathtt{Z}, \mathtt{a}-\mathtt{z}, \mathtt{0}-\mathtt{9}, \ldots\}^* \rightarrow \{1, \ldots, w\}$. Each word $x$ in the dictionary of prohibited passwords is hashed under each hash function $\mathsf{hash}_k$, and the entry $T_{\mathsf{hash}_k(x),k}$ is set to 1. When a proposed password $y$ is to be tested for membership in the list of prohibited passwords, if all of the values $T_{\mathsf{hash}_k(y),k}$ are equal to 1, then $y$ is deemed to be prohibited, but if at least one of those table entries is zero, then $y$ is not prohibited. There are no *false negatives*, meaning that it is impossible for a password that is prohibited to not be recognized as such, but there may be *false positives*, meaning that some passwords that are not prohibited may, due to collisions on all rows, still be identified by the table as prohibited. Assuming the hash functions are independent random functions, the false positive rate $(1-(1-\frac{h}{w})^N)^h$, where $N$ is the number of prohibited passwords originally added to the table [10].

The *count-min sketch* [5] enhances the Bloom filter by storing a table of integers, not bits. The $\mathsf{update}(x, c)$ function records that string $x$ has been used $c$ more times by adding $c$ to each table entry $T_{\mathsf{hash}_k(x),k}$. The $\mathsf{estimate}(x)$ function returns an estimate on the number of times that string $x$ has been used by computing $\min\{T_{\mathsf{hash}_k(x),k} : 1 \leq k \leq h\}$. The use of count-min sketches for recording password frequency was proposed by Schechter, Herley, and Mitzenmacher [9], who propose preventing users from registering with passwords whose current popularity is above a certain threshold. As with the Bloom filter, false negatives cannot occur, meaning that for any password $x$ it is impossible for the $\mathsf{estimate}(x)$ to return a value lower than the sum of the $c$ over all calls of $\mathsf{update}(x, c)$. However, false positives may still occur, meaning that $\mathsf{estimate}(x)$ may over-estimate the frequency of password $x$, due to collisions across all hash functions. For a single hash, the expected error due to collisions is $N/w$, where $N$ is the total sum of the counts of all passwords; this is because the total count of $N$ will spread approximately evenly across all $w$ columns in the table. By the Markov inequality, the error of the count min-sketch (the minimum across all hash functions) is at most $2N/w$ with probability at least $1 - (\frac{1}{2})^h$.

### 1.2 Contributions

When used for passwords, the Bloom filter and the count-min sketch can be useful in prohibiting certain passwords or limiting the frequency with which any password is registered. However, they cannot accommodate any relation between 'similar' passwords. For example, a user who tries to register the password `password` and finds that it is prohibited may try again with `p@ssword` or `passw∅rd`. An attacker, knowing this prevalence for 'leetspeak', may also make use of these similarities in an attack strategy that targets 'almost popular' passwords. Since the Bloom filter and count-min sketch use independent random passwords, they lose any semantic connection between such similar strings.

In this work, we explore a variant of the count-min sketch that allows one to compute estimates not only for the given string but for all strings within a fixed Hamming distance of the given string. This technique allows a system to detect frequently used passwords at the registration phase. Our technical approach is to introduce 'wildcard' characters and then record and estimate based on all strings that can be constructed with wildcard characters within the required Hamming distance. This technique imposes a computational overhead of $\binom{\ell}{d}$, where $\ell$ is the length of the password in question and $d$ is the maximum Hamming distance. This compares favourably with the naive technique which would have a computational overhead of $\binom{\ell\alpha}{d}$, where $\alpha$ is the size of the alphabet. We provide false positive error rates as well, and compared with the naive method our technique provides better accuracy for the same size table for a wide range of password lengths. Our technique can naturally be extended to higher Hamming distances.

## 2 Related Work

While the Bloom filter and count-min sketch are widely used, there are several other methods to store estimate counts, some of which may be useful in obtaining statistics on groups of similar passwords.

There have been some systems in the past which have stored cleartext passwords alongside their counts. This method is mostly outdated due to the lack of security it provides. If attackers gain access to password databases such as these, a highly successful statistical guessing attack can be carried out because the actual passwords are visible and there are no false positives - the exact counts are known. Using hash functions is preferable to this method.

Decision trees, as suggested by Bergadano, Crispo and Ruffo, can be used to determine password membership. These decision trees consist of nodes for particular attributes, arcs for values of these attributes and leaves for classification of whether or not the password has been used before in that database [1]. When using decision trees, there is a training phase in which the nodes, arcs and leaves are chosen to produce the best results. For each update to the database, the training phase must be repeated. Due to this, decision trees can be used for determining membership but are not as practical when frequent updates may be involved.

There has also been some work on the topic of determining popularity of similar words. This includes work which allows for checking membership of words within Levenshtein distance 1, that is words which have only one insertion, deletion or substitution. Manber and Wu [7] suggested an approach similar to the original Bloom filter, with the main difference being that the membership of words within Levenshtein distance 1 are checked in the estimation stage. It is proposed that if any password within distance 1 appears to be positive, the password cannot be used [7]. As with all Bloom filter-based techniques, this only records the binary data of whether two similar passwords have been used, not counts on how frequently similar passwords are used.

## 3 Construction

The adaptation of the original count-min sketch that is introduced in this paper allows for the popularity estimations of words within Hamming distance zero to two, that is words that differ by up to two substitutions. Like the count-min sketch, this adaptation consists of three main parts: *hashing* the passwords, *updating* the table and *estimating* the counts. In Appendix A, we provide a worked example of each of the three operations—hashing, updating, and estimating.

### 3.1 Hash

We represent passwords $x$ as integers $X$. Any canonical representation of a word as an integer will suffice. For expository convenience, we can imagine a mapping of an $\ell$-letter word $x = x_1 \dots x_\ell$ where each character $x_i$ is coded as a two-digit integer $X_i \in \{00, \dots, 98\}$; the integer 99 is reserved to represent a wildcard character $*$. The vector $\langle X_1, \dots, X_\ell \rangle$ is then viewed as an integer $X = \sum_{i=1}^{\ell} 100^{i-1} X_i$. This suffices to encode for example all passwords that can be typed using characters on a standard US keyboard.

As in [6], each hash function $\mathsf{hash}_k$ is a Carter–Wegman 2-universal hash function [4] of the form

$$\mathsf{hash}_k(x) = (((a_k X + b_k) \bmod p_k) \bmod w) + 1 \enspace ,$$

where $X$ is the canonical integer representation of the word $x$, $p_k$ is a prime number much larger than $w$ (recall $w$ is the width of the table), say $p_k = 2^{31}-1$ or $p_i = 2^{61}-1$, and $a_k$ and $b_k$ are chosen uniformly at random from $\{0, \ldots, p_k - 1\}$. We can simplify to using the same prime $p = p_1 = \cdots = p_h$ across all hash functions, but all $a_k$ and $b_k$ need to be chosen independently to ensure negligible probability of collisions; otherwise the benefits of using multiple hash functions will be eliminated.

We define the vector-wise hash function $\mathsf{hash}(x)$ where the $k$th entry of $\mathsf{hash}(x)$ consists of $\mathsf{hash}_k(x)$ with $a_k$ and $b_k$.

## 3.2 Update

The function $\mathsf{update}(T, x, a, b, c) \to T'$ updates the sketch based on the previous table $T$, the password $x$ to be updated, the values or vectors of $a$ and $b$ for the hash functions and the amount of times $c$ that the password is being added. The update algorithm in this work differs greatly to that in the traditional count-min sketch.

Firstly, the count for the actual password is updated. Like the original count-min sketch, this is done by finding the hash of the password for each hash function then updating the count at these positions in the table. Next, the updates are done for words within distance 1. To do this, a 'wildcard character' is introduced. This wildcard character, which for our canonical encoding above is denoted by 99, is used to group similar passwords together: for example if the word `abcd`, or $\langle \mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d} \rangle$, is represented in integer form as $\langle 1, 2, 3, 4 \rangle$, then $\langle 99, 2, 3, 4 \rangle$ represents `*bcd`, all four letter words ending in `bcd`. A wildcard character indicates that any character could go in its place. After the count of the actual password is updated, all wildcard passwords within Hamming distance 1 are updated by creating $\ell = \text{length}(x)$ variations of the password, each with a different character from the initial word replaced with 99, then hashing these variations and updating their positions in the table. Next, the wildcard words within Hamming distance 2 of the password $x$ are hashed and their positions in the table are updated.

In Algorithm 1, `update`, using the wildcard character means that $\ell$ additional updates are done for adding wildcard words within distance 1. The alternative, naive technique, would be to search all $\alpha\ell$ passwords within distance 1 during the estimate algorithm, where $\alpha$ is the size of the alphabet. When the password length $\ell \ll \alpha$, the proposed technique is much more efficient than the naive technique.

## 3.3 Estimate

The function $\mathsf{estimate}(T, x, a, b)$ is used to obtain estimates of the count of passwords at *exactly* Hamming distances 0, 1, and 2 of $x$ and is specified in full in Algorithm 2.[1]

[1] Note that we have `estimate` return estimates for counts of passwords at exactly, rather than within, the specified Hamming distance for clarity of exposition; estimates within the specified distance can be found by summing the estimates exactly at all distances less than or equal to.

**Algorithm 1.** $\mathsf{update}(T, x, a, b, c)$

```
1:  $T' \leftarrow T$
2:  $H \leftarrow \mathsf{hash}(x, a, b)$
3:  for $k = 1$ to $h$ do
4:      $T'_{H_k,k} \leftarrow T'_{H_k,k} + c$
5:  for $i = 1$ to $\ell = \text{length}(x)$ do
6:      $x' \leftarrow x$
7:      $x'_i \leftarrow 99$
8:      $H' \leftarrow \mathsf{hash}(x', a, b)$
9:      for $k = 1$ to $h$ do
10:         $T'_{H'_k,k} \leftarrow T'_{H'_k,k} + c$
11: for $i = 1$ to $\ell - 1$ do
12:     for $j = i + 1$ to $\ell$ do
13:         $x'' \leftarrow x$
14:         $x''_i \leftarrow 99$
15:         $x''_j \leftarrow 99$
16:         $H'' \leftarrow \mathsf{hash}(x'', a, b)$
17:         for $k = 1$ to $h$ do
18:             $T'_{H''_k,k} \leftarrow T'_{H''_k,k} + c$
19: return $T'$
```

*At distance 0.* When the specified Hamming distance $d$ is zero, that is, when the count of the actual word is desired, the estimate process is very similar to the original count-min sketch. The $h$ different hashes of the password and the counts at all $h$ positions in the table are found. These counts can then be put into a vector $\mathsf{est}^0$ of length $h$, where each element represents the count at a different hash function. The resulting estimate $\overline{\mathsf{est}}^0$ is the minimum of these counts.

*At distance 1.* At a maximum Hamming distance of one, the process is slightly more complex. First, estimates $\mathsf{est}^1_i$, $i = 1, \ldots, \ell$, for the frequency of words that may differ from $x$ in the $i$th character; each of these estimates can be found using the technique estimating the (distance 0) occurrences of the wildcard word. In other words, $\mathsf{est}^1_i$ is the (distance 0) estimate for the wildcard "word" $x'$ where the $i$th character of $x$ has been replaced with the special wildcard character. Once estimates $\mathsf{est}^1_i$ for the frequency of words that may differ from $x$ in the $i$th character have been found, we can find an estimate $\overline{\mathsf{est}}^1$ for the number of occurrences of words at distance 1 from $x$ by summing $\mathsf{est}^1_i$ for all $i$, then subtract $\ell\overline{\mathsf{est}}^0$.

The reason for the subtraction is as follows. As explained in the $\mathsf{update}$ section, when a single password $x$ is added into the database, the counts for the exact password and the counts for all $\ell = \text{length}(x)$ passwords within distance 1 are all increased. In the estimation stage, the counts for all passwords with one wildcard character, or within distance 1, are summed together. This means that the count for the exact password has been included $\ell$ times in the calculation, when it should only have been included once. To overcome this problem, the count of that exact password multiplied by the length of the password is subtracted.

*At distance 2.* The frequency estimate $\overline{\mathsf{est}}^2$ for passwords at distance 2 is found in a similar manner to that of distance 1. For each of the $\binom{\ell}{2}$ wildcard passwords of $x$ that may differ in the $i$th and $j$th characters, we compute (distance 0) estimates $\mathsf{est}^2_{i,j}$ of the wildcard "word" $x''$ where the $i$th and $j$th characters of $x$ have been replaced with the special wildcard character. We then sum these estimates of words within distance 2. However, again we have overcounted. The words at precisely distance 1 have been counted $\ell - 1$ times. Similarly, the word at precisely distance 0 has been counted $\binom{\ell}{2}$ times.

**Algorithm 2.** $\mathsf{estimate}(T, x, a, b)$

```
1:  $H \leftarrow \mathsf{hash}(x, a, b)$
2:  for $k = 1$ to $h$ do
3:      $\mathsf{est}^0_k \leftarrow T_{H_k,k}$
4:  $\overline{\mathsf{est}}^0 \leftarrow \min_k\{\mathsf{est}^0_k\}$
5:  for $i = 1$ to $\ell = \text{length}(x)$ do
6:      $x' \leftarrow x$
7:      $x'_i \leftarrow 99$
8:      $H' \leftarrow \mathsf{hash}(x', a, b)$
9:      for $k = 1$ to $h$ do
10:         $\mathsf{est}^1_{i,k} \leftarrow T_{H'_k,k}$
11:     $\mathsf{est}^1_i \leftarrow \min_k\{\mathsf{est}^1_{i,k}\}$
12: $\overline{\mathsf{est}}^1 \leftarrow \left(\sum_i \mathsf{est}^1_i\right) - \ell\,\overline{\mathsf{est}}^0$
13: for $i = 1$ to $\ell - 1$ do
14:     for $j = i + 1$ to $\ell$ do
15:         $x'' \leftarrow x$
16:         $x''_i \leftarrow 99$
17:         $x''_j \leftarrow 99$
18:         $H'' \leftarrow \mathsf{hash}(x'', a, b)$
19:         for $k = 1$ to $h$ do
20:             $\mathsf{est}^2_{i,j,k} \leftarrow T_{H''_k,k}$
21:         $\mathsf{est}^2_{i,j} \leftarrow \min_k\{\mathsf{est}^2_{i,j,k}\}$
22: $\overline{\mathsf{est}}^2 \leftarrow \left(\sum_{i,j} \mathsf{est}^2_{i,j}\right) - (\ell - 1)\,\overline{\mathsf{est}}^1 - \binom{\ell}{2}\,\overline{\mathsf{est}}^0$
23: return $\langle \overline{\mathsf{est}}^0, \overline{\mathsf{est}}^1, \overline{\mathsf{est}}^2 \rangle$
```

## 4 Analysis of Construction

### 4.1 Error

Like the original count-min sketch, collisions can occur in this adaptation. These collisions result in some error, causing either slight overestimations of counts or false positives.

The error in estimating exact words can be expressed fairly simply. In determining this error relative to the original sketch, it is important to note that the total count for this method is not the same as what it would be in the original count-min sketch. In this adaptation of the sketch, the total count across the sketch is $(1+\ell+\binom{\ell}{2})$ times larger than the actual total of passwords used. This additional count is the result of the update stage, in which the password itself, all $\ell$ passwords within distance 1 and all $\binom{\ell}{2}$ combinations of passwords within distance 2 are added into the sketch. Let $\hat{N} = (1+\ell+\binom{\ell}{2})N$ denote the total count on the new sketch, where N is the total number of (non-distinct) passwords entered. In general, hash outputs are expected to spread approximately evenly across all columns of the table, making the average count per hash the quotient of the total sum of counts and the width $w$ of the table: $\hat{N}/w$. The expected maximum error is therefore $\hat{N}/w$, which may occur, for example, if an estimate is done on a password which is not present but happens to have the same hash across all $h$ hash functions as other passwords which have a count of $\hat{N}/w$.

The expected maximum error for estimating counts of passwords at distance 1 can be derived from the formula for distance 1 in Algorithm 2:

$$\text{error}(\overline{\mathsf{est}}^1) \leq \text{error}\left(\sum_i \mathsf{est}_i^1\right) + \text{error}(\ell\,\overline{\mathsf{est}}^0) \enspace .$$

Since each term of $\sum_i \mathsf{est}_i^1$ is calculated as distance-0 estimate, the error of each term is the same as the error in a distance-0 estimate, and thus

$$\text{error}(\overline{\mathsf{est}}^1) \leq \ell\,\text{error}(\overline{\mathsf{est}}^0) + \text{error}(\ell\,\overline{\mathsf{est}}^0) \leq 2\ell\,\text{error}(\overline{\mathsf{est}}^0) = 2\ell\frac{\hat{N}}{w} \enspace .$$

For passwords at distance 2, there is another increase in error. Based on the estimate for words at distance 2 in Algorithm 2, the expected maximum error is

$$\text{error}(\overline{\mathsf{est}}^2) \leq \text{error}\left(\sum_{i,j} \mathsf{est}_{i,j}^2\right) + \text{error}((\ell-1)\,\overline{\mathsf{est}}^1) + \text{error}\left(\binom{\ell}{2}\overline{\mathsf{est}}^0\right) \enspace .$$

Since each term of $\sum_{i,j} \mathsf{est}_{i,j}^2$ is calculated as distance-0 estimate, the error of each term is the same as the error in a distance-0 estimate, and thus

$$\text{error}(\overline{\mathsf{est}}^2) \leq \binom{\ell}{2}\text{error}(\overline{\mathsf{est}}^0) + \text{error}((\ell-1)\,\overline{\mathsf{est}}^1) + \text{error}\left(\binom{\ell}{2}\overline{\mathsf{est}}^0\right) \enspace .$$

Simplifying and substituting, we get

$$\begin{aligned}
\mathrm{error}(\overline{\mathsf{est}}^2) &\le \binom{\ell}{2}\mathrm{error}(\overline{\mathsf{est}}^0) + (\ell-1)\,\mathrm{error}(\overline{\mathsf{est}}^1) + \binom{\ell}{2}\mathrm{error}(\overline{\mathsf{est}}^0) \\
&\le \left(\binom{\ell}{2} + (\ell-1)(2\ell) + \binom{\ell}{2}\right)\mathrm{error}(\overline{\mathsf{est}}^0) \\
&= \left(2\binom{\ell}{2} + 2\ell(\ell-1)\right)\mathrm{error}(\overline{\mathsf{est}}^0) \\
&= \left(2\binom{\ell}{2} + 4\binom{\ell}{2}\right)\mathrm{error}(\overline{\mathsf{est}}^0) \\
&= 6\binom{\ell}{2}\mathrm{error}(\overline{\mathsf{est}}^0) = 6\binom{\ell}{2}\frac{\hat{N}}{w}\ .
\end{aligned}$$

## 4.2 Comparison with Naive Method

The purpose of the adaptation is to provide a more effective method of estimating counts of passwords within specified Hamming distances. While the count-min sketch can be used to do this, this *naive method* using the original sketch—incrementing the entry for every one of the $\alpha\binom{\ell}{d}$ words within distance $d$ in an alphabet of size $\alpha$—would be less efficient and have a higher error rate.

In this section, a comparison of efficiency and error is carried out. All of the following graphs have the same response variable: the expected average error as a multiple of $\frac{N}{w}$ where $N$ is the total number of passwords entered (also the total count in the original sketch) and $w$ is the width of the table.

*Estimates at distance 0.* When estimating how many times a specific word appears in a database (i.e. estimating passwords at distance zero), the original count-min sketch is more effective. The expected maximum error in estimating counts is $\frac{N}{w}$ in the original method and $\frac{\hat{N}}{w}$, or $(1+\ell+\binom{\ell}{2})N/w$, in this variation. Therefore the expected maximum error for estimating counts at distance zero will always be $1+\ell+\binom{\ell}{2}$ times larger in this variation. However, this increase in error is not a major problem since this sketch was not designed for estimating exact word counts.

*Estimates at distance 1.* The benefits of using this modified sketch are more evident when estimating counts at distance 1 of a password. In order to estimate counts at distance 1 of a specified password using the original sketch, multiple estimates would have to be done. More specifically, the number of estimates would be $\alpha\ell$, where $\alpha$ is the size of the alphabet and $\ell$ is the length of the password. Since the error in estimating each individual password is $\frac{N}{w}$, the expected maximum error in estimating $\alpha\ell$ passwords is $\alpha\ell N/w$. With the new method, the expected maximum error in estimating words at distance 1 is $2\ell\hat{N}/w = 2\ell(1+\ell+\binom{\ell}{2})N/w$. A comparison of these errors can be seen in Figure 1, where the size of the alphabet has been set to $\alpha = 72$ (26 uppercase and

26 lowercase letters, 10 numbers, and 10 special characters). From this graph, it can be seen that the error for the proposed method is lower until the length of the password reaches nine characters.

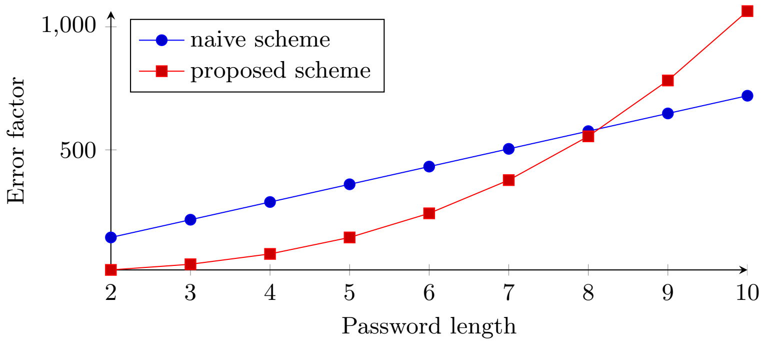


**Fig. 1.** Error as a multiple of $N/w$ for passwords at distance **1**, using a count-min sketch supporting maximum distance of **2**, compared with naive scheme

Note that this calculation above assumes that the `update` stage is as described in Section 3.2, allowing for estimates within distance 2, even though we are currently estimating distance 1. If the sketch is *not* going to be used to estimate counts at distance 2 at all, then it is possible to remove the distance 2 section. Updates for passwords at distance 2 would then not be included, making the expected maximum error for words at distance 1 equal to $2\ell(1+\ell)N/w$. A graph comparing the expected maximum error using the traditional method and this variation in which only distance 1 is included can be seen below, in Figure 2. For this graph, the size of the alphabet is again $\alpha = 72$. The proposed method has a lower error than the naive method until the length of the password reaches 36 characters.

*Estimates at distance 2.* For passwords at distance 2, $\binom{\ell}{2}\alpha$ estimates would have to be done using the original sketch. This makes the error for the original sketch $\alpha\binom{\ell}{2}N/w$ whereas our proposed technique has an error of $6\binom{\ell}{2}(1+\ell+\binom{\ell}{2})N/w$. When the size of the alphabet is $\alpha = 72$, the resulting differences in error can be seen in Figure 3: the error when using our proposed scheme is better than the naive method when the length of the word is less than 24 characters.

## 5 Example Parameter Instantiation

By using the error estimates found in previous sections, it is possible to estimate the size of the table required in order to obtain certain levels of accuracy. To do this, the Markov inequality can be applied to all expected maximum errors. The results are as follows:

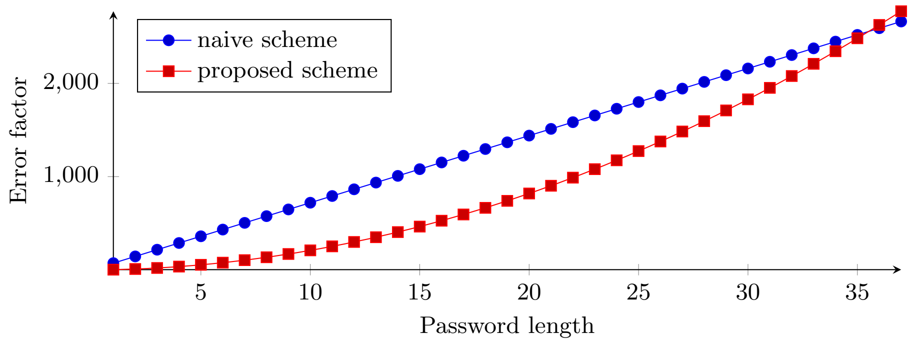


**Fig. 2.** Error as a multiple of $N/w$ for passwords at distance **1**, using a count-min sketch supporting maximum distance of **1**, compared with naive scheme

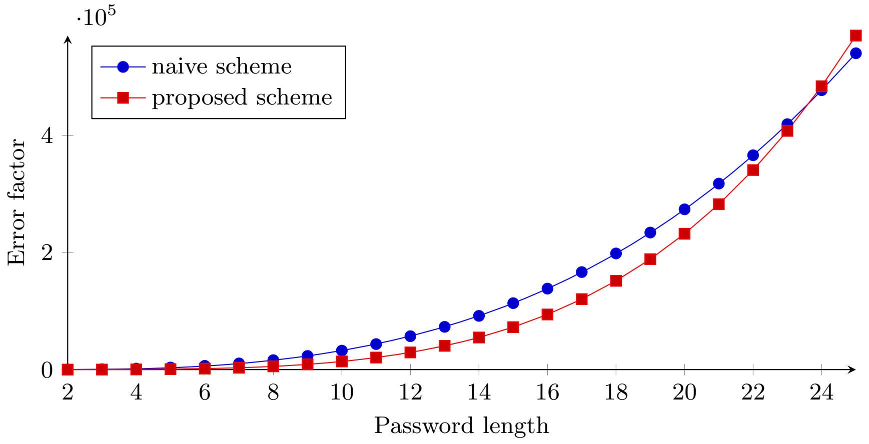


**Fig. 3.** Error as a multiple of $N/w$ for passwords at distance 2, compared with naive scheme

- When estimating the number of times a specific word appears, there is an error of at most $2(1+\ell+\binom{\ell}{2})N/w$ with probability of at least $1-(\frac{1}{2})^h$, where h is the number of hash functions.
- There is an error of at most $4\ell(1+\ell+\binom{\ell}{2})N/w$ with probability of at least $1-(\frac{1}{2})^h$ when estimating words within distance 1.
- There is an error of at most $12\binom{\ell}{2}(1+\ell+\binom{\ell}{2})N/w$ with probability of at least $1-(\frac{1}{2})^h$ when estimating words within distance 2.

The following exact parameter calculations are done for distances one and two, but not for Hamming distance zero because, if the user desires a certain error rate for estimating exact words, it is preferable to use the original count-min sketch.

If the purpose of this sketch was to estimate passwords within distance 1 with an error of at most 1% with probability of at least 99.9%, then the width $w$ of the table and the number $h$ of hash functions would have to be as follows:

$$\frac{4\ell(1+\ell+\binom{\ell}{2})}{w} = \frac{1}{100} \implies w = 100 \cdot 4\ell\left(1+\ell+\binom{\ell}{2}\right)$$

$$1-\left(\frac{1}{2}\right)^h = \frac{999}{1000} \implies h \approx 6.64$$

The number of hash functions would have to be 7 or more and the width of the table would depend on the length of the password. If the length of the passwords is 6 characters, then the width of the table would have to be at least 48,400. This size would be smaller if the table only included passwords within distance zero to one, as suggested previously.

Similarly, if the sketch was needed to estimate passwords within distance 2 with an error of at most 1% with probability of at least 99.9%, then the number of hash functions would still be at least 7 but the width of the table would be:

$$\frac{12\binom{\ell}{2}(1+\ell+\binom{\ell}{2})}{w} = \frac{1}{100} \implies w = 100 \cdot 12\binom{\ell}{2}\left(1+\ell+\binom{\ell}{2}\right)$$

For 6-character passwords, this would make the width 338,800. While this width seems large, it is possible that the false positive (or error) rate could be higher when estimating words within specified distances.

## 6 Conclusion

When no restrictions are placed on passwords choices, users tend to choose popular passwords. This leaves systems vulnerable to statistical guessing attacks. By limiting the percentage of popular passwords, these attacks are not as successful. In order to do this, efficient tools must be available to track password usage. The count-min sketch can be used to estimate the counts of password usage within a system. However, the count-min sketch is not as effective when estimating the counts of passwords within specified Hamming distances. In this paper, we have proposed a variant of the count-min sketch using wildcard characters that can be used to calculate estimates of words that are close in Hamming distance.

As with the original count-min sketch, there will never be false negatives — where the estimate algorithm under-reports usage of the password — but there may be false positives — meaning the estimate algorithm may over-report usage of the password due to collisions. We have calculated the error rate for estimation as the Hamming distance increases, which allows for calculation of sketch size for a given error rate. For a reasonable alphabet size, the error rates in our proposed method are lower than they would be using the naive approach on a standard count-min sketch for a wide range of password lengths, up to 36- and 24-character passwords when estimating passwords within distances 1 and 2 respective.

Our technique is most suited when all passwords in the database have the same length. In future work, it may be desirable to develop another variation in which *edit distance*, such as Levenshtein distance, is used, rather than Hamming distance, to take into account passwords that are close due to deletions or insertions of characters. Other future work may be to consider the impact of using fractional contributions for passwords within a certain distance, where the fraction is either a function of the Hamming distance divided by the password length, or based on some model of textual similarity: for example, in 'leetspeak', `5` is a common substitution for `s`.

## References

1. Bergadano, F., Crispo, B., Ruffo, G.: Proactive password checking with decision trees. Computer and Communications Security 4(1), 67–77 (1997)
2. Bloom, B.H.: Space/time trade-offs in hash coding with allowable errors. Communications of the ACM 13(7), 422–426 (1970)
3. Bonneau, J.: The science of guessing: analyzing an anonymized corpus of 70 million passwords. In: Proc. 2012 IEEE Symposium on Security and Privacy, S&P (2012)
4. Lawrence Carter, J., Wegman, M.N.: Universal classes of hash functions. Journal of Computer and System Sciences 18(2), 143–154 (1979)
5. Cormode, G., Muthukrishnan, S.: An improved data stream summary: the count-min sketch and its applications. Journal of Algorithms 55(1), 58–75 (2005)
6. Cormode, G., Muthukrishnan, S.: Approximating data with the count-min sketch. IEEE Software 29(1), 64–69 (2012)
7. Manber, U., Wu, S.: An algorithm for approximate membership checking with application to password security. Information Processing Letters 50(1), 191–197 (1994)
8. Pliam, J.O.: On the incomparability of entropy and marginal guesswork in brute-force attacks. In: Roy, B., Okamoto, E. (eds.) INDOCRYPT 2000. LNCS, vol. 1977, pp. 67–79. Springer, Heidelberg (2000)
9. Schechter, S., Herley, C., Mitzenmacher, M.: Popularity is everything: A new approach to protecting passwords from statistical guessing attacks. In: Proc. 5th USENIX Conference on Hot Topics in Security, HotSec (2010)
10. Spafford, E.: Preventing weak password choices. Purdue University Computer Science Technical Reports, paper 875, report number 91-028 (1991), `http://docs.lib.purdue.edu/cstech/875`

## A Example Count-Min Sketch Calculation

Fix the table $T$ to be of width $w = 101$ and height $h = 2$.

### A.1 Hash

First we show how the hash values can be calculated for a single password, say `abcd`, under two hash functions.

We encode each character as two-digit integer, say `abcd` $\mapsto x = \langle 1, 2, 3, 4\rangle$, then find the integer representation, $X = 01020304$.

Set a common prime $p = 3571$, and for each of the two hash functions $\mathsf{hash}_k$, choose the parameters $a_k$ and $b_k$ at random modulo $p$; for example, $a = [1151, 2111]$ and $b = [941, 1433]$.

The hashes of `abcd` are as follows:

$$\mathsf{hash}_1(X) = ((1151 \cdot 01020304 + 2111) \mod 3571) \mod 101 = 20$$
$$\mathsf{hash}_2(X) = ((941 \cdot 01020304 + 1433) \mod 3571) \mod 101 = 83\ .$$

The vector-wise hash is thus $\mathsf{hash}(\texttt{abcd}, a, b) = \langle 20, 83\rangle$.

## A.2 Update

We now show how to update the table $T$ to record the use of a password, first updating just the entries for the password itself, then the entries for passwords within Hamming distance 2.

Suppose the table $T$ is currently as follows:

| column | | 19 | 20 | 21 | | 37 | 38 | 39 | | 82 | 83 | 84 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| row 1 | ⋯ | 0 | 0 | 0 | ⋯ | 0 | 0 | 0 | ⋯ | 0 | 0 | 0 | ⋯ |
| row 2 | ⋯ | 0 | 0 | 0 | ⋯ | 0 | 0 | 0 | ⋯ | 0 | 0 | 0 | ⋯ |

*Update at distance 0.* Now, suppose we call $\mathsf{update}(T, x = \texttt{abcd}, a, b, c = 1) \mapsto T'$, which is meant to increment (since $c = 1$) the use of the password `abcd`. Assume $a$ and $b$ are as in the previous subsection. Then we have that $\mathsf{hash}(\texttt{abcd}) = \langle 20, 83\rangle$. Thus, we increment the 20th entry of row 1 and the 83rd entry of row 2, to obtain $T'$:

| column | | 19 | 20 | 21 | | 37 | 38 | 39 | | 82 | 83 | 84 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| row 1 | ⋯ | 0 | **1** | 0 | ⋯ | 0 | 0 | 0 | ⋯ | 0 | 0 | 0 | ⋯ |
| row 2 | ⋯ | 0 | 0 | 0 | ⋯ | 0 | 0 | 0 | ⋯ | 0 | **1** | 0 | ⋯ |

*Update at distance 1.* Next, we increment all four wildcard passwords within distance 1 of `abcd`, namely `*bcd`, `a*cd`, `ab*d`, and `abc*`. This is done by computing the corresponding hashes

$$\mathsf{hash}(\texttt{*bcd}) = \langle 38, 17\rangle \qquad \mathsf{hash}(\texttt{a*cd}) = \langle 37, 47\rangle$$
$$\mathsf{hash}(\texttt{ab*d}) = \langle 37, 63\rangle \qquad \mathsf{hash}(\texttt{abc*}) = \langle 78, 1\rangle$$

and updating all the corresponding entries of the table accordingly. Notice that a few partial collisions occur, for example, both `ab*d` and `a*cd` collide under $\mathsf{hash}_1$, but fortunately do not collide under $\mathsf{hash}_2$.

In our table extract, we only see a few of the 8 updates since not all columns are shown (though we imagine all the updates are applied):

| column | | 19 | 20 | 21 | | 37 | 38 | 39 | | 82 | 83 | 84 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| row 1 | ··· | 0 | 1 | 0 | ··· | **2** | **1** | 0 | ··· | 0 | 0 | 0 | ··· |
| row 2 | ··· | 0 | 0 | 0 | ··· | 0 | 0 | 0 | ··· | 0 | 1 | 0 | ··· |

*Update at distance 2.* Finally, we increment all $\binom{4}{2} = 6$ wildcard passwords within distance 2 of `abcd`; namely

$$\begin{aligned} \mathsf{hash}(\texttt{**cd}) &= \langle 55, 82 \rangle & \mathsf{hash}(\texttt{*b*d}) &= \langle 55, 33 \rangle \\ \mathsf{hash}(\texttt{*bc*}) &= \langle 31, 36 \rangle & \mathsf{hash}(\texttt{a**d}) &= \langle 18, 27 \rangle \\ \mathsf{hash}(\texttt{a*c*}) &= \langle 95, 66 \rangle & \mathsf{hash}(\texttt{ab**}) &= \langle 95, 82 \rangle \end{aligned}$$

In our table extract, we again only see a few of the 12 updates:

| column | | 19 | 20 | 21 | | 37 | 38 | 39 | | 82 | 83 | 84 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| row 1 | ··· | 0 | 1 | 0 | ··· | 2 | 1 | 0 | ··· | 0 | 0 | 0 | ··· |
| row 2 | ··· | 0 | 0 | 0 | ··· | 0 | 0 | 0 | ··· | **2** | 1 | 0 | ··· |

## A.3 Estimate

Suppose we now $\mathsf{estimate}(T, x = \texttt{bbcd}, a, b)$ to obtain the estimate $\overline{\mathsf{est}}^1$ of the frequency of passwords at distance 1 of `bbcd`.

*Estimate at distance 0.* First, we need to compute an estimate $\overline{\mathsf{est}}^0$ for the number of times `bbcd` itself has been used. We do this by computing $\mathsf{hash}_k(\texttt{bbcd})$ and retrieving the corresponding cell from row $i$, then taking the minimum. In this case, $\mathsf{hash}(\texttt{bbcd}) = \langle 76, 53 \rangle$. The 76th entry in the first row and the 53rd entry in the second row are both 0, so this yields a (correct) estimate that the exact password `bbcd` has been seen 0 times before.

*Estimate at distance 1.* To compute an estimate $\overline{\mathsf{est}}^1$ for the number of times any password at distance 1 of `bbcd` has been used, we first need to compute estimates for the number of times each of the four wildcard passwords `*bcd`, `b*cd`, `bb*d`, and `bbc*` has been used before.

$$\begin{aligned} \mathsf{hash}(\texttt{*bcd}) &= \langle 38, 17 \rangle & &\Longrightarrow \mathsf{est}_1^1 = \min\{1, 1\} = 1 \\ \mathsf{hash}(\texttt{b*cd}) &= \langle 100, 8 \rangle & &\Longrightarrow \mathsf{est}_2^1 = \min\{0, 0\} = 0 \\ \mathsf{hash}(\texttt{bb*d}) &= \langle 100, 60 \rangle & &\Longrightarrow \mathsf{est}_3^1 = \min\{0, 0\} = 0 \\ \mathsf{hash}(\texttt{bbc*}) &= \langle 76, 63 \rangle & &\Longrightarrow \mathsf{est}_4^1 = \min\{0, 1\} = 0 \end{aligned}$$

Then, we sum these values and subtract the estimate for the number of times the string `bbcd` itself was used:

$$\overline{\mathsf{est}}^1 = \left( \sum_i \mathsf{est}_i^1 \right) - \ell\, \overline{\mathsf{est}}^0 = 1 \ .$$

# A Rational Secret Sharing Protocol with Unconditional Security in the Synchronous Setting

Yang Yu[1,2] and Zhanfei Zhou[1]

[1] State Key Laboratory of Information Security
Institute of Information Engineering, Chinese Academy of Sciences
[2] University of Chinese Academy of Sciences
Beijing 100093, P.R. China
{yyu,zhouzhanfei}@is.ac.cn

**Abstract.** In order to realize unconditionally secure rational secret sharing over a synchronous (non-simultaneous) channel, previous works either rely on the existence of honest players or induce the approximate notion of $\epsilon$-Nash equilibrium. In this paper, we design two rational $t$-out-of-$n$ secret sharing protocols for $t < \lceil \frac{n}{3} \rceil$ and $t < \lceil \frac{n}{2} \rceil$ respectively, which achieve unconditional security and run in the synchronous setting without requiring any honest player. The former protocol is based on the use of verifiable secret sharing, and the latter protocol extends the former one by using the information checking protocol. Moreover, both of our protocols achieve an enhanced notion of *$\mathcal{C}$-resilient strict Nash equilibrium* ($\mathcal{C}$ consists of the coalitions of less than $t$ players), which guarantees that the prescribed strategy is the only best response even for colluding players, and is stronger than $\epsilon$-Nash equilibrium.

**Keywords:** Rational secret sharing, strict Nash equilibrium, coalition, synchronous, unconditionally secure.

## 1 Introduction

### 1.1 Background

Rational secret sharing (RSS) proposed by Halpern and Teague [8] has attracted much attention in recent years. It studies the problem of secret sharing in the game-theoretic model under the assumption that all players are rational. Unlike honest players who stick to the protocol and malicious players who always disrupt the execution of the protocol in cryptography, rational players are selfish and try to maximize their payoffs.

Many classical secret sharing schemes fail in the rational setting, because rational players prefer to keep silent in the reconstruction stage. Halpern and Teague designed the first randomized RSS protocol, where all players are motivated to cooperate, and suggested the notion of Nash equilibrium surviving iterated deletion of weakly dominated strategies as a standard for capturing the stability of a protocol. Some subsequent works [1,11,10,15] using cryptographic techniques needed to weaken this notion, so as to tolerate the small probability of the broken of cryptography, and they achieved an approximate equilibrium ($\epsilon$-Nash equilibrium). Kol and Naor [10] pointed out that the iterated deletion cannot rule out all bad strategies, and suggested the enhanced notion of strict Nash equilibrium, where the equilibrium strategy is strictly better.

In order that all players can recover the secret in the rational setting, many works [8,6,1,11] relied on the assumption of simultaneous communication, which requires that a player cannot receive messages from other players before sending out his message. Since this assumption is too strong, it is necessary to realize rational secret sharing and achieve fairness that all players get the secret over standard (synchronous but non-simultaneous) channels, where players move sequentially. Fuchsbauer et al. [5] and Zhang et al. [14] did so by delaying the signal which identifies the last iteration. Their protocols induced a $(t-1)$*-resilient computational strict Nash equilibrium*. However, they relied on cryptographic primitives for authentication, which may be broken after an exponential number of rounds. As pointed out in [10], this instability makes rational protocols problematic. To avoid this problem, many works [12,15,9] designed unconditionally secure protocols where players have unbounded computing power by special means. Ong et al. [12] gave a fair RSS protocol by considering the mixture of rational players and honest players, but it cannot resist the collusion attack of even two players. Based on the existence of honest players, William K. Moses Jr. and C. Pandu Rangan [9] designed an efficient protocol tolerating coalitions, which induces a strict Nash equilibrium and $\epsilon$-Nash equilibrium with probability $\frac{k+1}{n}$ and $\frac{n-k-1}{n}$ respectively. However, the assumption of honest players makes their model slightly different from the classical RSS model where all players are rational, and limits the realization of these protocols. Thus, Zhang and Liu [15] got rid of honest players. However, they can only achieve $\epsilon$-Nash equilibrium, which cannot ensure the only best response and is weaker than strict Nash equilibrium, so does the protocol of [10]. Moreover, they needed information theoretically secure MACs for detecting deviations, it may be problematic when their scheme is used as a subroutine for rational multiparty computation.

**Problem Statement.** The protocols [12,9,15,10] realized RSS with information-theoretic security over synchronous channels, but they either achieved ($\epsilon$-)Nash equilibrium which is not appealing, or relied on the existence of honest players. *Strict Nash equilibrium* is a much stronger equilibrium notion, but the work [9] cannot always induce it. In this paper, we remove the honest players and design protocols inducing strict Nash equilibrium.

## 1.2 Our Contribution

We first design a protocol with unconditional security for rational $t$-out-of-$n$ secret sharing where $t < \lceil \frac{n}{3} \rceil$ based on verifiable secret sharing, and then extend it to allow $t < \lceil \frac{n}{2} \rceil$ by using the *information checking protocol.* Our protocols do not rely on simultaneous channels. Instead, they only require synchronous broadcast channels and synchronous private channels. Compared with previous constructions [12,9,15], our protocols do not rely on the existence of honest players, and tolerate the set $\mathcal{C}$ of coalitions of size less than $t$. Moreover, they induce an enhanced notion of *$\mathcal{C}$-resilient strict Nash equilibrium*, which is more appealing than ($\epsilon$-)Nash equilibrium. As previous works [15,10], we remove the on-line dealer. However, unlike them, we do not require players to store all the shares to be used during the protocol in the initial stage.

## 2 Model and Definitions

### 2.1 Game Theoretic Model

The execution of a rational protocol is a game $\Gamma$. At each step, a player decides how to move based on all the messages he has received. The visible *history* $h$ records the sequence of all the actions that have happened. The history after which no more choices have to be made is terminal. With each terminal history $h$ we can associate players' utilities, denoted $\boldsymbol{u}(h)$.

**Definition 1.** *The game $\Gamma$ consists of*

- *A finite set $P = \{P_i | i \in \{1, ..., n\}\}$ (the set of* **players***).*
- *For each player $P_i \in P$ a nonempty set $A_i$ (the set of* **actions** *available to $P_i$), let $A = A_1 \times A_2 \times ... \times A_n$ be the set of action profiles.*
- *A set of sequences $H = \{(\boldsymbol{a}^k)_{k=0,...,T} | \boldsymbol{a}^k \in A, T \in \mathbb{N}\}$ that satisfies the following properties.*
  - $\boldsymbol{a}^0 = \varnothing \in H$
  - *If $(\boldsymbol{a}^k)_{k=0,...,K} \in H$ and $L < K$ then $(\boldsymbol{a}^k)_{k=1,...,L} \in H$*
  - *If an infinite sequence $(\boldsymbol{a}^k)_{k=1,...}$ satisfies $(\boldsymbol{a}^k)_{k=1,...,L} \in H$ for every positive integer $L$ then $(\boldsymbol{a}^k)_{k=1,...} \in H$*

  *Each member of $H$ is a history. A history $(\boldsymbol{a}^k)_{k=0,...,K} \in H$ is* **terminal** *if there is no $\boldsymbol{a}^{K+1}$ s.t. $(\boldsymbol{a}^k)_{k=0,...,K+1} \in H$ or if it is infinite. $Z$ denotes the set of terminal histories.*
- *For each player $P_i \in P$, a function $u_i$ (the* **utility function***) assigns to each terminal history a real value, $u_i : Z \longrightarrow \mathbb{R}$.*

Let $\sigma_i : H \backslash Z \longrightarrow A_i$ be the **strategy** of player $P_i$ $(i \in \{1, ..., n\})$, which determines $P_i$'s action after each history. $\boldsymbol{\sigma} = (\sigma_1, ..., \sigma_n)$ denotes players' strategy profile. We define the **outcome** $O(\boldsymbol{\sigma})$ of $\boldsymbol{\sigma}$ to be the terminal history (possibly infinite) $(\boldsymbol{a}^0, ..., \boldsymbol{a}^K) \in Z$ that results when each player follows the precepts of $\boldsymbol{\sigma}$, it holds that for $0 \leq k < K$, $\boldsymbol{\sigma}(\boldsymbol{a}^0, ..., \boldsymbol{a}^k) = \boldsymbol{a}^{k+1}$. We can define the utility function over strategies $u_i(\boldsymbol{\sigma}) = u_i(O(\boldsymbol{\sigma}))$.

To describe the scenario where players form coalitions, we consider the *game $\Gamma$ with respect to coalitions $\mathcal{C}$*. $\mathcal{C} \subset 2^P$ denotes the set of coalitions (subsets of players) that can be formed. It satisfies that if $C \in \mathcal{C}$ and $C' \subset C$ then $C' \in \mathcal{C}$. For a given coalition $C \in \mathcal{C}$, let $A_C = \times_{P_i \in C} A_i$, we can define its strategy $\sigma_C : H \backslash Z \longrightarrow A_C$. $S_C$ denotes the set of strategies of $C$, and $S_{-C} = \times_{P_j \notin C} S_j$. Following [7], we treat the coalition as a whole. Colluding players share all their information, have the same output and a common utility, instead of an independent utility function for each player. Let $u_C : Z \longrightarrow \mathbb{R}$ denote the uniform utility function of coalition $C$, which describes colluding players' common preference over the outcomes. Once a coalition $C$ is formed, it has a uniform utility function $u_C$, instead of $u_i$ for each $P_i \in C$. Let $u_C(\boldsymbol{\sigma}) = u_C(O(\boldsymbol{\sigma}))$. Our definition captures the scenario where one player controls a number of players. We assume that, in our work other than the coalition $C$ all the other players behave individually ($P_i \notin C$ has his own utility $u_i$); there is at most one coalition in a game. Next we define the subgame.

**Definition 2.** *The subgame of the game $\Gamma =< P, A, H, \boldsymbol{u} >$ with respect to coalition $\mathcal{C}$ that follows the history $h$ is the game $\Gamma(h) =< P, A, H|_h, \boldsymbol{u}|_h >$ with respect to coalition $\mathcal{C}$, where $H|_h$ is the set of $h'$ for which $(h, h') \in H$. For each $C \in \mathcal{C}$, $u_C|_h$ is defined by $u_C|_h(h') = u_C(h, h')$.*

*Given a strategy $\sigma_C$ of coalition $C$ and a history $h$ in the game $\Gamma$ with coalition $\mathcal{C}$, denote by $\sigma_C|_h$ the strategy that $\sigma_C$ induces in the subgame $\Gamma(h)$ with respect to coalition $\mathcal{C}$, i.e. $\sigma_C|_h(h') = \sigma_C(h, h')$.*

## 2.2 Utility Assumption

Let $\mu(\boldsymbol{\sigma}) = (out_1, ..., out_n)$ be a tuple such that, $out_i = 1$ if $P_i$ learns the secret when all players follow $\boldsymbol{\sigma}$ and $out_i = 0$ otherwise. For a given coalition $C$, $\mu_C(\boldsymbol{\sigma}) = (out_i)_{P_i \in C}$. $C$ is said to learn the secret and $out_i = 1$ for each $P_i \in C$ if at least one colluding player learns the secret, otherwise $out_i = 0$ ($\forall P_i \in C$). Following [8] we assume that all rational players prefer to get the secret and prefer the fewest of other players who get the secret. We formalize the utility assumption of $C \in \mathcal{C}$ as follows (it is just the utility assumption of a rational player if the size of $C$ is 1):

A1. $u_C(\boldsymbol{\sigma}) = u_C(\boldsymbol{\sigma}')$ if $\mu(\boldsymbol{\sigma}) = \mu(\boldsymbol{\sigma}')$

A2. $u_C(\boldsymbol{\sigma}) > u_C(\boldsymbol{\sigma}')$ if $\mu_i(\boldsymbol{\sigma}) = 1$ and $\mu_i(\boldsymbol{\sigma}') = 0$ for each $P_i \in C$

A3. $u_C(\boldsymbol{\sigma}) > u_C(\boldsymbol{\sigma}')$ if $\mu_C(\boldsymbol{\sigma}) = \mu_C(\boldsymbol{\sigma}')$, for each $P_j \notin C$ $\mu_j(\boldsymbol{\sigma}) \leq \mu_j(\boldsymbol{\sigma}')$ and there exists a $P_k \notin C$ such that $\mu_k(\boldsymbol{\sigma}) < \mu_k(\boldsymbol{\sigma}')$

Among all utilities, we define the following utilities of $C$ [2].

1. $U_C^+$ is the utility of $C$ when only $C$ gets the secret.
2. $U_C$ is the utility of $C$ when all players get the secret.
3. $U_C^-$ is the maximal utility of $C$ when $C$ gets no secret.
4. $U_C^r$ is the utility of $C$ if $C$ tries to guess the secret.

Let $\mathcal{S}$ be the field from which the secret is chosen, $C$ can guess the secret successfully with negligible probability $1/|\mathcal{S}|$, we have $U_C^r = \frac{1}{|\mathcal{S}|}U_C^+ + (1 - \frac{1}{|\mathcal{S}|})U_C^-$. To motivate players to participate, we assume that players cannot benefit from guessing. Based on our assumption it holds that $U_C^+ > U_C > U_C^r > U_C^-$.

## 2.3 Equilibrium

We strengthen the notion of strict Nash equilibrium to allow for deviating coalitions, (a similar *computational version* of strict Nash equilibrium was proposed by Fuchsbauer et al. [5]). We first pay attention to the following phenomenon. In rational secret sharing over synchronous channels, players may not learn the secret at the same time. Rational players who have learned the secret cannot be motivated to follow the protocol. Once enough of them deviate, some players may be cheated and are not able to learn the secret. It may even happen in the protocol achieving strict Nash equilibrium, because the equilibrium can only guarantee that one player's deviation is not preferable, but does not consider the deviation of several players. However few previous works pay attention to it. Since rational players cannot be prevented from deviating after learning the secret, in order to achieve fairness that all players get the secret, it should be guaranteed that all deviations have no bad influence after at least one player outputs. In addition, strict equilibrium requires that all deviations decrease utilities before at least one player outputs.

We consider the terminal history $h_{\boldsymbol{\sigma}} \in Z$ that results when all players follow the prescribed strategy profile $\boldsymbol{\sigma}$. Let $h^*_{\boldsymbol{\sigma}}$ be the shortest truncation of $h_{\boldsymbol{\sigma}}$ such that at least one player gets the output after it, and we call it *$\boldsymbol{\sigma}$-effective history.* $h_C = (\boldsymbol{a}^0, ..., \boldsymbol{a}^K)$ is the shortest truncation of $h_{\boldsymbol{\sigma}}$ such that a given coalition $C \in \mathcal{C}$ can output after $h_C$. We find it useful to avoid distinguishing the strategies of a player which are different after he outputs, so we define the *reduced strategy* of $\sigma_C$ to be the strategy $\sigma'_C$ which satisfies $\sigma'_C(\boldsymbol{a}^0, ..., \boldsymbol{a}^k) = \sigma_C(\boldsymbol{a}^0, ..., \boldsymbol{a}^k)$ for $0 \le k < K$, denoted $\sigma'_C \approx \sigma_C$. To motivate players not to deviate, we require that $\tau_C \not\approx \sigma_C$ is strictly worse than $\sigma_C$ (Definition 3 Item 1), which ensures that a coalition decreases utility if it deviates before outputting. To achieve fairness we require that in $\Gamma(h^*_{\boldsymbol{\sigma}})$ no strategy profile earns a better outcome than the prescribed strategy profile for each coalition. (Definition 3 Item 2). Now we define what it means to ″induce a $\mathcal{C}$-resilient strict Nash equilibrium″ (which is just strict Nash equilibrium when each $C \in \mathcal{C}$ has only one player).

**Definition 3.** *$\boldsymbol{\sigma}$ is the prescribed strategy of a protocol $\Pi$, $h^*_{\boldsymbol{\sigma}}$ is the $\boldsymbol{\sigma}$-effective history. $\Pi$ induces a $\mathcal{C}$-resilient strict Nash equilibrium, if*

1. *$\forall C \in \mathcal{C}$, it holds that $u_C(\sigma_C, \boldsymbol{\sigma}_{-C}) > u_C(\tau_C, \boldsymbol{\sigma}_{-C})$, $\forall \tau_C \in S_C$, $\tau_C \not\approx \sigma_C$.*
2. *$\Pi$ is seen as a game $\Gamma$ with respect to $\mathcal{C}$, in $\Gamma(h^*_{\boldsymbol{\sigma}})$ with respect to $\mathcal{C}$ it holds that $u_C|_{h^*_{\boldsymbol{\sigma}}}(\boldsymbol{\sigma}|_{h^*_{\boldsymbol{\sigma}}}) + \xi \ge u_C|_{h^*_{\boldsymbol{\sigma}}}(\boldsymbol{\tau})$, $\forall C \in \mathcal{C}, \forall \boldsymbol{\tau} \in S$ ($\xi$ is negligible).*

## 3 Verifiable Secret Sharing

In $(t, n)$ threshold secret sharing, the dealer generates shares $s_1, ..., s_n$ based on the secret $s$ and gives $s_i$ to $P_i$ as his share, so that any subset of at least $t$ players can jointly recover $s$. However, in the rational setting, rational players do not trust each other. They may distribute inconsistent shares or forge shares in the reconstruction stage if it is in their interest to do so. Hence, we rely on verifiable secret sharing (VSS) technology to prevent players being cheated. This section gives a VSS scheme (a simplified version of [4]), which will be used as a tool in our RSS protocols for authentication.

**The** $(t, n)$ **VSS protocol** ($t \le \lceil \frac{n}{2} \rceil$): $M$ is a $n \times t$ matrix, $\boldsymbol{v}_i = (1, x_i, ..., x_i^{t-1})$ is the $i$-th row of $M$, $x_1, ..., x_n$ are $n$ distinct points in field $\mathbb{Z}_q^*$.

1. To share a secret $s \in \mathbb{Z}_q$, the dealer chooses a symmetric $t \times t$ matrix $R$ randomly, except that it has $s$ in the upper left corner. He sends $\boldsymbol{u}_i = R \cdot \boldsymbol{v}_i^T$ to $P_i$, the first coordinate of $\boldsymbol{u}_i$ is the share $s_i$ given to $P_i$.
2. $P_i$ computes $\boldsymbol{v}_j \cdot \boldsymbol{u}_i$ and sends it to $P_j$ as his sub-share $s_{ij}$. $P_j$ compares it to $\boldsymbol{v}_i \cdot \boldsymbol{u}_j$, and broadcasts a complaint $(i, j)$ if the values are not equal.

**Definition 4.** *In the $(t, n)$ VSS scheme above, let $S = \{s_1, ..., s_n\}$, $S_i = \{s_{ij} | j \in \{1, ..., n\}\}$ for $i \in \{1, ..., n\}$. We say that $\{s_1, ..., s_n\}$ are t-consistent if:*

1. *For $i \in \{1, ..., n\}$, there exists a polynomial $f_i(x)$ of degree $t - 1$, such that $f_i(x_j) = s_{ij}$, $\forall s_{ij} \in S_i$ and $f_i(0) = s_i$.*
2. *There exists a polynomial $f(x)$ of degree $t-1$, such that $f(x_i) = s_i, \forall s_i \in S$. We say that $f(0)$ is reconstructed from $S$ consistently.*

In this VSS scheme, players can verify the consistency of shares without revealing their shares. If the dealer shares inconsistently, he is sure to be caught since at least one player complains. If the dealer behaves honestly, all the shares determine the secret uniquely. In the reconstruction stage, players broadcast all their shares and sub-shares, so they can detect deviations of less than $t$ forged shares through the verification of consistency. Thereby, we will use this VSS scheme in our two RSS protocols for catching the deviating player who shares inconsistently and detecting forged shares in the reconstruction stage.

Let $view_C$ denote all the messages $C$ receives during the protocol. Lemma 1,2 prove the privacy and the correctness of the VSS scheme. We will show that rational players prefer not to deviate from the VSS scheme in the security analysis of our RSS protocols.

**Lemma 1.** *In the $(t, n)$ VSS scheme, for any subset $C$ of less than t players, it holds that* $\Pr[C \textit{ learn } s | view_C] = \Pr[C \textit{ learn } s]$.

**Proof.** Given $t - 1$ shares $(s'_1, ..., s'_{t-1})$, for every $s' \in \mathbb{Z}_q$, there exists a polynomial $f$ of degree $t - 1$, s.t. $f(0) = s'$, $f(x_b) = s'_b$, $\forall b = \{1, ..., t - 1\}$. It follows that players cannot recover a secret from less than $t$ shares.

Start from the subset $C$ of size $t - 1$, let $C = \{P_1, ..., P_{t-1}\}$. Note that $view_C = (s_1, ..., s_{t-1}, \{s_{j1}, ..., s_{j(t-1)} | j \in \{1, ..., n\}\})$. Players in $C$ cannot recover the unknown share $s_k$ $(k \in \{t, ..., n\})$ because they have only $t - 1$ shares of each $s_k$, not to mention $s$. The subset $C$ of size less than $t-1$ has fewer shares, so it also gets no information about $s$. □

**Lemma 2.** *In the VSS scheme all players are rational. When $< t$ players deviate from the protocol, if no one complains, there is a polynomial of degree $t - 1$ that passes through the interpolation points of the members who behave faithfully.*

**Proof.** Without loss of generality, we assume that $P_1, ..., P_{n-t+1}$ follow the protocol, and we call them honest players. If there is no complaint, then each pair of honest players agrees with the sub-shares they have in common. We can define a symmetric $n \times n$ matrix $S$, which contains all $s_{ij}$ agreed by all honest players ($s_{ij}$ $i, j > n - t + 1$ is undefined), so all the values in $S$ are correct. For $i \leq n - t + 1$, the $i$-th column uniquely defines $s_i$. The liner combination of the first $n-t+1$ rows determines the shares $(s_1, ..., s_n)$, which determine $s$ consistently. Thus, if the dealer passes the verification, all players who behave faithfully are sure to hold consistent shares. □

Let $a$ and $b$ be two secrets shared through the VSS scheme respectively. We now show how to compute $a+b$ and $a \cdot b$. To compute the addition of $a$ and $b$ is simple, we just add the corresponding shares and sub-shares, i.e. $a_i + b_i$ is the share of $a+b$ held by $P_i$. The multiplication of $a$ and $b$ is more complex. Let $a$ be shared through a $t-1$ degree polynomial $f(x)$, the coefficients of which is the first row of matrix $R$ in the VSS scheme. $b$ is assumed to be shared through $g(x)$. The multiplication is done as follows.

1. For $i \in \{1, ..., n\}$, $P_i$ multiplies his shares of $a$ and $b$, $c_i$ denotes the product, then $P_i$ shares $c_i$ through a $(t, n)$ VSS scheme. Players check whether $c_i = a_i \cdot b_i$, we omit the details here, which can be found in [3,13].
2. By the extrapolation theorem, there exist constants $\gamma_1, ..., \gamma_{2t-1}$ such that $a \cdot b = \sum_{j=1}^{2t-1} \gamma_j \cdot c_j$. For $i \in \{1, ..., n\}$, $P_i$ computes $c'_i \leftarrow \sum_{k=1}^{2t-1} \gamma_k \cdot c_{ki}$, $c'_{ji} \leftarrow \sum_{k=1}^{2t-1} \gamma_k \cdot c_{kji}$, which are the share and sub-share of $a \cdot b$.

## 4 Rational Secret Sharing Protocol *Π*: $t \leq \lceil \frac{n}{3} \rceil$

We provide a RSS protocol that is resilient to coalitions of less than $t$ players. Here all players are computationally unbounded. The protocol runs in a sequence of true/false iterations, only the last one is true where players can recover the secret $s$. We do not assume any on-line dealer. Unlike previous works [15,9], to do so, we do not require players to store all the shares to be used during the entire protocol in the initial stage, which is inconvenient. Instead, players generate ″one-time″ shares in each iteration by themselves. Thus, a credible dealer can quit after sharing the secret $s$ in the initial stage.

We give the main idea of our protocol. In each iteration, players generate a parameter $c$ and the shares for $s+c$, which are valid shares for $s$ when $c$ is 0. After that players recover $s+c$ first, and then verify the current iteration status (i.e. whether $c$ is 0 or not), so as to run the protocol over synchronous channels. To keep the secret private in the false iteration, no information about $c$ except whether it is 0 should be revealed. Thus, when verifying the iteration status, players negotiate a non-zero $g$ and calculate $c \cdot g$. If $c \cdot g \neq 0$ players restart a new iteration, otherwise, players can believe $c = 0$ and output the secret. We ensure that the faithful players get as much information as the deviating players, if not more. Now we describe the protocol in details.

**Initialization.** Let $s \in \mathbb{Z}_q^*$ be the secret ($q$ is large enough), the dealer distributes $s$ through a double sharing protocol: when the dealer shares $s$ through $(t, n)$ threshold secret sharing and generates the shares $s_1, ..., s_n$, he also shares each

$s_i$ through $(t, n)$ threshold secret sharing and generates the sub-shares $s_{i1}, ..., s_{in}$, so $P_i$ gets $s_i, s_{1i}, ..., s_{ni}$.

**The Protocol in Iteration $r$.** Each iteration consists of two phases: the new shares generation phase and the reconstruction phase, which run as follows.

**New shares generation phase**:

1. Generating parameter $c$: Each $P_i$ chooses $\widetilde{c}_i \in_R \mathbb{Z}_q$ s.t. $\Pr[\widetilde{c}_i = 0] = \delta_0 > \frac{1}{q}$, $\Pr[\widetilde{c}_i = l] = \frac{1-\delta_0}{q-1}$ $(\forall l \in \mathbb{Z}_q^*)$, and then shares $\widetilde{c}_i$ through the VSS scheme. Let $c = \sum_{i=1}^{n} \widetilde{c}_i$. $P_i$ computes $c_i = \sum_{k=1}^{n} \widetilde{c}_{ki}$, $c_{ji} = \sum_{k=1}^{n} \widetilde{c}_{kji}$.
2. Generating parameter $g$: Each $P_i$ shares $\widetilde{g}_i \in_R \mathbb{Z}_q^*$ through the VSS scheme. Calculate the shares of $g = \prod_{i=1}^{n} \widetilde{g}_i$ through the multiplication of VSS.
3. Computing $s + c$: Generate the shares of $s + c$ by adding the corresponding shares. Each $P_i$ gets $s_i + c_i$ and $s_{ji} + c_{ji}$, $j \in \{1, ..., n\}$.
4. Computing $c \cdot g$: Generate the shares of $c \cdot g$ through the multiplication of VSS. Each $P_i$ gets $(c \cdot g)_i$ and $(c \cdot g)_{ji}$, $j \in \{1, ..., n\}$.
5. Halt the protocol and output a random guess of the secret once there is a complaint.

**Reconstruction phase**:

1. Players broadcast the shares of $s+c$ and recover $s^{(r)} = s+c$. If the shares are inconsistent, players halt the protocol and then output $s^{(r)}$. If $s^{(r)}$ cannot be computed, players guess the secret.
2. Open $c \cdot g$ to each player in private, take $P_i$ as an example:
   - Each $P_j$ sends $(c \cdot g)_j, (c \cdot g)_{1j}, ..., (c \cdot g)_{nj}$ to $P_i$ privately.
   - $P_i$ halts the protocol and outputs $s^{(r)}$ if he receives partial messages or the shares are inconsistent.

   Proceed to the next iteration if $c \cdot g \neq 0$. Terminate the protocol and output $s^{(r)}$ if $c \cdot g = 0$.

**Remark 1.** *We check the iteration status by calculating $c \cdot g$ instead of opening $c$ directly. This is done to prevent the parameter $c$ from leaking, so that the secret $s$ is private in the false iteration.*

**Remark 2.** *Opening $c \cdot g$ in private prevents players who learn the secret earlier from forging shares undetected, so it avoids the acceptance of a forged $c \cdot g$.*

## 4.1 Protocol Analysis

This protocol is *feasible*. As long as all players do not deviate, valid shares for $s + c$ are revealed in the reconstruction stage, so that all players can recover the secret when $c$ is 0. This protocol guarantees the *privacy*. Without loss of generality, we assume $P_1, ..., P_{t-1}$ collude. According to Lemma 1, they cannot learn $s$ or $c$ from their $t - 1$ shares in the new shares generation phase. In the first step of the reconstruction phase it is obvious that no information about $s$ or $c$ can be revealed from the sum of them. In the second step, although $(c \cdot g)_i = \sum_{k=1}^{2t-1} \gamma_k \cdot (c_k \cdot g_k)_i$ ($i \in \{1, ..., n\}$) are published, coalition members cannot compute $(c_j \cdot g_j)_l$ ($j, l \geq t$), since they have no enough sub-shares, and then they cannot compute $c_j \cdot g_j$ ($j \geq t$) from $t-1$ shares. Hence, $c$ is not revealed if $c \neq 0$. We can conclude that, players cannot identify the true iteration before $s + c$ is recovered, and cannot compute $s$ from the sum of $s$ and an unknown $c$ in the false iteration.

When all players follow the protocol, the parameter $c$ equals 0 with probability $\delta = \frac{1}{q}(1 + \frac{(\delta_0 \cdot q-1)^n}{(q-1)^{n-1}})$. Let $C$ be any coalition of size at most $t - 1$ ($|C|$ is the cardinality of $C$), coalition members can compute $c - \sum_{P_i \in C} \widetilde{c}_i$, which equals 0 with probability $\delta_C = \frac{1}{q}(1 + \frac{(\delta_0 \cdot q-1)^{n-|C|}}{(q-1)^{n-1-|C|}})$. When $C$ has $t-1$ players, $\delta_C$ reaches a maximum $\delta_{max} = \frac{1}{q}(1 + \frac{(\delta_0 \cdot q-1)^{n-t+1}}{(q-1)^{n-t}})$. Now we show why our protocol induces a $\mathcal{C}$-resilient strict Nash equilibrium for an appropriate choice of $\delta_0$.

**Theorem 1.** *Let $\mathcal{C}$ consist of the subsets of $< t$ players, the $(t, n)$ RSS protocol $\Pi$ ($t \leq \lceil \frac{n}{3} \rceil$) induces a $\mathcal{C}$-resilient strict Nash equilibrium if $\delta_{max} < \frac{U_C - U_C^-}{U_C^+ - U_C^-}$.*

**Proof.** Let $C$ be any coalition of size at most $t - 1$. All players can learn the secret if they all follow the protocol, in this case $C$ gets utility $U_C$. We first show that $C$ decreases utility by deviating before it outputs the secret. Let $\boldsymbol{\sigma}$ denote the prescribed strategy of the protocol. Assume that the players not in $C$ stick to the strategy profile $\boldsymbol{\sigma}_{-C}$, and coalition members follow the deviating strategy $\tau_C \not\approx \sigma_C$, which requires $C$ to deviate in the $l$-th iteration before learning the secret. Players may deviate by keeping silent or sending forged messages, both of these deviations lead to the same outcome once they are detected. Since the deviation of keeping silent is sure to be detected, for simplicity, we analyze the scenario where players deviate by forging messages as follows.

In the new shares generation phase, there are four possible deviations. (1) A player in $C$ shares inconsistently among non-coalition members. There must be a pair of non-coalition members who disagree with the sub-shares they have in common, even if at most $t-1$ coalition members cheat. As a result the protocol halts and all players output a random guess of the secret. (2) Coalition members declare a complaint when there is no need to. However, the protocol halts and all players guess the secret. (3) $P_i \in C$ shares a forged $c_i \cdot g_i$ (or $\widetilde{g}_{ji} \cdot \widetilde{g}_{ki}$) when computing $c \cdot g$ (or $\widetilde{g}_j \cdot \widetilde{g}_k$). However, he is sure to be caught and the outcome is the same as above. Therefore, the utility that $C$ gets from the above three deviations is $U_C^r < U_C$. (4) $C$ deviates by sharing $g_i = 0$ ($P_i \in C$). However, all players output $s^{(l)} = s + c$ and terminate the protocol since $c \cdot g = 0$, it means that all players output the real secret or the false secret at the same time. Thus, using this type of deviation, $C$ can get the maximum utility $\delta U_C + (1 - \delta) U_C^-$, which is less than $U_C$ independently of $\delta$.

In the reconstruction phase. (1) Coalition members broadcast forged shares when recovering $s + c$. Since at least $n - t + 1 \geq 2t - 1$ players behave faithfully, so the revealed shares are not $t$-consistent and the protocol will halt. However, we can find a subset of at least $2t-1$ players whose shares are consistent, and recover $s^{(l)} = s + c$ from their shares, since at least $t$ players of them behave faithfully. The scenario where $C$ deviates in the process of opening $c \cdot g$ is same, i.e. the protocol halts and $s+c$ has been recovered. Thus, this deviation earns $C$ the maximum utility $\delta U_C + (1-\delta)U_C^- < U_C$. (2) After $s+c$ has been recovered, coalition $C$ quits the protocol and outputs $s+c$. However, all players output the same value, so $C$ gets the maximum utility $\delta U_C + (1-\delta)U_C^- < U_C$. (3) After $s+c$ has been recovered, coalition $C$ quits the protocol and outputs $s + c - \sum_{P_i \in C} \widetilde{c}_i$. If $c - \sum_{P_i \in C} \widetilde{c}_i$ equals 0 (with the probability $\delta_C$), then only $C$ learns the secret and gets utility $U_C^+$, otherwise $C$ gets at most $U_C^-$. Thus, the expected utility of $C$ with this type of deviation is at most $\delta_{max} U_C^+ + (1-\delta_{max})U_C^-$, which is less than $U_C$ if $\delta_{max} < \frac{U_C - U_C^-}{U_C^+ - U_C^-}$. Even if $C$ can compute the same $s + c - \sum_{P_i \in C} \widetilde{c}_i$ in two iterations, they cannot ensure whether $c - \sum_{P_i \in C} \widetilde{c}_i$ equals 0, thus, they will not deviate.

Putting all these analysis together, we have that $\forall\ C \in \mathcal{C}$ and $\forall\ \tau_C \in S_C$, $\tau_C \not\approx \sigma_C$, it holds that $u_C(\sigma_C, \boldsymbol{\sigma}_{-C}) > u_C(\tau_C, \boldsymbol{\sigma}_{-C})$ for an appropriate $\delta_0$.

Next we prove the fairness. When all players follow, $P_1$ is the first player who learns the secret, just after players open $c \cdot g = 0$ to him. Let $h^*_{\boldsymbol{\sigma}}$ be the shortest history after which $P_1$ learns the secret, we consider the scenario in the subgame $\Gamma(h^*_{\boldsymbol{\sigma}})$. If someone is cheated to believe $c \cdot g \neq 0$ then he may not be able to learn the secret. We show that it happens with negligible probability.

Without loss of generality, we assume that in subgame $\Gamma(h^*_{\boldsymbol{\sigma}})$ with respect to $\mathcal{C}$ players follow deviating strategy $\boldsymbol{\tau}$, which requires $m$ players to deviate when it is time to open $c \cdot g$ to $P_i$. Sending no message at all will be caught, so we analyze the scenario where players forge shares here. When $m \leq n-t$, at least $t$ shares of $c \cdot g$ are valid, so $P_i$ receives inconsistent shares and can output the real secret. When $m > n-t$, $P_i$ will be deceived into believing $c \cdot g \neq 0$ if the forged shares can be accepted and are consistent with the valid shares. Since players send shares to $P_i$ in private, no one can see the shares sent by others, so in order to cheat $P_i$ successfully players can only guess. Given $g_1, ..., g_t$ (some of them may be forged), a player can guess $g_{t+1}$ that is consistent with $g_1, ..., g_t$ with probability $\epsilon = 1/q$ ($q$ is large enough that $\epsilon$ is negligible). To forge shares that determine $c \cdot g \neq 0$ consistently, at least $n-t$ deviating players should guess the corresponding shares correctly, which happens with probability $\epsilon^{n-t}$. In addition, each forged share $(c \cdot g)_j$ should be recovered from its sub-shares consistently, of which at least $n-t+1$ should be forged, and it happens with probability $\epsilon^{n-t+1}$. We can get that the probability that players cheat $P_i$ successfully is at most $\epsilon^{n-t} \cdot (\epsilon^{n-t+1})^{n-t+1}$, which is negligible. Thus, in the subgame following $h^*_{\boldsymbol{\sigma}}$ all kinds of deviations lead to the outcome that all players learn the secret except with negligible probability, so that deviating players can only increase their utility by negligible amount. $\forall\ C \in \mathcal{C}$ and $\forall\ \boldsymbol{\tau} \in S|_{h^*_{\boldsymbol{\sigma}}}$ there exists a negligible $\xi$ such that $u_C|_{h^*_{\boldsymbol{\sigma}}}(\boldsymbol{\sigma}|_{h^*_{\boldsymbol{\sigma}}}) + \xi \geq u_C|_{h^*_{\boldsymbol{\sigma}}}(\boldsymbol{\tau})$.

We conclude that $\Pi$ induces a $\mathcal{C}$-resilient strict Nash equilibrium. $\square$

## 5 Rational Secret Sharing Protocol $\Pi'$: $t \leq \lceil \frac{n}{2} \rceil$

We extend the protocol in section 4 to the case where $t \leq \lceil \frac{n}{2} \rceil$. Here we use the information checking protocol (ICP) to provide a check vector for each share, so that players can authenticate the revealed shares by using check vectors over a broadcast channel, not only by checking the consistency of shares.

### 5.1 The Information Checking Protocol

We use the information checking protocol of [13] to carry out the authentication, which does not rely on any cryptographic assumption. We roughly introduce the process. The dealer $D$ holding a secret $s \in \mathbb{Z}_p$ chooses two random numbers $b \neq 0$ and $y$ both in $\mathbb{Z}_p$, and hands to an intermediary $INT$ the pair $(s, y)$. The dealer $D$ computes $s + by = d$, then sends to the recipient $R$ the vector $(b, d)$ which is called *Check Vector*. At a later time, the intermediary passes $s$ on to $R$ and provides the verification information $y$, then $R$ computes $s + by$ and accepts $s$ if $s + by = d$, which means $R$ believes that the value originated with the dealer. This scheme is secure and no information about the secret reveals, we omit the proof of Lemma 3 here, which can be found in [13].

**Lemma 3.** *When the dealer is honest, $INT$ will deceive $R$ with probability $\frac{1}{p-1}$. The receiver has no information about the secret from his check vector.*

### 5.2 Our Construction

Our new construction of RSS is based on the protocol in section 4. Players not only generate shares for $s + c$ through the VSS scheme, but also provide check vectors for shares through ICP. In the reconstruction phase, players need to send out their shares together with the verification information, so that the deviation of less than a half of players can be caught. The same as the protocol in section 4, a credible dealer distributes the secret $s \in \mathbb{Z}_p^*$ through a double sharing protocol during the process of initialization. Now we give a detailed description of the protocol in iteration $r$ as follows.

**Negotiation about parameters**:

1. Generating parameter $c$: Each $P_i$ shares $\widetilde{c_i} \in_R \mathbb{Z}_p$ through the VSS scheme such that $\Pr[\widetilde{c_i} = 0] = \beta_0 > \frac{1}{p}$, $\Pr[\widetilde{c_i} = l] = \frac{1-\beta_0}{p-1}$ $(\forall l \in \mathbb{Z}_p^*)$. Players compute the shares of $c = \sum_{i=1}^{n} \widetilde{c_i}$.
2. Generating parameter $h$: Each $P_i$ shares $\widetilde{h_i} \in \mathbb{Z}_p$ which is chosen uniformly through the VSS scheme. Players compute the shares of $h = \sum_{i=1}^{n} \widetilde{h_i}$.
3. Halt the protocol and guess the secret once there is a complaint.

**Generation of new shares**:

1. Each $P_i$ calculates $s_i + c_i$, $s_{ji} + c_{ji}$, $j \in \{1, ..., n\}$.
2. Each $P_i$ creates check vectors for each $s_{ij} + c_{ij}$ and sends them to players, i.e. $P_i$ hands $y_{ij1}, ..., y_{ijn}$ and check vectors $(b_{i1j}, d_{i1j}), ..., (b_{inj}, d_{inj})$ over to $P_j$, where $(b_{ikj}, d_{ikj})$ is the check vector of $P_j$ for $s_{ik} + c_{ik}$.
3. Calculate the shares of $c \cdot h$ through the multiplication of the VSS scheme.
4. Each $P_i$ creates check vectors for each $h_{ij}$. $P_i$ hands $y'_{ij1}, ..., y'_{ijn}$ over to $P_j$, and hands check vectors $(b'_{i1j}, d'_{i1j}), ..., (b'_{inj}, d'_{inj})$ over to $P_j$.

**Reconstruction**:

1. Reconstruction of $s + c$.
   (a) Each $P_i$ broadcasts his shares $s_i + c_i$, $s_{ji} + c_{ji}$ and $y_{ji1}, ..., y_{jin}$ ($j \in \{1, ..., n\}$).
   (b) Each $P_i$ checks for each $(s_{jk} + c_{jk}, y_{jki})$ whether he accepts $s_{jk} + c_{jk}$ by using his check vector $(b_{jki}, d_{jki})$.
   (c) $P_i$ accepts $s_j + c_j$ if he accepts at least $t$ shares of $s_j + c_j$ and these shares determine $s_j + c_j$ consistently.
   (d) $P_i$ recovers $s^{(r)} = s + c$ if he accepts at least $t$ shares and all these shares are consistent.
   (e) If a player rejects a sub-share or the shares are inconsistent, then players halt the protocol and output $s^{(r)}$.
2. Reconstruction of $c \cdot h$.
   Each $P_i$ broadcasts the shares of $c \cdot h$. If the shares are consistent, players reconstruct $c \cdot h$; otherwise, they halt the protocol and output $s^{(r)}$. If $c \cdot h \neq 0$, players proceed to the next iteration, otherwise, they enter the next step to verify $h$.
3. Checking $h$.
   (a) Each $P_i$ broadcasts $\{h_i, h_{ji}, y'_{ji1}, ..., y'_{jin} | j \in \{1, ..., n\}\}$.
   (b) Each $P_i$ checks for $(h_{jk}, y'_{jki})$ if he accepts $h_{jk}$ by using his check vector $(b'_{jki}, d'_{jki})$. $h_j$ is accepted if all its shares are accepted and determine $h_j$ consistently. Players halt the protocol and output $s^{(r)}$ if someone rejects a share or the shares are inconsistent.
   (c) Players reconstruct $h$ if the shares are consistent and are accepted. If $h = 0$, players proceed to the next iteration, otherwise, they terminate the protocol and output $s^{(r)}$.

**Remark 3.** *Players continue to open $h$ when $c \cdot h = 0$. If $h = 0$, we treat this iteration as a fake one. Although it may happen that $c = 0$, we do not further identification of c. If $h \neq 0$, we can believe that the true iteration arrives.*

**Analysis.** Each $P_i$ can verify whether each $s_{jk} + c_{jk}$ is valid by using check vectors. The cheaters will be caught except with negligible probability. As long as all the players follow the protocol, valid shares of $s + c$ will be broadcasted, so players can recover the real secret in the true iteration. Moreover, since the information checking protocol is secure, so the same as the protocol $\Pi$, no information about the secret is revealed before the protocol ends, and no information about the iteration status is revealed before $s+c$ is recovered.

The probability that $c=0$ is $\beta = \frac{1}{p}(1+\frac{(\beta_0 \cdot p-1)^n}{(p-1)^{n-1}})$ when players stick to the protocol. Let $C$ be any coalition of size at most $t-1$. The probability that $c - \sum_{P_i \in C} \widetilde{c_i} = 0$ is $\beta_C = \frac{1}{p}(1+\frac{(\beta_0 \cdot p-1)^{n-|C|}}{(p-1)^{n-1-|C|}})$. $t-1$ coalition members can increase the probability that $c=0$ to a maximum $\beta_{max} = \frac{1}{p}(1+\frac{(\beta_0 \cdot p-1)^{n-t+1}}{(p-1)^{n-t}})$ by picking $\widetilde{c_i}=0$. Similarly, they may not choose $h_i$ uniformly, but $\Pr[h=0]=\frac{1}{p}$ will not be changed, let $\alpha = \frac{1}{p}$. Next we show that players prefer not to deviate from the protocol, and the change of $\Pr[c=0]$ has no bad influence.

**Theorem 2.** *Let $\mathcal{C}$ consist of subsets of $< t$ players, the $(t,n)$ RSS protocol $\Pi'$ ($t \le \lceil \frac{n}{2} \rceil$) induces a $\mathcal{C}$-resilient strict Nash equilibrium, if $\frac{\beta_{max}}{\alpha(1-\beta_{max})} < \frac{U_C - U_C^-}{U_C^+ - U_C}$.*

**Proof.** Let $C \in \mathcal{C}$ be any coalition of size at most $t-1$. $\boldsymbol{\sigma}$ is the prescribed strategy of the protocol $\Pi'$. It is obvious that $C$ gets utility $U_C$ when all players follow the protocol. Now we analyze the scenario where $C$ sticks to the deviating strategy $\tau_C \not\approx \sigma_C$ as follows, which deviates from $\sigma_C$ in iteration $l$. The remaining rational players are assumed to follow the strategy profile $\boldsymbol{\sigma}_{-C}$.

Firstly, we consider the deviations during the first phase. (1) A member of coalition $C$ shares inconsistently or keeps silent. However he will fail even if all coalition members deviate, since at least one non-coalition members complains. (2) Coalition members complain when there is no need to. In this case the protocol halts at once. Both of these deviations result in the halt of the protocol. The best thing $C$ can do is to guess the secret, which earns at most $U_C^r < U_C$.

Secondly, coalition $C$ forges check vectors for the shares of $s_i + c_i$ or $h_i$ ($P_i \in C$). However, the protocol halts in the reconstruction stage, when some sub-shares are rejected because of the forged check vectors. Fortunately, players can still recover $s+c$ because at least $t$ shares are accepted, so all players get the same output. In this case the maximum expected utility that $C$ gets is $\beta U_C + (1-\beta)U_C^- < U_C$.

Thirdly, when players compute $c \cdot h$, $C$ may cheat others by sharing a forged $c_i' \cdot h_i' \neq c_i \cdot h_i$ ($P_i \in C$). Then non-coalition members recover a forged value $c' \cdot h'$ and only $C$ learns $c \cdot h$, although it happens with negligible probability since we use the multiplication scheme of [13]. (1) If $c \cdot h$ is 0, $C$ will forge a non-zero $c' \cdot h'$, then players proceed to the next iteration. However, coalition members do not know whether $c$ is 0 or not. If $c=0$, they will get $U_C^+$ at most by outputting $s+c$ and deviating (keeping silent or forging values), otherwise, they will get $U_C^-$ at most. $\Pr[c=0|c \cdot h=0] = \frac{\beta}{\alpha+\beta-\alpha\cdot\beta}$, so the expected utility that $C$ gets from deviations is $U_1 = \frac{\beta}{\alpha+\beta-\alpha\cdot\beta} \cdot U_C^+ + \frac{\alpha-\alpha\cdot\beta}{\alpha+\beta-\alpha\cdot\beta} \cdot U_C^-$. $C$ can also output $s+c-\sum_{P_i \in C} \widetilde{c_i}$. As we pointed out above, when $C$ always picks $\widetilde{c_i}=0$, $C$ can get the maximum utility $U_3 = \frac{\beta_{max}}{\alpha+\beta_{max}-\alpha\cdot\beta_{max}} \cdot U_C^+ + \frac{\alpha-\alpha\cdot\beta_{max}}{\alpha+\beta_{max}-\alpha\cdot\beta_{max}} \cdot U_C^-$. (2) If $c \cdot h$ is not 0, $C$ can forge $c' \cdot h' = 0$ with probability $\frac{1}{p}$. In this case, players continue to verify $h$, which is not 0 actually, and then terminate the protocol and output the fake secret $s^{(l)}$. However, $C$ can learn $c$ after $h$ is published and then calculate $s$. In addition, $C$ forges a non-zero $c' \cdot h'$ with probability $1-\frac{1}{p}$. In this case, players proceed to the next iteration. Since $c$ is not 0, coalition members can quit the protocol and output $s+c-\sum_{P_i \in C} \widetilde{c}_i$, which equals $s$ when $c-\sum_{P_i \in C} \widetilde{c}_i$ is 0. $C$ can get utility at most $\beta' U_C^+ + (1-\beta')U_C^-$, $\beta' = \frac{\beta_{max}(1-\alpha)}{1-2\alpha+\alpha\cdot\beta_{max}}$. Therefore, we can get that when $c \cdot h$ is not 0, the maximum expected utility that $C$ gets by cheating during the process of computing $c \cdot h$ is $U_4 = \alpha U_C^+ + (1-\alpha)(\beta' U_C^+ + (1-\beta')U_C^-) < U_3$. Therefore, in this stage $C$ suffers losses by deviating if it holds that:

$$\frac{\beta_{max}}{\alpha+\beta_{max}-\alpha\cdot\beta_{max}} \cdot U_C^+ + \frac{\alpha-\alpha\cdot\beta_{max}}{\alpha+\beta_{max}-\alpha\cdot\beta_{max}} \cdot U_C^- < U_C \tag{1}$$

Fourthly, $C$ keeps silent or forges shares when reconstructing $s+c$, however, the revealed shares are inconsistent. If $C$ is caught, other players can recover the secret, in this case $C$ gets the maximum utility $\beta U_C + (1-\beta)U_C^- < U_C$. If $C$ cheats during carrying out ICP and forges a share undetected, then non-coalition players cannot recover $s+c$, although it happens with negligible probability. In this case, only $C$ gets the secret $s+c$ when the current iteration is true, and the expected utility that $C$ gets is at most $U_2 = \beta U_C^+ + (1-\beta)U_C^- < U_3$. $C$ may output $s+c-\sum_{P_i \in C} \widetilde{c}_i$, which earns it $U_C^+$ when $c-\sum_{P_i \in C} \widetilde{c}_i$ is 0. Thus, $C$ gets the maximum utility $U_5 = \beta_{max} U_C^+ + (1-\beta_{max})U_C^- < U_3$. Both deviations earn $C$ a utility less than $U_C$ if equation 1 is satisfied.

Fifthly, coalition members quit before $s+c$ is recovered, then the protocol halts and $C$ gets utility $U_C^r < U_C$. Coalition members quit after $s+c$ is recovered. If coalition members output the same value $s+c$ as other players, all players get the real secret or the false secret at the same time, which earns $C$ a utility less than $U_C$. If coalition members output $s+c-\sum_{P_i \in C} \widetilde{c}_i$ before verifying $c \cdot h$, they get utility $U_5$ at most. After $c \cdot h$ is published, the maximum expected utility that coalition members get is $\beta' U_C^+ + (1-\beta')U_C^- < U_3$, when $c \cdot h$ is not 0. Thus, the above deviations earn $C$ a utility less than $U_C$ if equation 1 is satisfied.

Finally, if $C$ provides forged verification information or rejects a valid sub-shares in the reconstruction stage, the protocol halts, but all players can output $s+c$. If $C$ forges shares when reconstructing $c \cdot h$ or $h$, the protocol halts because of the inconsistent shares and then all players output $s+c$. As analyzed above, deviating is strictly worse than following.

We can conclude that if $\frac{\beta_{max}}{\alpha-\alpha\cdot\beta_{max}} < \frac{U_C-U_C^-}{U_C^+-U_C}$ (equation 1 is satisfied), it holds that $u_C(\sigma_C, \boldsymbol{\sigma}_{-C}) > u_C(\tau_C, \boldsymbol{\sigma}_{-C})$, $\forall C \in \mathcal{C}$, $\forall \tau_C \not\approx \sigma_C$.

Next we consider the situation in the subgame $\Gamma(h^*_{\boldsymbol{\sigma}})$ with respect to $\mathcal{C}$ ($h^*_{\boldsymbol{\sigma}}$ is the $\boldsymbol{\sigma}$-effective history). Only if 0 is recovered from the revealed shares consistently can some players be deceived. A given coalition $C \in \mathcal{C}$ can recover $h$ after $t-|C|$ non-coalition members broadcast their shares ($1 \le |C| \le t-1$). Thus in this subgame a player $P_c \in C \cap \{P_1, ..., P_{t-1}\}$ may deviate first after he learns the secret, or $P_t$ becomes the first deviating player after the first $t-1$ players reveal shares faithfully. We consider that all players follow the strategy profile $\boldsymbol{\tau}$ that $m$ players deviate by forging shares of $h$. If $m < n-t+1$, the shares cannot be consistent. If $m \geq n-t+1$, some players will be deceived when all the forged shares pass the verification. Actually, a forged share $h'_j$ can pass the verification only if all its sub-shares are accepted and determine $h'_j$ consistently, so there must exist at least $n-t+1$ forged sub-shares $h'_{jk}$. In the ICP, a player can be deceived with probability $\frac{1}{p-1}$, denoted $\varepsilon$. We assume that there are $x$ coalition members among these $m$ deviating players. When $P_j \in C$, all the coalition members know the check vectors held by $P_i$, so they can forge sub-shares accepted by $P_i$. Thus $P_i$ accepts $h'_j$ with probability at most $\varepsilon^{(n-t+1)-x}$, when $m-x$ non-coalition members guess the check vectors. When $P_j \notin C$, only he knows the check vector, so $P_i$ accepts $h'_j$ with probability at most $\varepsilon^{(n-t+1)-1}$. Therefore, we can calculate that $P_i$ can be cheated with probability at most $\eta = (\varepsilon^{(n-t+1)-x})^x(\varepsilon^{n-t})^{m-x}$. In this protocol, at least one player is deceived with probability at most $\eta' = 1-(1-\eta)^{t-1}$. The expected utility that $C$ gets by following the strategy profile $\boldsymbol{\tau}$ is at most $\eta' U_C^+ + (1-\eta')U_C = U_C + \eta'(U_C^+ - U_C)$. Since $\eta'$ is negligible, $\eta'(U_C^+ - U_C)$ is also negligible. We can get that there exists a negligible $\xi$ such that $u_C|_{h^*_{\boldsymbol{\sigma}}}(\boldsymbol{\sigma}|_{h^*_{\boldsymbol{\sigma}}}) + \xi \geq u_C|_{h^*_{\boldsymbol{\sigma}}}(\boldsymbol{\tau})$, $\forall\ C \in \mathcal{C}$, $\forall\ \boldsymbol{\tau} \in S|_{h^*_{\boldsymbol{\sigma}}}$.

Therefore, our protocol induces a $\mathcal{C}$-resilient strict Nash equilibrium. □

## 6 Discussion

Our two rational secret sharing protocols and previous works [12,15,9,10] have some similar features: all of them are secure against coalitions; none of them relies on simultaneous channels; all of them are information-theoretically secure. However, we have advantages over previous works:

– *Equilibrium notion.* Our protocols induce a strong notion of $\mathcal{C}$-resilient strict Nash equilibrium, which ensures that the equilibrium strategy is the only best response. The works [12,15,9,10] achieve ($\epsilon$-)Nash equilibrium.
– *No honest player.* In contrast with the works [12,9], we do not assume any honest player in our two protocols. Hence, our protocols are easier to be realized and is more appropriate to be generalized.

## 7 Conclusion

This paper studies $t$-out-of-$n$ rational secret sharing with unconditional security over synchronous channels. We design two protocols based on verifiable secret sharing, the first one tolerates less than $t < \lceil \frac{n}{3} \rceil$ colluding players, the second one is secure against coalitions of less than $t < \lceil \frac{n}{2} \rceil$ players based on the use of ICP. Both of the two protocols remove the assumption of honest players and induce the enhanced notion of $\mathcal{C}$-resilient strict Nash equilibrium. In addition, compared with the work [15], we do not use MACs for authentication, so our protocols can be used in rational multiparty computation conveniently.

## References

1. Abraham, I., Dolev, D., Gonen, R., Halpern, J.: Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation. In: Proceedings of the Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing, PODC 2006, pp. 53–62. ACM, New York (2006)
2. Asharov, G., Lindell, Y.: Utility dependence in correct and fair rational secret sharing. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 559–576. Springer, Heidelberg (2009)
3. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: STOC 1988: Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing, pp. 1–10. ACM, New York (1988)
4. Cramer, R., Damgård, I., Maurer, U.: General secure multi-party computation from any linear secret-sharing scheme. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 316–334. Springer, Heidelberg (2000)
5. 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)
6. 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)
7. Halpern, J.Y., Pass, R.: A computational game-theoretic framework for cryptography (2009)
8. Halpern, J.Y., Teague, V.: Rational secret sharing and multiparty computation: extended abstract. In: Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing, STOC 2004, pp. 623–632 (2004)
9. Moses Jr., W.K., Pandu Rangan, C.: Rational secret sharing over an asynchronous broadcast channel with information theoretic security. CoRR, abs/1112.4033 (2011)
10. Kol, G., Naor, M.: Games for exchanging information. In: STOC, pp. 423–432 (2008)
11. 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)
12. Ong, S.J., Parkes, D.C., Rosen, A., Vadhan, S.: Fairness with an honest minority and a rational majority. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 36–53. Springer, Heidelberg (2009)
13. Rabin, T., Ben-Or, M.: Verifiable secret sharing and multiparty protocols with honest majority. In: Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing, STOC 1989, pp. 73–85. ACM, New York (1989)
14. Zhang, Y., Tartary, C., Wang, H.: An efficient rational secret sharing scheme based on the chinese remainder theorem. In: Parampalli, U., Hawkes, P. (eds.) ACISP 2011. LNCS, vol. 6812, pp. 259–275. Springer, Heidelberg (2011)
15. Zhang, Z., Liu, M.: Unconditionally secure rational secret sharing in standard communication networks. In: Rhee, K.-H., Nyang, D. (eds.) ICISC 2010. LNCS, vol. 6829, pp. 355–369. Springer, Heidelberg (2011)

# Secret Sharing Schemes with Conversion Protocol to Achieve Short Share-Size and Extendibility to Multiparty Computation

Ryo Kikuchi, Koji Chida, Dai Ikarashi, Koki Hamada, and Katsumi Takahashi

NTT Secure Platform Laboratories
{kikuchi.ryo,chida.koji,ikarashi.dai,hamada.koki,
takahashi.katsumi}@lab.ntt.co.jp

**Abstract.** Secret sharing scheme (SSS) has been extensively studied since SSSs are important not only for secure data storage but also as the fundamental building block for many cryptographic protocols such as multiparty computation (MPC). Although both code efficiency and application of MPC are important for SSSs, it is difficult to satisfy both. There have been many studies about MPC on Shamir's and replicated SSS while their share size is large, and computationally secure SSS and a ramp scheme have a short share size while there have been few studies concerning their MPC. We propose a new computational SSS, and show how to convert shares of our SSS and a ramp SSS to those of multiparty-friendly SSS such as Shamir's and replicated SSS. This enables one to secretly-share data compactly and extend secretly-shared data to MPC if needed.

**Keywords:** secret-sharing scheme, multiparty computation, code efficiency.

## 1 Introduction

Blakley [2] and Shamir [24] independently proposed a secret-sharing scheme (SSS). An SSS has been extensively studied since SSSs are important not only for secure data storage and disaster recovery, but also as the fundamental building block for many cryptographic protocols. In the SSS model, a dealer first divides a *secret* into *shares* and distributes them among parties. After that, a qualified coalition of parties can reconstruct the secret from their shares, and no one else can obtain information about that secret. $(k, n)$-threshold secret-sharing schemes ($(k, n)$-SSSs) are a common class of SSSs. In this class, there are $n$ parties and any coalition that includes $k$ or more parties is qualified.

Coding efficiency is one of the most important measures for evaluating the performance of SSSs. This represents how many times the total amount of shares is larger than that of a secret. For example, the code efficiency of Shamir's $(k, n)$-SSS [24] is $n$ since there are $n$ parties and each share is the same size as the secret. This means that if one wants to secretly-share 2 GB of data to five parties with Shamir's SSS, the total amount of shares is 10 GB. Such a large amount of data is objectionable not only in terms of storage cost but also communication cost between a dealer and parties.

An SSS is applied to many cryptographic protocols. For example, threshold cryptosystems [12] and, fuzzy identity-based encryption for biometrics [23]. A secure multiparty computation (MPC) is especially closely linked to an SSS. MPC on an SSS is conducted such that a secret is preliminary secretly-shared among parties, and computations, such as addition and multiplication, are conducted while keeping a secret-shared form. There have been many studies of MPCs on SSSs about classical theoretical results, such as those by Ben-Or et al. [1] and Chaum et al. [7] and recent practical results such as those by Bogdanov et al. [5] and Damgård et al. [11].

## 1.1 Compatibility of Code Efficiency and Extendibility to Multiparty Computation

Although both code efficiency and application of MPC are important for SSSs, it is difficult to satisfy both simultaneously.

To reduce the amount of share size, variants of SSSs have been proposed. One is an SSS with computational security proposed by Krawczyk et al. [17], denoted as *computational SSS*. A computational SSS is secure against only polynomially bounded adversaries, while Shamir's SSS is secure against unbounded adversaries. Another approach is a ramp scheme independently proposed by Blakley and Meadows [3], and Yamamoto [25]. A ramp scheme allows unqualified coalitions to obtain partial information of a secret according to a number of corrupted parties.

Next we discuss the extendibility of an SSS to MPC. While Cramer et al. [10] showed that MPC can be conducted on a wide class of SSSs called linear SSSs, most practical results of MPC are on a specific SSSs such as Shamir's $(k,n)$-SSS or replicated $(k,n)$-SSS [16], which have certain preferred properties, perfect privacy and homomorphicity. Consequently, an SSS with good code efficiency does not extend to MPC and vice-versa.

## 1.2 Our Contribution

We propose a method for satisfying both good code efficiency and extendibility to MPC. In the dealing and reconstruction phase, an SSS is with a high regard for code efficiency. If one can conduct MPC on an SSS, parties convert shares to those of an multiparty-friendly SSS. More precisely, we propose a new computationally secure SSS, and two conversion protocols which convert a share of our new computational SSS or homomorphic ramp schemes into one of any any SSSs with homomorphic property including Shamir's and replicated SSS. Therefore, This enables one to secretly-share data compactly and extend secretly-shared data to MPC if needed.

We assume that our proposed schemes is used in secure data storage with MPC, which is a real-use model for MPC on an SSS such as a data aggregation of network traffic statistics [6]. There are users and servers. Users store their data in servers through SSS, and MPC is conducted among servers if needed. In this model, it seems that some of the data is stored but never used in MPC. Therefore, the code efficiency should be small while an additional protocol is required when MPC is conducted.

Our conversion protocols are secure against a semi-honest adversary with static corruption. That is, the adversary corrupts some parties upto $k$ in the beginning of the protocol, and follow the protocol but tries to obtain some information about a secret. This model roughly corresponds to the real-world situation in which we assume the protocol implementations are fairly difficult to tamper with, whereas their inputs and outputs could be eavesdropped on. This model can be a reasonable assumption for most practical purposes. In fact, practical result of MPC on an SSS is usually in the same security model.

### 1.3 Related Works

Krawczyk [17] first proposed a computationally secure SSS. This scheme is proceeded as follows. In the dealing phase, a dealer encrypts a secret with a symmetric key encryption, a symmetric key is distributed through an ordinary $(k,n)$-SSS, and the ciphertext is distributed through $(k,n)$-IDA [18]. The coding efficiency of $(k,n)$-IDA is $n/k$, and the share size of a symmetric key is relatively small. Therefore, the code efficiency of Krawczyk's scheme is almost optimal for $(k,n)$-SSS. Resch and Plank [20] modified Krawczyk's SSS using the All-Or-Nothing Transform (AONT) [21]. The modified scheme eliminates the use of the symmetric key due to the AONT: thus, code efficiency is slightly reduced. However, it is not trivial to perform MPC on Krawczyk's SSS or the modified scheme since they have no homomorphisms.

As a different approach to the schemes described above for reducing code efficiency, Cramer et al. [9] proposed *pseudo-random secret sharing* which replaces the shares of random numbers in a replicated additive SSS setting with pseudo-random numbers. Moreover, their scheme converts the replicated shares into the Shamir representation locally. Note that also discussed a negative result for the conversion from Shamir shares to replicated ones locally.

On the other hand, some MPCs based on a ramp scheme have been proposed so far. Franklin and Yung [13] achieved parallel multiplications on MPC setting to increase computational efficiency. In their scheme some secrets are shared by a ramp scheme after being defined as a vector of finite field. Cramer et al. [8] presented an efficient MPC based on a ramp scheme that achieves one-round secure multiplication and arithmetic circuits. Their scheme allows one to convert shares between different types of linear ramp schemes, while at the same time computing a number of linear functions on secret values in the ramp scheme in parallel.

Unlike the previous works based on a ramp scheme described above, our scheme can use existing MPCs by converting share of homomorphic ramp schemes to that of any homomorphic SSS.

Ghodosi et al. [14] proposed a conversion protocol of multiplicative secret sharing into the additive secret sharing with the help of some auxiliary information distributed in a trusted setup phase. Their conversion protocol is not for better code-efficiency but for efficient MPCs under the case of dishonest majority.

## 2 Preliminaries

We review the notations we use in this paper and the primitives used as components of our constructions and their security notions. We consider the case in which there are $n$

parties $P_1, \dots, P_n$ who receive a share, execute a protocol, and are connected by secure channels.

First, we introduce the notations and notions we use. $x \leftarrow y$ means that $x$ is uniformly at random if $y$ is a finite set; otherwise, simply substitute $y$ into $x$. For probabilistic algorithm $\mathsf{A}$, $y \leftarrow \mathsf{A}(x)$ means that $y$ is the output of $\mathsf{A}$ with input $x$ and uniformly picked randomness. If $\mathbb{T}$ is a set, $|\mathbb{T}|$ means the number of elements that belong to $\mathbb{T}$. We say a function $f(x)$ is negligible in $x$ if for all polynomial $g(x)$, $\lim_{x\to\infty} f(x)g(x) = 0$ holds. We denote a probabilistic polynomial time as PPT.

## 2.1 Multiparty Computation against Semi-honest Adversasries

We propose conversion protocols that belong to MPCs. Therefore, we give their security definition.

We assume unconditional/computational security against a semi-honest adversary with static corruption of at most $t$. This means that the adversary can execute unbounded/polynomially bounded computation, must follow a protocol, and corrupt at most $t$ parties only before the protocol is conducted. More technically, we say that a protocol is secure if there is a simulator that simulates the view of corrupted parties from the inputs and outputs of the protocol. We use $\mathbb{U}$ to denote the set of all parties, and $\mathbb{I} = \{P_{i_1}, \dots, P_{i_t}\} \subset \mathbb{U}$ to denote the parties that are corrupted.

Now we give the formal definition of security against a semi-honest adversary with static corruption. Let $\overrightarrow{x} = (x_1, \dots, x_n)$, $\overrightarrow{x}_{\mathbb{I}} = (x_{i_1}, \dots, x_{i_t})$, $f_i(\overrightarrow{x})$ be the $i$-th output of $f(\overrightarrow{x})$, and $f_{\mathbb{I}}(\overrightarrow{x}) = (f_{i_1}(\overrightarrow{x}), \dots, f_{i_t}(\overrightarrow{x}))$. We denote the view of $P_i$ during the execution of protocol $\rho$ on inputs $\overrightarrow{x}$ as $\textsc{View}^{\rho}_{P_i}(\overrightarrow{x}) = (x_i, r_i; \mu_1, \dots, \mu_\ell)$ where $r_i$ is $P_i$'s random tape, and $\mu_j$ is the $j$-th message $P_i$ received in the protocol execution. We also denote the output of $P_i$ as $\textsc{Output}^{\rho}_{P_i}(\overrightarrow{x})$.

We are now ready to define the security notion in the presence of semi-honest adversaries.

**Definition 1 ([15]).** *Let* $f : (\{0,1\}^*)^n \to (\{0,1\}^*)^n$ *be a probabilistic n-ary functionality,* $\rho$ *be a protocol,* $\textsc{View}^{\rho}_{\mathbb{I}}(\overrightarrow{x}) = (\textsc{View}^{\rho}_{P_{i_1}}(\overrightarrow{x}), \dots, \textsc{View}^{\rho}_{P_{i_t}}(\overrightarrow{x}))$*, and* $\textsc{Output}^{\rho}(\overrightarrow{x}) = (\textsc{Output}^{\rho}_{P_1}(\overrightarrow{x}), \dots, \textsc{Output}^{\rho}_{P_n}(\overrightarrow{x}))$.

*We say that* $\rho$ *perfectly/statistically/computationally t-privately computes f if there exists a PPT algorithm* $\mathcal{S}$ *such that for all* $\mathbb{I} \subset \mathbb{U}$ *of cardinality of at most t and all* $\overrightarrow{x}$ *, it holds that*

$$\left\{(\mathcal{S}(\mathbb{I}, \overrightarrow{x}_{\mathbb{I}}, f_{\mathbb{I}}(\overrightarrow{x})), f(\overrightarrow{x}))\right\} \overset{\mathrm{p/s/c}}{\approx} \left\{(\textsc{View}^{\rho}_{\mathbb{I}}(\overrightarrow{x}), \textsc{Output}^{\rho}(\overrightarrow{x}))\right\}$$

*, where* $\overset{\mathrm{p/s/c}}{\approx}$ *means that two joint distributions are perfectly/statistically/computationally indistinguishable.*

## 2.2 Information Dispersal Algorithm

An information dispersal algorithm (IDA) is used for splitting data into some pieces in such a way that one can reconstruct data from a certain set of pieces. A threshold case of IDA, denoted as $(k, n)$-IDA, was proposed by Rabin [18,19]. $(k, n)$-IDA is the same as $(k, n)$-SSS except that data privacy is not considered. Alternatively, $(k, n)$-IDA has optimal code efficiency, $n/k$, where each $k$-coalitions can reconstruct the original data.

We denote a $(k, n)$-IDA as a tuple of two algorithms. One is the data splitting algorithm $\mathsf{IDA.Split}$, which on input $s$ outputs $(a_1, \ldots, a_n)$. The other is the data reconstruction algorithm $\mathsf{IDA.Rec}$, which on input $(a_{i_1}, \ldots, a_{i_k})$ outputs $s$. An $(k, n)$-IDA must satisfy correctness such that for all $s$ and for all $(i_1, \ldots, i_k)$, $\Pr[(a_1, \ldots, a_n) \leftarrow \mathsf{IDA.Split}(s) : s = \mathsf{IDA.Rec}(a_{i_1}, \ldots, a_{i_k})] = 1$ holds. Furthermore, we assume a $(k, n)$-IDA to be *samplable*. This means that when $(a_{i_1}, \ldots, a_{i_{k-1}})$ is given, one can sample $a_{i_k}$ with the distribution that satisfies

$$\Pr[s' = \mathsf{IDA.Rec}(a_{i_1}, \ldots, a_{i_{k-1}}, a_{i_k})] = \frac{1}{\left|\{s' \mid \Pr[s' \leftarrow \mathsf{IDA.Rec}(a_{i_1}, \ldots, a_{i_{k-1}}, \cdot)] \neq 0\}\right|}$$

for all $s' \in \{s' \mid \Pr[s' \leftarrow \mathsf{IDA.Rec}(a_{i_1}, \ldots, a_{i_{k-1}}, \cdot)] \neq 0\}$. Rabin's scheme satisfies this property. This is required when we strictly estimate the security of MPC.

### 2.3 Pseudorandom Generator

Intuitively speaking, a pseudorandom generator (PRG) is a function that takes a short random string and stretches it to a longer string that seems to be random, in the sense that any PPT algorithm cannot distinguish the output string from a truly random string except with negligible probability. We formally define a PRG as follows.

**Definition 2.** *We call a function* $\mathsf{G} : \{0, 1\}^k \rightarrow \{0, 1\}^{\ell(k)}$ *as* $\epsilon$*-secure PRG if for any PPT adversary* $\mathcal{A}$*, it holds that*

$$\left|\Pr[x \leftarrow \{0, 1\}^k : \mathcal{A}(\mathsf{G}(x)) = 1] - \Pr[x' \leftarrow \{0, 1\}^{\ell(k)} : \mathcal{A}(x') = 1]\right| \leq \epsilon(k).$$

## 3 Secret-Sharing Schemes

In this section, we introduce a model, definitions, and security notions for certain variants of SSSs. First, we explain the commonality of certain variants of SSSs we use in this paper.

We focus on a class of SSSs called $(k, n)$-threshold ($(k, n)$-SSS.). This means that the shares are shared by $n$ parties in such a way that any coalition of $k$ or more parties can together reconstruct the secret, but no coalition fewer than $k$ parties can.

Let $[s]_{P_i}$ be a *share* for party $P_i$ where a *secret* is $s \in \mathcal{R}$. In this paper, a secret belongs to a ring $\mathcal{R}$ including not only $\mathbb{Z}/p\mathbb{Z}$, which is common and used in Shamir's SS, but also $\mathbb{Z}_{2^{32}}$, which is used for efficient MPC [4]. Let $\mathbb{Q}$ be a coalition of parties and $[s]_{\mathbb{Q}}$ denote a set of shares $\{[s]_{P_i} \mid P_i \in \mathbb{Q}\}$. We assume that the number of corrupted parties is $t$ such that $t < k$ holds. We say $([s]_{P_1}, \ldots, [s]_{P_n})$ is *uniformly random* if it is uniformly randomly chosen from the set of shares whose secret is $s$. Let $\mathsf{SSS}$ be a secret-sharing scheme and if we classify a share $[s]_{P_i}$ as belonging to which SSSs, we denote is as $[s]_{P_i}^{\mathsf{SSS}}$ where $[s]_{P_i}$ is a share of $\mathsf{SSS}$.

An SSS consists of two algorithms, a share generation algorithm $\mathsf{Share}$ and share reconstruction algorithm $\mathsf{Reconst}$. The share generation algorithm takes a secret $s$ as input and outputs a uniformly random a set of shares $([s]_{P_1}, \ldots, [s]_{P_n})$. The reconstruction algorithm takes $k$ shares and outputs $s$.

The security requirement of $(k, n)$-SSS is that no coalition fewer than $k$ parties can know the information of the secret. This requirement is guaranteed by a security notion called *privacy*. We define perfect privacy as follows:

**Definition 3.** *Let $S_{P_i}$ be the random valuable of $s_{P_i}$ for $1 \leq i \leq n$ and $S$ be the random valuable corresponding to $s$. We say a secret-sharing scheme* $\mathsf{SSS}$ *has perfect privacy if the following holds for all $i_1, \ldots, i_{k-1} \in \{1, \ldots, n\}$.*

$$\Pr[S = s \mid S_{P_{i_1}} = s_{P_{i_1}}, \ldots, S_{P_{i_{k-1}}} = s_{P_{i_{k-1}}}] = \Pr[S = s].$$

Furthermore, we require an SSS to be *simulatable*. Intuitively speaking, simulatability means one can choose a share from a group of shares in such a way that the distribution of a set of shares is the same as that generated in an ordinary way. We formally define simulatability as follows. If an SSS is simulatable, one can derive $[\Diamond]$ from $([s]_{P_{i_1}}, \ldots, [s]_{P_{i_{k-1}}})$ such that the distribution of $[\Diamond]$ is identical to that of $[s]_{P_{i_k}}$ where $([s]_{P_{i_1}}, \ldots, [s]_{P_{i_{k-1}}}, [s]_{P_{i_k}}) \leftarrow \mathsf{Share}(\mathsf{Reconst}([s]_{P_{i_1}}, \ldots, [s]_{P_{i_{k-1}}}, [\Diamond]))$. It is easy to confirm that Shamir's and replicated SSS are simulatable since one simply sets $[\Diamond]$ as the random element in the share space.

## 3.1 Computationally Secure SSS

A computationally secure SSS was first proposed by Krawczyk [17]. This scheme's procedure is as follows. In the dealing phase, a dealer encrypts a secret with a symmetric key encryption, a symmetric key is distributed through an ordinary $(k, n)$-SSS, and the ciphertext is distributed through $(k, n)$-IDA. The coding efficiency of $(k, n)$-IDA is $n/k$, and the share size of a symmetric key is relatively small. Therefore, the code efficiency of Krawczyk's scheme is roughly $n/k$, which is optimal for $(k, n)$-SSS. However, it seems difficult to conduct MPC on Krawczyk's SSS since a secret is encrypted with the symmetric key encryption.

In the computational setting, perfect privacy described in the previous section can no longer be satisfied. Therefore, another definition of privacy is required. We use the *computational privacy* defined in [22]. To separate a computational SSS from other SSSs, we denote it as $\mathsf{Comp}$ consisting of $(\mathsf{Comp.Share}, \mathsf{Comp.Reconst})$. The computational privacy for $(k, n)$-SSS is formally defined through the following game between a challenger $C$ and an adversary $\mathcal{A}$.

1. $\mathcal{A}$ outputs $(s_0, s_1)$ where $|s_0| = |s_1|$.
2. $C$ randomly chooses $b \in \{0, 1\}$ and computes $([s_b]_{P_1}, \ldots, [s_b]_{P_n}) \leftarrow \mathsf{Comp.Share}(s_b)$.
3. $\mathcal{A}$ is permitted to issue queries to a corrupt oracle. If $\mathcal{A}$ issues $i$, $C$ responds with $[s_b]_{P_i}$ until the number of issues is up to $k - 1$. After this, $C$ responds with $\perp$.
4. $\mathcal{A}$ outputs $b' \leftarrow \{0, 1\}$.

$\mathcal{A}$ wins the above game if $b' = b$ and its advantage is defined by $\mathsf{Adv}^{\text{privacy}}_{\mathsf{Comp}}(\mathcal{A}) = 2\Pr[b = b'] - 1$.

**Definition 4.** *We say a* $\mathsf{Comp}$ *has $\epsilon$-computational privacy if $\mathsf{Adv}^{privacy}_{\mathsf{Comp}}(\mathcal{A}) \leq \epsilon$ for any PPT adversary $\mathcal{A}$.*

### 3.2 Ramp Scheme

Ramp schemes are a variant of an SSS, and were independently proposed by Blakley and Meadows [3], and Yamamoto [25]. The advantage of a ramp scheme is its short share size. while keeping information theoretical security.

We focus on threshold and ramp schemes which is parameterized by three integers, $(k, L, n)$. That is, we can reconstruct an $s$ from arbitrary $k$ or more shares but cannot derive any information from $k - L$ or less shares. Furthermore, any $t$ shares in which $k - L < t < k$ leaks partial information of a secret such that the amount of information increases with the number of collected shares. If $L = 1$, a ramp scheme is identical to an ordinary perfect SSS. To separate a ramp scheme from other SSSs, we denote a ramp scheme as $\mathsf{Ramp}$, which consists of $(\mathsf{Ramp.Share}, \mathsf{Ramp.Rec})$.

Ramp schemes may leak partial information of a secret. Therefore, they also cannot have perfect privacy; therefore an other privacy notion is proposed. A privacy notion of ramp schemes is formally defined as follows where $H$ represents entropy.

**Definition 5.** *Let $s$ be a secret and $[s]^{\mathsf{Ramp}}_{P_1}, \ldots, [s]^{\mathsf{Ramp}}_{P_n}$ be a set of shares of $s$. We say $(k, L, n)$-Ramp scheme* $(\mathsf{Ramp.Share}, \mathsf{Ramp.Rec})$ *has privacy if following requirements are satisfied:*

1. *For any coalition $([s]^{\mathsf{Ramp}}_{P_{i_1}}, \ldots, [s]^{\mathsf{Ramp}}_{P_{i_t}})$, where $k \le t$,*
   *$H(s \mid [s]^{\mathsf{Ramp}}_{P_{i_1}}, \ldots, [s]^{\mathsf{Ramp}}_{P_{i_t}}) = 0$ holds.*
2. *For any coalition $([s]^{\mathsf{Ramp}}_{P_{i_1}}, \ldots, [s]^{\mathsf{Ramp}}_{P_{i_t}})$, where $k - L < t < k$,*
   *$H(s \mid [s]^{\mathsf{Ramp}}_{P_{i_1}}, \ldots, [s]^{\mathsf{Ramp}}_{P_{i_t}}) = \frac{k-t}{L} H(s)$ holds.*
3. *For any coalition $([s]^{\mathsf{Ramp}}_{P_{i_1}}, \ldots, [s]^{\mathsf{Ramp}}_{P_{i_t}})$, where $t \le k - L$,*
   *$H(s \mid [s]^{\mathsf{Ramp}}_{P_{i_1}}, \ldots, [s]^{\mathsf{Ramp}}_{P_{i_t}}) = H(s)$ holds.*

### 3.3 (Additively) Homomorphic Secret-Sharing Scheme

Our aim was to first store secrets through an SSS with better code efficiency and convert shares to "multiparty-friendly" shares if one want to conduct MPC. Although we concretely assume that the output of conversion is shares of Shamir's and replicated SSS, we propose a conversion protocol whose output is the shares of a homomorphic SSS, which is a broad class that includes Shamir's and replicated SSS.

Roughly speaking, if an SSS is homomorphic, a party can compute from shares whose secrets are $a_1$ and $a_2$, to a share of $a_1 + a_2$ locally. This property is of great help in MPC since one fundamental operation can be conducted efficiently.

A homomorphic SSS, denote as $\mathsf{Hom} = (\mathsf{Hom.Share}, \mathsf{Hom.Rec})$, formally allows the following property. There exists an operation $\odot$ such that for all $s \in \mathcal{R}$ and $i$, it holds that

$$[s_1 + s_2]^{\mathsf{Hom}}_{P_i} = [s_1]^{\mathsf{Hom}}_{P_i} \odot [s_2]^{\mathsf{Hom}}_{P_i}.$$

### 3.4 Conversion Protocol

A conversion protocol converts a set of shares to another set of shares while preserving a secret.

Different from the share generation and reconstruction algorithm, the conversion protocol is executed among certain parties. Therefore, it belongs to MPC. To prove the security, we define that the functionality of conversion, which is a description of the conversion protocol, should be satisfied. Given a coalition of parties $\mathbb{Q}$, where $\mathbb{Q} = \{P_{i_1}, \ldots, P_{i_k}\}$ and $\mathbb{U}$ is all parties, we define the conversion function $f^{\mathsf{Convert}}_{\mathsf{SSS}_1,\mathsf{SSS}_2}$ as follows.

$f^{\mathsf{Convert}}_{\mathsf{SSS}_1,\mathsf{SSS}_2}$*: On input* $[s]^{\mathsf{SSS}_1}_{P_{i_j}}$ *for each* $P_{i_j} \in \mathbb{Q}$*, it reveals a secret* $s$ *with the reconstruction algorithm of* $\mathsf{SSS}_1$*, and generates a set of shares* $([s]^{\mathsf{SSS}_2}_{P_{i_1}}, \ldots, [s]^{\mathsf{SSS}_2}_{P_{i_k}})$ *with the share generation algorithm of* $\mathsf{SSS}_2$*. Finally, it outputs* $[s]^{\mathsf{SSS}_2}_{P_i}$ *for each* $P_i \in \mathbb{U}$*.*

## 4 Computational Secret Sharing Scheme with Conversion Protocol

In this section we propose a new computational SSS, denoted as $\mathsf{Comp}$, and a conversion protocol for it. The scheme consists of two algorithms and a protocol, the share generation algorithm $\mathsf{Comp.Share}$, share reconstruction algorithm $\mathsf{Comp.Reconst}$, and share conversion protocol $\mathsf{Comp.Convert}$, where a secret belongs to a certain ring $\mathscr{R}$, $\phi$ is a PRG such that $\phi : \mathscr{S} \rightarrow \mathscr{R}$, $\mathsf{SSS} = (\mathsf{SSS.Share}, \mathsf{SSS.Rec})$ and $\mathsf{Hom} = (\mathsf{Hom.Share}, \mathsf{Hom.Rec})$ are $(k, n)$-SSS and $\mathsf{Hom}$ with homomorphicity, and $\mathsf{IDA} = (\mathsf{IDA.Split}, \mathsf{IDA.Rec})$ is the information dispersal algorithm.

Roughly speaking, in $\mathsf{Comp.Share}$ a secret is masked by pseudo-random value. The masked secret is then distributed through $\mathsf{IDA}$, and seeds of PRG are distributed through $\mathsf{SSS}$. Each seed is unknown if $\mathsf{SSS}$ has perfect privacy, and the masked secret is just a ciphertext of one-time pad with pseudo-random value instead of true randomness. Therefore, our proposed scheme is computationally private. The code efficiency of $\mathsf{IDA}$ is $n/k$, and the size of seed $|\mathscr{S}|$ is much smaller than $|\mathscr{R}|$. Therefore, the code efficiency of our scheme is almost $n/k$, which is optimal for $(k, n)$-threshold. Note that if we consider share generation and reconstruction only, just one pseudo-random value is sufficient to prove computational privacy. However, $\mathsf{Comp.Convert}$ requires $k$ pseudo-random value, so we choose $e_i$ for $i \leq k$.

Our proposed scheme is described as follows.

*Share Generation:* On input of a secret $s \in \mathscr{R}$, $\mathsf{Comp.Share}$ outputs a set of shares $([s]^{\mathsf{Comp}}_{P_1}, \ldots, [s]^{\mathsf{Comp}}_{P_2})$ as follows:

1. Choose random seeds $e_1, \ldots, e_k \in \mathscr{S}$.
2. Compute $t_i = \phi(e_i)$ for $i \leq k$.
3. Set $s' = s - \sum_{i \leq k} t_i$.
4. Compute shares of each seed as $([e_i]^{\mathsf{SSS}}_{P_1}, \ldots, [e_i]^{\mathsf{SSS}}_{P_n}) \leftarrow \mathsf{SSS.Share}(e_i)$ for $i \leq k$ and split $s'$ with $\mathsf{IDA}$ as $(a_{P_1}, \ldots, a_{P_n}) \leftarrow \mathsf{IDA.Split}(s')$.
5. Set a share of party $P_i$ as $[s]^{\mathsf{Comp}}_{P_i} = ([e_1]^{\mathsf{SSS}}_{P_i}, \ldots, [e_k]^{\mathsf{SSS}}_{P_i}, a_{P_i})$ for $i \leq k$ and output $([s]^{\mathsf{Comp}}_{P_1}, \ldots, [s]^{\mathsf{Comp}}_{P_n})$.

*Share Reconstruction:* On input of a list of $k$ shares $([s]^{\mathsf{Comp}}_{P_{i_1}}, \ldots, [s]^{\mathsf{Comp}}_{P_{i_k}})$, $\mathsf{Comp.Reconst}$ outputs an $s$ as follows:

1. Parse $[s]^{\mathsf{Comp}}_{P_{i_\ell}}$ to $([e_1]^{\mathsf{SSS}}_{P_{i_\ell}}, \ldots, [e_k]^{\mathsf{SSS}}_{P_{i_\ell}}, a_{P_{i_\ell}})$ for $\ell \leq k$.
2. Reconstruct $e_i \leftarrow \mathsf{SSS.Rec}([e_i]^{\mathsf{SSS}}_{P_{i_1}}, \ldots, [e_i]^{\mathsf{SSS}}_{P_{i_k}})$ for $i \leq k$ and compute $t_i = \phi(e_i)$.
3. Reconstruct $s' \leftarrow \mathsf{IDA.Rec}(a_{P_{i_1}}, \ldots, a_{P_{i_k}})$.
4. Set $s = s' + \sum_{i \leq k} t_i$ and output $s$.

*Share Conversion:* Let parties $P_{i_1}, \ldots, P_{i_k}$ be a coalition of parties and preliminarily determined. Each party $P_i$ has a share $[s]^{\mathsf{Comp}}_{P_i}$. After $\mathsf{Comp.Convert}$, each party $P_i$ has a share of homomorphic SSS $[s]^{\mathsf{Hom}}_{P_i}$. The precise procedure of the protocol is as follows:

1. Each $P_{i_\ell}$ for $\ell \leq k$ parses $[s]^{\mathsf{Comp}}_{P_{i_\ell}}$ to $([e_1]^{\mathsf{SSS}}_{P_{i_\ell}}, \ldots, [e_k]^{\mathsf{SSS}}_{P_{i_\ell}}, a_{P_{i_\ell}})$.
2. Each $P_{i_\ell}$ sends $[e_j]^{\mathsf{SSS}}_{P_{i_\ell}}$ to $P_{i_j}$ for $j \leq k$.
3. Each $P_{i_\ell}$ reconstructs $e_\ell \leftarrow \mathsf{SSS.Rec}([e_\ell]^{\mathsf{SSS}}_{P_{i_1}}, \ldots, [e_\ell]^{\mathsf{SSS}}_{P_{i_k}})$ and computes $t_\ell = \phi(e_\ell)$.
4. Each $P_{i_\ell}$ sends $a_{P_{i_\ell}}$ to $P_{i_1}$.
5. $P_{i_1}$ reconstructs $s' \leftarrow \mathsf{IDA.Rec}(a_{P_{i_1}}, \ldots, a_{P_{i_k}})$, generates $([s']^{\mathsf{Hom}}_{P_1}, \ldots, [s']^{\mathsf{Hom}}_{P_n}) \leftarrow \mathsf{Hom.Share}(s')$ and sends $[s']^{\mathsf{Hom}}_{P_i}$ to $P_j$ for $j \leq n$.
6. Each $P_{i_\ell}$ computes $([t_\ell]^{\mathsf{Hom}}_{P_1}, \ldots, [t_\ell]^{\mathsf{Hom}}_{P_n}) \leftarrow \mathsf{Hom.Share}(t_\ell)$ and sends $[t_\ell]^{\mathsf{Hom}}_{P_j}$ to $P_j$ for $j \leq n$.
7. Each $P_i$ computes $[s]^{\mathsf{Hom}}_{P_i} = [s']^{\mathsf{Hom}}_{P_i} \odot (\bigodot_{i \leq k} [t_\ell]^{\mathsf{Hom}}_{P_i})$.

## 4.1 Security

First, we show that $\mathsf{Comp}$ has computational privacy. We now consider $\mathsf{Comp.Share}$ and $\mathsf{Comp.Reconst}$ and then discuss the security of $\mathsf{Comp.Convert}$.

**Theorem 1.** *Suppose the SSSs* $\mathsf{SSS}$ *and* $\mathsf{Hom}$ *have perfect privacy, and* $\phi : \mathcal{S} \rightarrow \mathcal{R}$ *is a* $\frac{\epsilon}{k}$*-secure pseudorandom generator.* $\mathsf{Comp}$ *then has* $\epsilon$*-computational privacy.*

*Proof.* We first define a series of games between an adversary $\mathcal{A}$ and challengers $C_i$, where $C_i$ corresponds to the $i$-th game. We denote the probability of $\mathcal{A}$ outputting $b'$ such that $b' = b$ in the $i$-th game as $\Pr[G_i^{\mathcal{A}}]$. A series of games is described as follows.

- **Game 0:** $C_0$ runs an original computational privacy game on $\mathsf{Comp}$.
- **From Game 1 to Game k:** A series of games from Game 1 to Game k is almost the same. We denote Game $i$ as a representative of these games.
  $C_i$ is identical to $C_{i-1}$ except that $C_i$ substitutes $t_{i-1}$ by a uniformly random element in $\mathcal{R}$.

Next we estimate the following equation.

$$\mathsf{Adv}^{\text{privacy}}_{\mathsf{SSS}}(\mathcal{A}) = \Pr[G_0^{\mathcal{A}}] = \sum_{1 \leq i \leq k} \left(\Pr[G_{i-1}^{\mathcal{A}}] - \Pr[G_i^{\mathcal{A}}]\right) + \Pr[G_k^{\mathcal{A}}].$$

First, we claim that for $1 \le i \le k$, $\Pr[G^{\mathcal{A}}_{i-1}] - \Pr[G^{\mathcal{A}}_{i}] \le \frac{\epsilon}{k}$ holds. Consider an adversary $\mathcal{B}_i$ that distinguishes a random element in $\mathscr{R}$ from an output of $\phi$ with the use of $\mathcal{A}$. On input of $x^*$ that is either a uniformly random element in $\mathscr{R}$ or pseudo-random value generated by $\phi$, $\mathcal{B}_i$ and $\mathcal{A}$ runs as follows:

1. $\mathcal{A}$ outputs $(s_1, s_2)$ where $|s_1| = |s_2|$ to $\mathcal{B}_i$.
2. $\mathcal{B}_i$ randomly choose $b \in \{0, 1\}$ and computes $([s_b]_{P_1}, \ldots, [s_b]_{P_n})$ as the same as the original algorithm $\mathsf{Share}$ except that $\mathcal{B}_i$ just substitutes $t_i$ as $t_i = x^*$ instead of $t_i = \phi(e_i)$.
3. $\mathcal{A}$ is permitted to issue queries to a corrupt oracle. If $\mathcal{A}$ issues $i$, $\mathcal{B}_i$ responds with $[s_b]_{P_i}$ where $[s_b]_{P_i}$ is modified as step 2 until the number of issues is up to $k - 1$.
4. $\mathcal{A}$ outputs $b' \leftarrow \{0, 1\}$.
5. $\mathcal{B}_i$ outputs 1 if $b' = b$, 0 otherwise.

The above simulation is identical to Game $i$ if $x^*$ is a random element in $\mathscr{R}$, and is identical to Game $(i - 1)$ if $x^*$ is pseudo-random value generated by $\phi$. Therefore, the following equation holds.

$$\begin{aligned}
\Pr[G^{\mathcal{A}}_{i-1}] - \Pr[G^{\mathcal{A}}_{i}] &= \Pr[x \leftarrow \mathscr{S} : \mathcal{B}_i(\phi(x)) = 1] - \Pr[x' \leftarrow \mathscr{R} : \mathcal{B}_i(x') = 1] \\
&\le \left|\Pr[x \leftarrow \mathscr{S} : \mathcal{B}_i(\phi(x)) = 1] - \Pr[x' \leftarrow \mathscr{R} : \mathcal{B}_i(x') = 1]\right| \\
&\le \frac{\epsilon}{k}
\end{aligned}$$

for $1 \le i \le k$. Finally, we claim that $\Pr[G^{\mathcal{A}}_{k}] = 0$ holds. In Game k, $t_i$ for $1 \le i \le k$ is a uniformly random element in $\mathscr{R}$. Therefore, $s'$ is uniformly random and independent of $b$. Other elements are also independent of $b$ from the beginning.

Consequently, the following equation holds.

$$\mathsf{Adv}^{\text{privacy}}_{\mathsf{SSS}}(\mathcal{A}) = \Pr[G^{\mathcal{A}}_{0}] = \sum_{1 \le i \le k} \left(\Pr[G^{\mathcal{A}}_{i-1}] - \Pr[G^{\mathcal{A}}_{i}]\right) + \Pr[G^{\mathcal{A}}_{k}] \le \epsilon.$$

□

Next, we discuss the security of conversion protocol. We consider a semi-honest adversary with static corruption.

**Theorem 2.** *Suppose SSSs* $\mathsf{Hom}$ *and* $\mathsf{SSS}$ *have perfectly privacy,* $\mathsf{Hom}$ *is homomorphic,* $\mathsf{SSS}$ *is simulatable, a PRG* $\phi$ *is* $\epsilon$*-secure where* $\epsilon$ *is negligible, and an IDA* $\mathsf{IDA}$ *is samplable.* $\mathsf{Comp.Convert}$ *then computationally* $(k - 1)$*-privately computes* $f^{Convert}_{\mathsf{Comp,Hom}}$.

Without loss of generality, we consider the case in which $\mathsf{Comp.Convert}$ is conducted by $P_1, \ldots, P_k$, $s'$ is reconstructed by $P_1$, and a set of corrupted parties is $\mathbb{I} = \{P_1, P_2, \ldots, P_{k-1}\}$.

The view of adversaries consists of their inputs $[s]_{\mathbb{I}} = ([e_1]^{\mathsf{SSS}}_{\mathbb{I}}, \ldots, [e_k]^{\mathsf{SSS}}_{\mathbb{I}}, a_{\mathbb{I}})$, random tapes, their outputs $[s]^{\mathsf{Hom}}_{\mathbb{I}}$, and shares sent from $P_k$ where $P_k$ sends $([e_1]^{\mathsf{SSS}}_{P_k}, \ldots, [e_{k-1}]^{\mathsf{SSS}}_{P_k})$, $[t_k]^{\mathsf{Hom}}_{\mathbb{I}}$, and $a_k$ to corrupted parties.

We construct the simulator $\mathcal{S}$ as follows. Inputs and outputs are the same as those of adversaries, and $\mathcal{S}$ selects random tapes uniformly at random. For the simulation of $([e_1]^{\mathsf{SSS}}_{P_k}, \ldots, [e_{k-1}]^{\mathsf{SSS}}_{P_k})$, $\mathcal{S}$ chooses $[\Diamond_{[e_1]^{\mathsf{SSS}}_{\mathbb{I}}}]^{\mathsf{SSS}}, \ldots, [\Diamond_{[e_{k-1}]^{\mathsf{SSS}}_{\mathbb{I}}}]^{\mathsf{SSS}}$ such that $\hat{e}_i$, where

$\hat{e}_i \leftarrow \mathsf{SSS.Rec}([e_i]^{\mathsf{SSS}}_{\mathbb{I}}, [\Diamond_{[e_i]^{\mathsf{SSS}}_{\mathbb{I}}}]^{\mathsf{SSS}})$ is uniformly randomly distributed in $\mathscr{S}$. This can be done thanks to the simulatability of $\mathsf{SSS}$. For the simulation of $[t_k]^{\mathsf{Hom}}_{\mathbb{I}}$, $\mathcal{S}$ randomly chooses $\hat{t}_k \in \mathscr{R}$, generates $([\hat{t}_k]^{\mathsf{Hom}}_{P_1}, \ldots, [\hat{t}_k]^{\mathsf{Hom}}_{P_n}) \leftarrow \mathsf{Hom.Share}(\hat{t}_k)$, and sets $[\hat{t}_k]^{\mathsf{Hom}}_{\mathbb{I}}$ as the simulation of $[t_k]^{\mathsf{Hom}}_{\mathbb{I}}$. For the simulation of $a_k$, $\mathcal{S}$ randomly chooses $\hat{a}_k$ in the possible space such that $\{a_{i_k} \mid \Pr[(b_1, \ldots, b_n) \leftarrow \mathsf{IDA.Split}(\mathsf{IDA.Rec}(a_{i_1}, \ldots, a_{i_{k-1}}, a_{i_k})) : b_{i_\ell} = a_{i_\ell} \text{ for } 1 \le \ell \le n] \neq 0\}$. This random sampling can be done thanks to the samplability.

We consider a series of distributions from $\left\{(\mathcal{S}(\mathbb{I}, \overrightarrow{x}_{\mathbb{I}}, f^{\mathrm{Convert}}_{\mathsf{Comp,Hom}\,\mathbb{I}}(\overrightarrow{x})), f^{\mathrm{Convert}}_{\mathsf{Comp,Hom}}(\overrightarrow{x}))\right\}$ and switches them in such a way that no PPT adversary can distinguish them. We have already written $\mathcal{S}$ such that $\left\{\left(\mathcal{S}(\mathbb{I}, \overrightarrow{x}_{\mathbb{I}}, f^{\mathrm{Convert}}_{\mathsf{Comp,Hom}\,\mathbb{I}}(\overrightarrow{x})), f^{\mathrm{Convert}}_{\mathsf{Comp,Hom}}(\overrightarrow{x})\right)\right\}$ is

$$\left\{\left(([e_1]^{\mathsf{SSS}}_{\mathbb{I}}, \ldots, [e_k]^{\mathsf{SSS}}_{\mathbb{I}}, a_{\mathbb{I}}), ([\Diamond_{[e_1]^{\mathsf{SSS}}_{\mathbb{I}}}]^{\mathsf{SSS}}, \ldots, [\Diamond_{[e_{k-1}]^{\mathsf{SSS}}_{\mathbb{I}}}]^{\mathsf{SSS}}), [\hat{t}_k]^{\mathsf{Hom}}_{\mathbb{I}}, \hat{a}_k, [s]^{\mathsf{Hom}}_{\mathbb{I}}\right), [s]^{\mathsf{Hom}}_{\mathbb{U}}\right\},$$

where $([e_1]^{\mathsf{SSS}}_{\mathbb{I}}, \ldots, [e_k]^{\mathsf{SSS}}_{\mathbb{I}}, a_{\mathbb{I}})$ are input, $([\Diamond_{[e_1]^{\mathsf{SSS}}_{\mathbb{I}}}]^{\mathsf{SSS}}, \ldots, [\Diamond_{[e_{k-1}]^{\mathsf{SSS}}_{\mathbb{I}}}]^{\mathsf{SSS}}), [\hat{t}_k]^{\mathsf{Hom}}_{\mathbb{I}}, \hat{a}_k$ are the simulated values described in a previous paragraph, and $[s]^{\mathsf{Hom}}_{\mathbb{I}}$ is output. We omit random tapes for simplicity. Notice that $[e_k]^{\mathsf{SSS}}_{\mathbb{I}}$ and $[\hat{t}_k]^{\mathsf{Hom}}_{\mathbb{I}}$ are uniformly random and independent of any other values since $\mathsf{SSS}$ and $\mathsf{Hom}$ have a perfect privacy and there are less than $k$ shares in the above joint distribution.

We claim that

$$\begin{aligned}&\left\{\left(([e_1]^{\mathsf{SSS}}_{\mathbb{I}}, \ldots, [e_k]^{\mathsf{SSS}}_{\mathbb{I}}, a_{\mathbb{I}}), ([\Diamond_{[e_1]^{\mathsf{SSS}}_{\mathbb{I}}}]^{\mathsf{SSS}}, \ldots, [\Diamond_{[e_{k-1}]^{\mathsf{SSS}}_{\mathbb{I}}}]^{\mathsf{SSS}}), [\hat{t}_k]^{\mathsf{Hom}}_{\mathbb{I}}, \hat{a}_k, [s]^{\mathsf{Hom}}_{\mathbb{I}}\right), [s]^{\mathsf{Hom}}_{\mathbb{U}}\right\}\\ \overset{\mathrm{p}}{\approx}\ &\left\{\left(([\hat{e}_1]^{\mathsf{SSS}}_{\mathbb{I}}, \ldots, [\hat{e}_{k-1}]^{\mathsf{SSS}}_{\mathbb{I}}, [e_k]^{\mathsf{SSS}}_{\mathbb{I}}, a_{\mathbb{I}}), ([\hat{e}_1]^{\mathsf{SSS}}_{P_k}, \ldots, [\hat{e}_{k-1}]^{\mathsf{SSS}}_{P_k}), [\hat{t}_k]^{\mathsf{Hom}}_{\mathbb{I}}, \hat{a}_k, [s]^{\mathsf{Hom}}_{\mathbb{I}}\right), [s]^{\mathsf{Hom}}_{\mathbb{U}}\right\},\end{aligned}$$

where $\hat{e}_i = \mathsf{SSS.Rec}([e_i]^{\mathsf{SSS}}_{\mathbb{I}}, [\Diamond_{[e_i]^{\mathsf{SSS}}_{\mathbb{I}}}]^{\mathsf{SSS}})$, for $1 \le i \le k-1$ since $[\Diamond_{[e_i]^{\mathsf{SSS}}_{\mathbb{I}}}]^{\mathsf{SSS}}$ is generated with simulatability that guarantees the equality of distributions between honestly generated shares and simulated shares,

We next claim that

$$\begin{aligned}&\left\{\left(([\hat{e}_1]^{\mathsf{SSS}}_{\mathbb{I}}, \ldots, [\hat{e}_{k-1}]^{\mathsf{SSS}}_{\mathbb{I}}, [e_k]^{\mathsf{SSS}}_{\mathbb{I}}, a_{\mathbb{I}}), ([\hat{e}_1]^{\mathsf{SSS}}_{P_k}, \ldots, [\hat{e}_{k-1}]^{\mathsf{SSS}}_{P_k}), [\hat{t}_k]^{\mathsf{Hom}}_{\mathbb{I}}, \hat{a}_k, [s]^{\mathsf{Hom}}_{\mathbb{I}}\right), [s]^{\mathsf{Hom}}_{\mathbb{U}}\right\}\\ \overset{\mathrm{p}}{\approx}\ &\left\{\left(([\hat{e}_1]^{\mathsf{SSS}}_{\mathbb{I}}, \ldots, [\hat{e}_{k-1}]^{\mathsf{SSS}}_{\mathbb{I}}, [e_k]^{\mathsf{SSS}}_{\mathbb{I}}, a_{\mathbb{I}}), ([\hat{e}_1]^{\mathsf{SSS}}_{P_k}, \ldots, [\hat{e}_{k-1}]^{\mathsf{SSS}}_{P_k}), [\hat{t}'_k]^{\mathsf{Hom}}_{\mathbb{I}}, \hat{a}_k, [s]^{\mathsf{Hom}}_{\mathbb{I}}\right), [s]^{\mathsf{Hom}}_{\mathbb{U}}\right\},\end{aligned}$$

where $\hat{s}' = \mathsf{IDA.Rec}(a_{\mathbb{I}}, \hat{a}_{\mathbb{I}})$ and $\hat{t}'_k = (s - \hat{s}') - \sum_{1 \le i \le k-1} \phi(\hat{e}_i)$ since $\mathsf{Hom}$ has perfect privacy.

We next claim that

$$\begin{aligned}&\left\{\left(([\hat{e}_1]^{\mathsf{SSS}}_{\mathbb{I}}, \ldots, [\hat{e}_{k-1}]^{\mathsf{SSS}}_{\mathbb{I}}, [e_k]^{\mathsf{SSS}}_{\mathbb{I}}, a_{\mathbb{I}}), ([\hat{e}_1]^{\mathsf{SSS}}_{P_k}, \ldots, [\hat{e}_{k-1}]^{\mathsf{SSS}}_{P_k}), [\hat{t}'_k]^{\mathsf{Hom}}_{\mathbb{I}}, \hat{a}_k, [s]^{\mathsf{Hom}}_{\mathbb{I}}\right), [s]^{\mathsf{Hom}}_{\mathbb{U}}\right\}\\ \overset{\mathrm{p}}{\approx}\ &\left\{\left(([e_1]^{\mathsf{SSS}}_{\mathbb{I}}, \ldots, [e_{k-1}]^{\mathsf{SSS}}_{\mathbb{I}}, [e_k]^{\mathsf{SSS}}_{\mathbb{I}}, a_{\mathbb{I}}), ([e_1]^{\mathsf{SSS}}_{P_k}, \ldots, [e_{k-1}]^{\mathsf{SSS}}_{P_k}), [\tilde{t}_k]^{\mathsf{Hom}}_{\mathbb{I}}, \hat{a}_k, [s]^{\mathsf{Hom}}_{\mathbb{I}}\right), [s]^{\mathsf{Hom}}_{\mathbb{U}}\right\},\end{aligned}$$

where $\tilde{t}_k = \hat{t}'_k + \sum_{1 \le i \le k-1}(\phi(\hat{e}_i) - \phi(e_i))$ since each $e_i$ for $1 \le i \le k-1$ is chosen uniformly at random in $\mathscr{S}$, and $\hat{t}'_k$ changes to $\tilde{t}_k$ in such a way that the distribution of $s$ is unchanged.

We next claims that

$$\left\{\left(([e_1]_{\mathbb{I}}^{\mathsf{SSS}},\ldots,[e_k]_{\mathbb{I}}^{\mathsf{SSS}},a_{\mathbb{I}}),([e_1]_{P_k}^{\mathsf{SSS}},\ldots,[e_{k-1}]_{P_k}^{\mathsf{SSS}}),[\tilde{t}_k]_{\mathbb{I}}^{\mathsf{Hom}},\hat{a}_k,[s]_{\mathbb{I}}^{\mathsf{Hom}}\right),[s]_{\mathbb{U}}^{\mathsf{Hom}}\right\}$$
$$\overset{\mathrm{c}}{\approx}\left\{\left(([e_1]_{\mathbb{I}}^{\mathsf{SSS}},\ldots,[e_k]_{\mathbb{I}}^{\mathsf{SSS}},a_{\mathbb{I}}),([e_1]_{P_k}^{\mathsf{SSS}},\ldots,[e_{k-1}]_{P_k}^{\mathsf{SSS}}),[\grave{t}_k]_{\mathbb{I}}^{\mathsf{Hom}},a_k,[s]_{\mathbb{I}}^{\mathsf{Hom}}\right),[s]_{\mathbb{U}}^{\mathsf{Hom}}\right\},$$

where $\hat{s}' = \mathsf{IDA.Rec}(a_{\mathbb{I}},\hat{a}_k)$, $s' = \mathsf{IDA.Rec}(a_{\mathbb{I}},a_k)$, and $\grave{t}_k = \tilde{t}_k + (\hat{s}' - s')$. We changes $\tilde{t}_k$ to $\grave{t}_k$ in such a way that the distribution of $s$ is unchanged. This switch of the joint distribution is not perfectly indistinguishable. Intuitively speaking, $\hat{a}_k$ is generated with samplability that guarantees the uniformity of $\hat{s}$ in the "possible" space determined by $a_{\mathbb{I}}$. However, an unbounded adversary can determine the range of $\phi$, which may narrow the "possible" space. Therefore, we claim that if there exists an adversary $\mathcal{A}$ that distinguishes these two joint distribution, we can construct an adversary $\mathcal{B}$ that distinguishes a random element in $\mathcal{R}$ from an output of $\phi$ with the use of $\mathcal{A}$. Before we describe the behavior of $\mathcal{B}$, we confirm that $\tilde{t}$ is uniformly at random in $\mathcal{R}$ since $\hat{t}$ is uniformly at random, and $\grave{t}$ is an output of $\phi(e_k)$, where $e_k \leftarrow \mathcal{S}$, since $e_1,\ldots,e_{k-1}$, $s$, $a_1,\ldots,a_k$ have already been determined. On input of $x^*$, which is either a uniformly random element in $\mathcal{R}$ or pseudo-random value generated by $\phi$, $\mathcal{B}$ and $\mathcal{A}$ run as follows:

1. $\mathcal{B}$ randomly chooses $e_1,\ldots,e_k$ and computes $([e_i]_{P_1}^{\mathsf{SSS}},\ldots,[e_i]_{P_n}^{\mathsf{SSS}}) \leftarrow \mathsf{SSS.Share}(e_i)$ for $1 \le i \le k$.
2. $\mathcal{B}$ chooses $s \in \mathcal{R}$ according to some distribution over $\mathcal{R}$. This distribution is unfixed but there exists $\mathcal{B}$ that can choose according to it since someone choose $s$ on his own. Then $\mathcal{B}$ computes $([s]_{P_1}^{\mathsf{Hom}},\ldots,[s]_{P_n}^{\mathsf{Hom}}) \leftarrow \mathsf{Hom.Share}(s)$.
3. $\mathcal{B}$ computes $t_i \leftarrow \phi(e_i)$ for $1 \le i \le k-1$, sets $s' = s - \sum_{1\le i\le k-1} t_i - x^*$, computes $(a_1,\ldots,a_k) \leftarrow \mathsf{IDA.Split}(s')$, and $([x^*]_{P_1}^{\mathsf{Hom}},\ldots,[x^*]_{P_n}^{\mathsf{Hom}}) \leftarrow \mathsf{Hom.Share}(s)$.
4. $\mathcal{B}$ sends
$$\left(\left(([e_1]_{\mathbb{I}}^{\mathsf{SSS}},\ldots,[e_k]_{\mathbb{I}}^{\mathsf{SSS}},a_{\mathbb{I}}),([e_1]_{P_k}^{\mathsf{SSS}},\ldots,[e_{k-1}]_{P_k}^{\mathsf{SSS}}),[x^*]_{\mathbb{I}}^{\mathsf{Hom}},a_k,[s]_{\mathbb{I}}^{\mathsf{Hom}}\right),[s]_{\mathbb{U}}^{\mathsf{Hom}}\right)$$
to $\mathcal{A}$.
5. If $\mathcal{A}$ outputs $b'$, then $\mathcal{B}$ also outputs $b'$.

In the above simulation, the view of $\mathcal{A}$ is sampled either from

$$\left\{\left(([e_1]_{\mathbb{I}}^{\mathsf{SSS}},\ldots,[e_k]_{\mathbb{I}}^{\mathsf{SSS}},a_{\mathbb{I}}),([e_1]_{P_k}^{\mathsf{SSS}},\ldots,[e_{k-1}]_{P_k}^{\mathsf{SSS}}),[\tilde{t}_k]_{\mathbb{I}}^{\mathsf{Hom}},\hat{a}_k,[s]_{\mathbb{I}}^{\mathsf{Hom}}\right),[s]_{\mathbb{U}}^{\mathsf{Hom}}\right\}$$

if $x^*$ is uniformly at random in $\mathcal{R}$, or from

$$\left\{\left(([e_1]_{\mathbb{I}}^{\mathsf{SSS}},\ldots,[e_k]_{\mathbb{I}}^{\mathsf{SSS}},a_{\mathbb{I}}),([e_1]_{P_k}^{\mathsf{SSS}},\ldots,[e_{k-1}]_{P_k}^{\mathsf{SSS}}),[\grave{t}_k]_{\mathbb{I}}^{\mathsf{Hom}},a_k,[s]_{\mathbb{I}}^{\mathsf{Hom}}\right),[s]_{\mathbb{U}}^{\mathsf{Hom}}\right\}$$

if $x^*$ is pseudo-random value of $\phi$. Therefore, the advantage of $\mathcal{B}$ is equal to that of $\mathcal{A}$, which contradicts that $\phi$ is an $\epsilon$-secure PRG where $\epsilon$ is negligible.

Finally we claim that

$$\left\{\left(([e_1]_{\mathbb{I}}^{\mathsf{SSS}},\ldots,[e_k]_{\mathbb{I}}^{\mathsf{SSS}},a_{\mathbb{I}}),([e_1]_{P_k}^{\mathsf{SSS}},\ldots,[e_{k-1}]_{P_k}^{\mathsf{SSS}}),[\grave{t}_k]_{\mathbb{I}}^{\mathsf{Hom}},a_k,[s]_{\mathbb{I}}^{\mathsf{Hom}}\right),[s]_{\mathbb{U}}^{\mathsf{Hom}}\right\}$$
$$\overset{\mathrm{p}}{\approx}\left\{\left(([e_1]_{\mathbb{I}}^{\mathsf{SSS}},\ldots,[e_k]_{\mathbb{I}}^{\mathsf{SSS}},a_{\mathbb{I}}),([e_1]_{P_k}^{\mathsf{SSS}},\ldots,[e_{k-1}]_{P_k}^{\mathsf{SSS}}),[t_k]_{\mathbb{I}}^{\mathsf{Hom}},a_k,[s]_{\mathbb{I}}^{\mathsf{Hom}}\right),[s]_{\mathbb{U}}^{\mathsf{Hom}}\right\}$$

since Hom has a perfect privacy. □

## 5 Conversion Protocol for Homomorphic Ramp Schemes

In this section we propose a conversion protocol that convert shares of homomorphic ramp schemes to the one of any homomorphic SSS.

A ramp scheme can be extended to MPC without conversion [8]. However, there is the difficulty to an operation between data which are embedded in different point such as $s_1 \times s_2$ where $s = (s_1, s_2)$. Furthermore, there have been more results in ordinary homomorphic SSSs. Therefore, we propose the conversion protocol.

We propose a protocol for converting $(k, L, n)$-ramp schemes to $(k - L, n)$-homomorphic SSSs and assume an adversary corrupts up to $k - L$ instead of an adversary corrupts up to $k$. This is because If an adversary has already corrupted more than $k - L$, he/she knows a partial information about the secret after the conversion. However, usual MPC protocols do not consider such case. and they may be insecure and cannot be used. This does not meet our aim.

Let $\mathsf{Hom} = (\mathsf{Hom.Share}, \mathsf{Hom.Rec})$ be $(k - L, n)$-SSS and $\mathsf{Ramp}$ and $\mathsf{Hom}$ be homomorphic. Roughly speaking, in the protocol we first reconstruct them in such a way that $k$ parties add a randomness to each secret $s_i$ for $i \leq L$ using homomorphicity, and then re-share $s_i$ with homomorphic SSSs.

Our proposed conversion protocol for ramp schemes, denoted as $\mathsf{Ramp.Convert}$, is described as follows.

*Share Conversion:* Let parties $P_{i_1}, \ldots, P_{i_k}$ be a coalition of parties and preliminarily determined. Each party $P_i$ has a share $[s]_{P_i}^{\mathsf{Ramp}}$ where $s = (s_1, \ldots, s_L) \in \mathcal{R}^L$. After $\mathsf{Ramp.Convert}$, each party $P_i$ has a tuple of shares $([s_1]_{P_i}^{\mathsf{Hom}}, \ldots, [s_L]_{P_i}^{\mathsf{Hom}})$. The precise procedure of the protocol is as follows:

1. Each $P_{i_\ell}$ randomly picks $r_{i_\ell} = (r_{i_\ell,1}, \ldots, r_{i_\ell,L}) \in \mathcal{R}^L$.
2. Each $P_{i_\ell}$ generates $([r_{i_\ell}]_{P_1}^{\mathsf{Ramp}}, \ldots, [r_{i_\ell}]_{P_n}^{\mathsf{Ramp}}) \leftarrow \mathsf{Ramp.Share}(r_{i_\ell})$ and sends $[r_{i_\ell}]_{P_j}^{\mathsf{Ramp}}$ to $P_j$ for $j \leq n$.
3. Each $P_{i_\ell}$ generates $([r_{i_\ell,m}]_{P_1}^{\mathsf{Hom}}, \ldots, [r_{i_\ell,m}]_{P_n}^{\mathsf{Hom}}) \leftarrow \mathsf{Hom.Share}(r_{i_\ell,m})$ for $m \leq L$, and sends $([r_{i_\ell,1}]_{P_j}^{\mathsf{Hom}}, \ldots, [r_{i_\ell,L}]_{P_j}^{\mathsf{Hom}})$ to $P_j$ for $j \leq n$.
4. Each $P_j$ computes $[s']_{P_j}^{\mathsf{Ramp}} = [s]_{P_j}^{\mathsf{Ramp}} - \sum_{\ell \leq k} [r_{i_\ell}]_{P_j}^{\mathsf{Ramp}}$ and sends $[s']_{P_j}^{\mathsf{Ramp}}$ to $P_1$.
5. $P_1$ reconstructs $s' \leftarrow \mathsf{Ramp.Rec}([s']_{P_{i_1}}^{\mathsf{Ramp}}, \ldots, [s']_{P_{i_k}}^{\mathsf{Ramp}})$ and parses $s'$ to $(s'_1, \ldots, s'_L)$.
6. $P_1$ generates $([s'_m]_{P_1}^{\mathsf{Hom}}, \ldots, [s'_m]_{P_n}^{\mathsf{Hom}}) \leftarrow \mathsf{Hom.Share}(s_m)$ for $m \leq L$ and sends $([s'_1]_{P_j}^{\mathsf{Hom}}, \ldots, [s'_L]_{P_j}^{\mathsf{Hom}})$ to $P_j$ for $j \leq n$.
7. Each $P_j$ computes $[s_m]_{P_j}^{\mathsf{Hom}} = [s'_m]_{P_j}^{\mathsf{Hom}} \odot \left( \bigodot_{\ell \leq k} [r_{i_\ell,m}]_{P_j}^{\mathsf{Hom}} \right)$ for $m \leq L$.

### 5.1 Security

Unlike the case of a computational SSS, The conversion of ramp schemes is unconditionally secure since ramp schemes is unconditionally secure too.

**Theorem 3.** *Suppose the SSS* $\mathsf{Hom}$ *has perfect privacy, and* $\mathsf{Hom}$ *and* $\mathsf{Ramp}$ *are homomorphic.* $\mathsf{Ramp.Convert}$ *then perfectly* $(k - L - 1)$*-privately computes* $f_{\mathsf{Ramp,Hom}}^{Convert}$.

Without loss of generality, we consider the case in which Ramp.Convert is conducted by $P_1, \dots, P_k$ and a set of corrupted parties is $\mathbb{I} = \{P_1, P_2, \dots, P_{k-1}\}$.

The view of adversaries consists of their inputs $[s]^{\mathsf{Ramp}}_{\mathbb{I}}$, random tapes, their outputs $[s]^{\mathsf{Hom}}_{\mathbb{I}}$, and shares sent from $P_k$, where $P_k$ sends $[r_k]^{\mathsf{Ramp}}_{\mathbb{I}}$, $[r_{k,i}]^{\mathsf{Hom}}_{\mathbb{I}}$ for $i \le L$, and $[s']^{\mathsf{Ramp}}_{P_k}$ to corrupted parties.

We construct the simulator $\mathcal{S}$ as follows. Inputs and outputs are the same as those of adversaries, and $\mathcal{S}$ selects random tapes uniformly at random. For the simulation of $[r_k]^{\mathsf{Ramp}}_{\mathbb{I}}$ and $[r_{k,i}]^{\mathsf{Hom}}_{\mathbb{I}}$ for $i \le L$, $\mathcal{S}$ does exactly the same as $P_k$. For the simulation of $[s']^{\mathsf{Ramp}}_{P_k}$, $\mathcal{S}$ selects $\hat{s}' \leftarrow \mathscr{R}$, computes $([\hat{s}']^{\mathsf{Ramp}}_{P_1}, \dots, [\hat{s}']^{\mathsf{Ramp}}_{P_n}) \leftarrow \mathsf{Ramp.Share}(\hat{s}')$, and sets $[\hat{s}']^{\mathsf{Ramp}}_{P_k}$ as the simulation of $[s']^{\mathsf{Ramp}}_{P_k}$.

We consider a series of distributions from $\left\{(\mathcal{S}(\mathbb{I}, \overrightarrow{x}_{\mathbb{I}}, f^{\mathsf{Convert}}_{\mathsf{Ramp,Hom}\mathbb{I}}(\overrightarrow{x})), f^{\mathsf{Convert}}_{\mathsf{Ramp,Hom}}(\overrightarrow{x}))\right\}$ and switches them in such a way that the distributions are identical.

We have already written $\mathcal{S}$ such that $\left\{\left(\mathcal{S}(\mathbb{I}, \overrightarrow{x}_{\mathbb{I}}, f^{\mathsf{Convert}}_{\mathsf{Ramp,Hom}\mathbb{I}}(\overrightarrow{x})), f^{\mathsf{Convert}}_{\mathsf{Ramp,Hom}}(\overrightarrow{x})\right)\right\}$ is

$$\left\{\left([s]^{\mathsf{Ramp}}_{\mathbb{I}}, [r_k]^{\mathsf{Ramp}}_{\mathbb{I}}, [r_{k,1}]^{\mathsf{Hom}}_{\mathbb{I}}, \dots, [r_{k,L}]^{\mathsf{Hom}}_{\mathbb{I}}, [\hat{s}]^{\mathsf{Hom}}_{P_k}, [s]^{\mathsf{Hom}}_{\mathbb{I}}\right), [s]^{\mathsf{Hom}}_{\mathbb{U}}\right\},$$

where $[s]^{\mathsf{Ramp}}_{\mathbb{I}}$ is input, $([r_k]^{\mathsf{Ramp}}_{P_1}, \dots, [r_k]^{\mathsf{Ramp}}_{P_{k-1}})$, $([r_{k,1}]^{\mathsf{Hom}}_{P_1}, \dots, [r_{k,1}]^{\mathsf{Hom}}_{P_{k-1}}), \dots, ([r_{k,L}]^{\mathsf{Hom}}_{P_1}, \dots, [r_{k,L}]^{\mathsf{Hom}}_{P_{k-1}})$, $[\hat{s}]^{\mathsf{Hom}}_{P_k}$ are the simulated values described in the previous paragraph, and $[s]^{\mathsf{Hom}}_{\mathbb{I}}$ is output. We omit random tapes for simplicity.

We claim that

$$\left\{\left([s]^{\mathsf{Ramp}}_{\mathbb{I}}, [r_k]^{\mathsf{Ramp}}_{\mathbb{I}}, [r_{k,1}]^{\mathsf{Hom}}_{\mathbb{I}}, \dots, [r_{k,L}]^{\mathsf{Hom}}_{\mathbb{I}}, [\hat{s}']^{\mathsf{Hom}}_{P_k}, [s]^{\mathsf{Hom}}_{\mathbb{I}}\right), [s]^{\mathsf{Hom}}_{\mathbb{U}}\right\}$$
$$\overset{\mathrm{p}}{\approx} \left\{\left([s]^{\mathsf{Ramp}}_{\mathbb{I}}, [\hat{r}_k]^{\mathsf{Ramp}}_{\mathbb{I}}, [\hat{r}_{k,1}]^{\mathsf{Hom}}_{\mathbb{I}}, \dots, [\hat{r}_{k,L}]^{\mathsf{Hom}}_{\mathbb{I}}, [\hat{s}']^{\mathsf{Hom}}_{P_k}, [s]^{\mathsf{Hom}}_{\mathbb{I}}\right), [s]^{\mathsf{Hom}}_{\mathbb{U}}\right\},$$

where $\hat{r}_k = (\hat{r}_{k,1}, \dots, \hat{r}_{k,L})$ satisfies $\sum_{1 \le i \le k-1} r_i + \hat{r}_k = s - \hat{s}'$ since $\hat{r}_k = (\hat{r}_{k,1}, \dots, \hat{r}_{k,L})$ are distributed through a perfectly private SSS and there are less than $k - L$ shares.

We next claim that

$$\left\{\left([s]^{\mathsf{Ramp}}_{\mathbb{I}}, [\hat{r}_k]^{\mathsf{Ramp}}_{\mathbb{I}}, [\hat{r}_{k,1}]^{\mathsf{Hom}}_{\mathbb{I}}, \dots, [\hat{r}_{k,L}]^{\mathsf{Hom}}_{\mathbb{I}}, [\hat{s}']^{\mathsf{Hom}}_{P_k}, [s]^{\mathsf{Hom}}_{\mathbb{I}}\right), [s]^{\mathsf{Hom}}_{\mathbb{U}}\right\}$$
$$\overset{\mathrm{p}}{\approx} \left\{\left([s]^{\mathsf{Ramp}}_{\mathbb{I}}, [\tilde{r}_k]^{\mathsf{Ramp}}_{\mathbb{I}}, [\tilde{r}_{k,1}]^{\mathsf{Hom}}_{\mathbb{I}}, \dots, [\tilde{r}_{k,L}]^{\mathsf{Hom}}_{\mathbb{I}}, [s']^{\mathsf{Hom}}_{P_k}, [s]^{\mathsf{Hom}}_{\mathbb{I}}\right), [s]^{\mathsf{Hom}}_{\mathbb{U}}\right\},$$

where $\tilde{r}_k = (\tilde{r}_{k,1}, \dots, \tilde{r}_{k,L})$ satisfies $\tilde{r}_k = (\hat{s}' - s') + \hat{r}_k$ due to the same reason described in the previous paragraph.

We finally claim that

$$\left\{\left([s]^{\mathsf{Ramp}}_{\mathbb{I}}, [\tilde{r}_k]^{\mathsf{Ramp}}_{\mathbb{I}}, [\tilde{r}_{k,1}]^{\mathsf{Hom}}_{\mathbb{I}}, \dots, [\tilde{r}_{k,L}]^{\mathsf{Hom}}_{\mathbb{I}}, [s']^{\mathsf{Hom}}_{P_k}, [s]^{\mathsf{Hom}}_{\mathbb{I}}\right), [s]^{\mathsf{Hom}}_{\mathbb{U}}\right\}$$
$$\overset{\mathrm{p}}{\approx} \left\{\left([s]^{\mathsf{Ramp}}_{\mathbb{I}}, [r_k]^{\mathsf{Ramp}}_{\mathbb{I}}, [r_{k,1}]^{\mathsf{Hom}}_{\mathbb{I}}, \dots, [r_{k,L}]^{\mathsf{Hom}}_{\mathbb{I}}, [s']^{\mathsf{Hom}}_{P_k}, [s]^{\mathsf{Hom}}_{\mathbb{I}}\right), [s]^{\mathsf{Hom}}_{\mathbb{U}}\right\}$$

also holds due to the same reason. □

## 6 Conclusion

We propose a new computationally secure SSS and two conversion protocols which convert a share of our new SSS and ramp schemes whose coding efficiency is good to that of any homomorphic threshold SSS including Shamir's SSS and replicated SSS. This enables one to store data efficiently and extend secretly-shared data to MPC if needed. Our proposed scheme and protocols are especially suitable for the secure data storage with MPC.

## References

1. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In: STOC, pp. 1–10 (1988)
2. Blakley, G.R.: Safeguarding cryptographic keys. In: Proceedings of the National Computer Conference, vol. 48, pp. 313–317 (1979)
3. Blakley, G.R., Meadows, C.: Security of ramp schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 242–268. Springer, Heidelberg (1985)
4. Bogdanov, D., Laur, S., Willemson, J.: Sharemind: A framework for fast privacy-preserving computations. In: Jajodia, S., Lopez, J. (eds.) ESORICS 2008. LNCS, vol. 5283, pp. 192–206. Springer, Heidelberg (2008)
5. Bogdanov, D., Niitsoo, M., Toft, T., Willemson, J.: High-performance secure multi-party computation for data mining applications. Int. J. Inf. Sec. 11(6), 403–418 (2012)
6. Burkhart, M., Strasser, M., Many, D., Dimitropoulos, X.A.: Sepia: Privacy-preserving aggregation of multi-domain network events and statistics. In: USENIX Security Symposium, pp. 223–240 (2010)
7. Chaum, D., Crépeau, C., Damgård, I.: Multiparty unconditionally secure protocols (extended abstract). In: STOC, pp. 11–19 (1988)
8. Cramer, R., Damgård, I., de Haan, R.: Atomic secure multi-party multiplication with low communication. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 329–346. Springer, Heidelberg (2007)
9. Cramer, R., Damgård, I., Ishai, Y.: Share conversion, pseudorandom secret-sharing and applications to secure computation. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 342–362. Springer, Heidelberg (2005)
10. Cramer, R., Damgård, I., Maurer, U.M.: General secure multi-party computation from any linear secret-sharing scheme. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 316–334. Springer, Heidelberg (2000)
11. 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)
12. Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 307–315. Springer, Heidelberg (1990)
13. Franklin, M.K., Yung, M.: Communication complexity of secure computation (extended abstract). In: STOC, pp. 699–710 (1992)
14. Ghodosi, H., Pieprzyk, J., Steinfeld, R.: Multi-party computation with conversion of secret sharing. Des. Codes Cryptography 62(3), 259–272 (2012)
15. Goldreich, O.: The Foundations of Cryptography. Basic Applications, vol. 2. Cambridge University Press (2004)
16. Ito, M., Saito, A., Nishizeki, T.: Secret sharing schemes realizing general access structure. In: Proc. of the IEEE Global Telecommunication Conf., Globecom 1987, pp. 99–102 (1987); Journal version: Multiple assignment scheme for sharing secret. J. of Cryptology 6(1), 15–20 (1993)
17. Krawczyk, H.: Secret sharing made short. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 136–146. Springer, Heidelberg (1994)

18. Rabin, M.O.: Efficient dispersal of information for security, load balancing, and fault tolerance. Journal of the ACM 36(2), 335–348 (1989)
19. Rabin, M.O.: Sequences, pp. 406–419. Springer-Verlag New York, Inc., New York (1990)
20. Resch, J.K., Plank, J.S.: Aont-rs: Blending security and performance in dispersed storage systems. In: FAST, pp. 191–202 (2011)
21. Rivest, R.L.: All-or-nothing encryption and the package transform. In: Biham, E. (ed.) FSE 1997. LNCS, vol. 1267, pp. 210–218. Springer, Heidelberg (1997)
22. Rogaway, P., Bellare, M.: Robust computational secret sharing and a unified account of classical secret-sharing goals. In: ACM Conference on Computer and Communications Security, pp. 172–184 (2007)
23. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
24. Shamir, A.: How to share a secret. Communications of the ACM 22(11), 612–613 (1979)
25. Yamamoto, H.: Secret sharing system using (k,l,n) threshold scheme. IECE Trans. J68-A(9), 945–952 (1985) (in Japanese); English translation: Electronics and Communications in Japan, Part I, vol. 69(9), pp. 46–54. Scripta Technica, Inc. (1986)

# Public Key Cryptography for Mobile Cloud

Yuliang Zheng

The University of North Carolina at Charlotte

## Abstract of the Invited Lecture

Mobile cloud is the integration of cloud computing and mobile communication systems to benefit mobile device users. As a mobile device such as a smartphone and a tablet computer has a limited amount of onboard storage, participation in mobile cloud provides the mobile device with access to a virtually unlimited amount of storage on an on-demand basis. In this talk I will focus on a specific technical challenge in building security-enhanced mobile cloud, namely how to encrypt data using public key cryptography in such a way that a sender can recover the data from a ciphertext stored in the cloud without reliance on the recipient of the ciphertext.

As a motivating example, consider a scenario where Alice has an important message to be sent to Bob in a secure manner. She can accomplish this by employing Bob's public key to encrypt the message into a ciphertext, followed by emailing Bob the ciphertext. After sending the encrypted message to Bob, Alice's email system dutifully keeps an identical copy of the ciphertext in her "Sent" folder in the cloud storage. Some time later Alice finds herself in a position where she needs to recover the message from the ciphertext, without access to Bob's decryption key. Obviously Alice would be out of luck if a regular public key encryption scheme is employed.

I will explain how the above problem can be solved by a new type of public key encryption techniques that admit decryption by the sender, without the need to modify existing messaging protocols typified by "sendmail". I will also show two generic methods for converting a regular public key encryption scheme into one that admits decryption by the sender, in such a way that the security of the resultant mechanism can be formally proven.

# Author Index