

**Howon Kim
Dong-Chan Kim (Eds.)**

LNCS 10779

Information Security and Cryptology – ICISC 2017

**20th International Conference
Seoul, South Korea, November 29 – December 1, 2017
Revised Selected Papers**



Springer

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, Lancaster, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Friedemann Mattern

ETH Zurich, Zurich, Switzerland

John C. Mitchell

Stanford University, Stanford, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

C. Pandu Rangan

Indian Institute of Technology Madras, Chennai, India

Bernhard Steffen

TU Dortmund University, Dortmund, Germany

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Gerhard Weikum

Max Planck Institute for Informatics, Saarbrücken, Germany

More information about this series at <http://www.springer.com/series/7410>

Howon Kim · Dong-Chan Kim (Eds.)

Information Security and Cryptology – ICISC 2017

20th International Conference

Seoul, South Korea, November 29 – December 1, 2017

Revised Selected Papers

Editors

Howon Kim
Pusan National University
Busan
Korea (Republic of)

Dong-Chan Kim
Kookmin University
Seoul
Korea (Republic of)

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-319-78555-4 ISBN 978-3-319-78556-1 (eBook)
<https://doi.org/10.1007/978-3-319-78556-1>

Library of Congress Control Number: 2018937384

LNCS Sublibrary: SL4 – Security and Cryptology

© Springer International Publishing AG, part of Springer Nature 2018

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Printed on acid-free paper

This Springer imprint is published by the registered company Springer International Publishing AG
part of Springer Nature
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

ICISC 2017, the 20th International Conference on Information Security and Cryptology, was held in Seoul, Korea, November 29 – December 1, 2017. This year conference was hosted by the KIISC (Korea Institute of Information Security and Cryptology) jointly with the NSR (National Security Research Institute).

The aim of this conference is to provide a international forum for the latest results of research, development, and applications in the field of information security and cryptology. This year we received 70 submissions, and were able to accept 20 papers (acceptance rate of 28%). The review and selection processes were carried out by the PC (Program Committee) members, prominent experts, via the EDAS review system. First, each paper was blind reviewed, by at least three PC members for most cases. Second, for resolving conflicts on the reviewers decisions, the individual review reports were open to all PC members, and detailed interactive discussions on each paper followed.

The conference features one invited talk: Modeling & Simulation Technologies of Intelligent and Connected Vehicles by Jian Wang. We thank the invited speaker for his kind acceptance and presentation.

We would like to thank all authors who submitted their papers to ICISC 2017 and all of PC members. It was a truly experience to work with such talented and hard-working researchers. We also appreciate the external reviewers for assisting the PC members in their particular areas of expertise.

Finally, we would like to thank all attendees for their active participation and the organizing members who nicely manage this conference. We look forward to seeing you again next year ICISC.

November 2017

Howon Kim
Dong-Chan Kim

Organization

General Chair

DongHoon Lee Korea University, South Korea

Organizing Chair

Dong-Guk Han Kookmin University, South Korea

Organizing Committee

Daeseon Choi Kongju National University, South Korea
Seokhie Hong Korea University, South Korea
Junbeom Hur Korea University, South Korea
Jongsung Kim Kookmin University, South Korea
So Jeong Kim National Security Research Institute, South Korea
Teakyoungh Kown Yonsei University, South Korea
Kihyo Nam UMLogics, Co., Ltd., South Korea
Young-Ho Park Sejong Cyber University, South Korea

Program Chairs

Howon Kim Pusan National University, South Korea
Dong-Chan Kim Kookmin University, South Korea

Program Committee

Paolo D' Arco University of Salerno, Italy
Joonsang Baek University of Wollongong, Australia
Lynn M. Batten Deakin University, Australia
Olivier Blazy XLim, Université de Limoges, France
Zhenfu Cao East China Normal University, China
Nilanjan Datta Indian Institute of Technology Kharagpur, India
Keita Emura NICT, Japan
Kazuhide Fukushima KDDI R&D Laboratories Inc., Japan
Johann Groschdl University of Luxembourg, Luxembourg
Dong-Guk Han Kookmin University, South Korea
Martin Hell Lund University, Sweden
Swee-Huay Heng Multimedia University, Malaysia
Deukjo Hong Chonbuk National University, South Korea
Seokhie Hong Korea University, South Korea
Xinyi Huang Fujian Normal University, China

David Jao	University of Waterloo, Canada
Dong Seong Kim	University of Canterbury, New Zealand
Huy Kang Kim	Korea University, South Korea
Jongsung Kim	Kookmin University, South Korea
Shinsaku Kiyomoto	KDDI Research Inc., Japan
Daesung Kwon	NSR, South Korea
Taekyoung Kwon	Yonsei University, South Korea
Alptekin Kp	Koc University, Turkey
Chang Hun Lee	Seoul National University of Science and Technology, South Korea
Hyung Tae Lee	Nanyang Technological University, Singapore
Jong-Hyouk Lee	Sangmyung University, South Korea
Jooyoung Lee	KAIST, South Korea
Kwangsue Lee	Sejong University, South Korea
Moon Sung Lee	University of Luxembourg, Luxembourg
Mun-Kyu Lee	Inha University, South Korea
Hua-Yi Lin	China University of Technology, Taiwan
Joseph K. Liu	Monash University, Australia
Zhe Liu	University of Luxembourg, Luxembourg
Jiqiang Lu	Institute for Infocomm Research, Singapore
Atul Luykx	KU Leuven, Belgium and University of California, Davis, USA
Sjouke Mauw	University of Luxembourg, Luxembourg
Florian Mendel	Infineon Technologies AG, Germany
Atsuko Miyaji	Osaka University/Japan Advanced Institute of Science and Technology, Japan
Yasuyuki Nogami	Graduate School of Natural Science and Technology, Okayama University, Japan
DaeHun Nyang	Inha University, South Korea
Katsuyuki Okeya	Hitachi High-Technologies Corporation, Japan
Jong Hwan Park	Sangmyung University, South Korea
Young-Ho Park	Sejong Cyber University, South Korea
Souradyuti Paul	Indian Institute of Technology, Gandhinagar, India
Pedro Peris-Lopez	Universidad Carlos III de Madrid, Spain
Josef Pieprzyk	Queensland University of Technology, Australia
Bimal Roy	Indian Statistical Institute, India
Hwajeong Seo	Hansung University, South Korea
Jae Hong Seo	Myongji University, South Korea
Sang Uk Shin	Pukyong National University, South Korea
Youngjoo Shin	Kwangwoon University, South Korea
Rainer Steinwandt	Florida Atlantic University, USA
Hung-Min Sun	National Tsing Hua University, Taiwan
Atsushi Takayasu	University of Tokyo, Japan
Jorge Villar	Universitat Politècnica de Catalunya, Spain
Yongzhuang Wei	Guilin University of Electronic Technology, China
Wenling Wu	Institute of Software, Chinese Academy of Sciences, China

Toshihiro Yamauchi	Okayama University, Japan
Sang-Soo Yeo	Mokwon University, South Korea
Yongjin Yeom	Kookmin University, South Korea
Okyeon Yi	Kookmin University, South Korea
Dae Hyun Yum	Myongji University, South Korea
Aaram Yun	UNIST, South Korea

Contents

Symmetric Key Encryption

CHAM: A Family of Lightweight Block Ciphers for Resource-Constrained Devices	3
<i>Bonwook Koo, Dongyoung Roh, Hyeonjin Kim, Younghoon Jung, Dong-Geon Lee, and Daesung Kwon</i>	
Improved Meet-in-the-Middle Attacks on Reduced Round Kuznyechik	26
<i>Mohamed Tolba and Amr M. Youssef</i>	
Security of Stateful Order-Preserving Encryption.	39
<i>Kee Sung Kim, Minkyu Kim, Dongsoo Lee, Je Hong Park, and Woo-Hwan Kim</i>	

Homomorphic Encryption

Cryptanalysis of Tran-Pang-Deng Verifiable Homomorphic Encryption	59
<i>Shuaijianni Xu, Yan He, and Liang Feng Zhang</i>	
Multi-party (Leveled) Homomorphic Encryption on Identity-Based and Attribute-Based Settings.	71
<i>Veronika Kuchta, Gaurav Sharma, Rajeev Anand Sahu, and Olivier Markowitch</i>	
Improved Key Generation Algorithm for Gentry’s Fully Homomorphic Encryption Scheme	93
<i>Yang Zhang, Renzhang Liu, and Dongdai Lin</i>	
Subring Homomorphic Encryption	112
<i>Seiko Arita and Sari Handa</i>	

Side Channel Analysis and Implementation

Novel Leakage Against Realistic Masking and Shuffling Countermeasures: Case Study on PRINCE and SEED	139
<i>Yoo-Seung Won, Aesun Park, and Dong-Guk Han</i>	
Detecting Similar Code Segments Through Side Channel Leakage in Microcontrollers	155
<i>Peter Samarin and Kerstin Lemke-Rust</i>	

Secure Number Theoretic Transform and Speed Record for Ring-LWE Encryption on Embedded Processors 175
Hwajeong Seo, Zhe Liu, Taehwan Park, Hyeokchan Kwon, Sokjoon Lee, and Howon Kim

Broadcast Encryption

Recipient Revocable Broadcast Encryption Schemes Without Random Oracles 191
Kamalesh Acharya and Ratna Dutta

Recipient Revocable Broadcast Encryption with Dealership 214
Joon Sik Kim, Youngkyung Lee, Jieun Eom, and Dong Hoon Lee

Elliptic Curve

Solving 114-Bit ECDLP for a Barreto-Naehrig Curve 231
Takuya Kusaka, Sho Joichi, Ken Ikuta, Md. Al-Amin Khandaker, Yasuyuki Nogami, Satoshi Uehara, Nariyoshi Yamai, and Sylvain Duquesne

On the Computational Complexity of ECDLP for Elliptic Curves in Various Forms Using Index Calculus. 245
Chen-Mou Cheng, Kenta Koderu, and Atsuko Miyaji

Signature and Protocol

Two Mutual Authentication Protocols Based on Zero-Knowledge Proofs for RFID Systems 267
Hafsa Assidi, Edoukou Berenger Ayebie, and El Mamoun Souidi

On New Zero-Knowledge Arguments for Attribute-Based Group Signatures from Lattices 284
Veronika Kuchta, Rajeev Anand Sahu, Gaurav Sharma, and Olivier Markowitch

Security Analysis of Improved Cubic UOV Signature Schemes. 310
Kyung-Ah Shim, Namhun Koo, and Cheol-Min Park

Network and System Security

Evaluating the Impact of Juice Filming Charging Attack in Practical Environments 327
Weizhi Meng, Wang Hao Lee, Zhe Liu, Chunhua Su, and Yan Li

Reading Network Packets as a Natural Language for Intrusion Detection 339
Mamoru Mimura and Hidema Tanaka

Friend-Safe Adversarial Examples in an Evasion Attack
on a Deep Neural Network. 351
Hyun Kwon, Hyunsoo Yoon, and Daeseon Choi

Author Index 369

Symmetric Key Encryption



CHAM: A Family of Lightweight Block Ciphers for Resource-Constrained Devices

Bonwook Koo^(✉), Dongyoung Roh, Hyeonjin Kim, Younghoon Jung,
Dong-Geon Lee, and Daesung Kwon

National Security Research Institute, Daejeon, Republic of Korea
{[bwkoo](mailto:bwkoo@nsr.re.kr),[dyroh](mailto:dyroh@nsr.re.kr),[mikjh](mailto:mikjh@nsr.re.kr),[sky1236](mailto:sky1236@nsr.re.kr),[guneez](mailto:guneez@nsr.re.kr),[ds_kwon](mailto:ds_kwon@nsr.re.kr)}@nsr.re.kr

Abstract. In this paper, we propose a family of lightweight block ciphers CHAM that has remarkable efficiency on resource-constrained devices. The family consists of three ciphers, CHAM-64/128, CHAM-128/128, and CHAM-128/256 which are of the generalized 4-branch Feistel structure based on ARX (Addition, Rotation, XOR) operations.

In hardware implementations, CHAM requires smaller areas (73% on average) than SIMON [8] through the use of a *stateless-on-the-fly* key schedule which does not require updating a key state. Regarding software performance, it achieves outstanding figures on typical IoT platforms in terms of the balanced performance metrics introduced in earlier works. It shows a level of performance competitive to SPECK [8] mainly due to small memory size required for round keys. According to our cryptanalysis results, CHAM is secure against known attacks.

Keywords: Lightweight block cipher · Stateless-on-the-fly · ARX

1 Introduction

We are in the pervasive computing era. One can interact with many computing devices, such as laptop computers, tablets, smart phone, and even glasses with computing capabilities. Gartner, Inc. forecasts that in 2017, up 31 percent from 2016, 8.4 billion connected devices will be in use worldwide, with the number reaching 20.4 billion by 2020. The development and spread of these computing devices have made human life more convenient and enriching. However, in terms of adverse effects, threats that existed in cyberspace have spread and expanded to our everyday lives. So, it has become crucial to address security issues in relation to highly constrained devices so as to protect our lives and property. The first step in securing pervasive devices is to ensure that all data from them are transferred confidentially. Cryptographic algorithms, especially block ciphers, have been used extensively to perform this role.

Lightweight cryptography is essential for reducing size and cost of computing and communicating devices. So in cryptography, lightness has been moved from implementation issues to design consideration. A lot of block

ciphers have been designed for being lightweight in various aspect (for example chip size, code size, power consumption, memory usage, and so on) on various platforms for the past decade. Some of these algorithms have been shunned for various reasons including security problems, but still many algorithms are being discussed, applied or standardized in the real world such as HIGHT [31], PRESENT [6], CLEFIA [44], KATAN/KTANTAN [21], PRINTCIPHER [34], LED [29], PICCOLO [43], PRINCE [19], SIMON/SPECK [8], LEA [30], PRIDE [1], MIDORI [3], SPARX [28], SKINNY [11] and recently proposed GIFT [4].

Among these algorithms, SIMON and SPECK designed by NSA in 2013, are the most focused and evaluated ciphers. They are families of non-S-box based lightweight block ciphers. SIMON is tuned for optimal performance in hardware, while SPECK is designed for optimal performance in software. SPECK is in ARX structure. They are arguably regarded as the best algorithms among previously proposed lightweight block ciphers on hardware and software platforms.

LEA is another ARX cipher designed by Hong et al. in 2013 and suitable for common platforms (mainly 32-bit microcontrollers), but its performance is not so impressive on 8-bit and 16-bit microcontrollers.

Contributions. We have attempted to improve LEA by increasing the level of suitability for resource-constrained environment such as hardware and 8-bit and 16-bit microcontrollers. As a result, we propose a new family of block ciphers, CHAM.

CHAM has the following features:

- CHAM uses an extremely simple key schedule possibly being implemented without updating key states. This feature helps to reduce the number of flip-flops when implementing our ciphers on hardware.
- Numbers of round keys are far fewer than the numbers of rounds, and round functions reuse them iteratively. This reduces the memory size necessary to store the round keys.
- Encryption uses two types of left rotation, by 1 bit and by 8 bits, to minimize the number of operations on 8-bit AVR microcontroller.
- Round indexes are used as round constants instead of random values to save resources while defending against slide attacks [18] and basic rotational attacks [33].

There are many lightweight block ciphers which have the same feature of key schedule, such as HIGHT, KATAN, PRINTCIPHER, LED, PICCOLO, PRINCE, PICARO [41], PRIDE, and KHUDRA [36]. However, some of these ciphers are prone to or fully broken by related-key attacks [22, 23, 25, 32, 37, 54]. Bearing this in mind, we analyze the security of CHAM against various attacks, including differential cryptanalysis and linear cryptanalysis. In particular, we give more attention to security analyses in the related-key model.

By efficiency evaluation and comparison, we draw the following results. On hardware, we implement SIMON and our ciphers by minimizing the area with the IBM 130 nm CMOS generic process library. The results are briefly presented

in Table 1. Table 2 shows a brief comparison with SPECK on the two widely adopted target platforms of Atmega128 and MSP430. CHAM shows efficiency competitive with one of the top-ranked algorithms, SPECK on the platforms. In summary, CHAM can be implemented using smaller area than SIMON in hardware environment and shows a similar level of efficiency to SPECK in 8- and 16-bit software environments.

Table 1. Area-optimized implementation results in gate for CHAM and SIMON.

Algorithm	Library	64/128	128/128	128/256	Ref.
CHAM	IBM 130 nm	665	1,057	1,180	This paper
SIMON	IBM 130 nm	958	1,234	1,782	[8]

Table 2. Software efficiency comparison in *rank* metric [9] (Larger is better).

Scenario	Algorithm	Atmega128	MSP430	Ref.
Fixed key [9]	CHAM-64/128	27.9	49.3	This paper
	SPECK-64/128	29.8	50.0	[8]
	CHAM-128/128	17.1	25.0	This paper
	SPECK-128/128	12.7	21.7	[8]
Communication [26, 27] (without decryption)	CHAM-64/128	7.2	11.1	This paper
	SPECK-64/128	6.3	9.7	FELICS website

2 Specifications and Design Principles

CHAM is a family of block ciphers with a 4-branch generalized Feistel structure. Each cipher is denoted by CHAM- n/k with a block size of n -bit and a key size of k -bit. Table 3 shows the list of ciphers in the family and their parameters. Here, r and w denote the number of rounds and the bit length of a branch (word), respectively.

Table 3. List of CHAM ciphers and their parameters.

Cipher	n	k	r	w	k/w
CHAM-64/128	64	128	80	16	8
CHAM-128/128	128	128	80	32	4
CHAM-128/256	128	256	96	32	8

CHAM consists of three algorithms with different parameters. All three algorithms in the family are suitable for resource-constrained environments, but at

the same time they have slightly different applications according to their parameters. CHAM-64/128 is more suitable for low-end devices, CHAM-128/128 is a better choice for general use on 32-bit microcontrollers, and CHAM-128/256 can serve where a higher security level is required.

2.1 Notations

We use the following notations to describe CHAM- n/k .

- $x\|y$: concatenation of bit strings x and y
- $x \boxplus y$: addition of x and y modulo 2^w
- $x \oplus y$: bit-wise exclusive OR (XOR) of x and y
- $\text{ROL}_i(x)$: rotation of w -bit word x to the left by i bits
- $w_H(x)$: Hamming weight of x , the number of nonzero bits in x .

2.2 Specifications

CHAM- n/k encrypts a plaintext $P \in \{0, 1\}^n$ to a ciphertext $C \in \{0, 1\}^n$ using a secret key $K \in \{0, 1\}^k$ by applying r iterations of the round function as follows.

Divide the plaintext P into four w -bit words $X_0[0]$, $X_0[1]$, $X_0[2]$, and $X_0[3]$, where $P = X_0[0]\|X_0[1]\|X_0[2]\|X_0[3]$. Next, compute the n -bit output $X_{i+1} = X_{i+1}[0]\|X_{i+1}[1]\|X_{i+1}[2]\|X_{i+1}[3]$ of the i -th round for $0 \leq i < r$ by

$$\begin{aligned} X_{i+1}[3] &\leftarrow \text{ROL}_8((X_i[0] \oplus i) \boxplus (\text{ROL}_1(X_i[1]) \oplus RK[i \bmod 2k/w])), \\ X_{i+1}[j] &\leftarrow X_i[j + 1] \text{ for } 0 \leq j \leq 2, \end{aligned}$$

if i is even, otherwise,

$$\begin{aligned} X_{i+1}[3] &\leftarrow \text{ROL}_1((X_i[0] \oplus i) \boxplus (\text{ROL}_8(X_i[1]) \oplus RK[i \bmod 2k/w])), \\ X_{i+1}[j] &\leftarrow X_i[j + 1] \text{ for } 0 \leq j \leq 2, \end{aligned}$$

where $RK[i \bmod 2k/w]$ is the round key.

Then, the ciphertext C is defined by $C = X_r[0]\|X_r[1]\|X_r[2]\|X_r[3]$.

The key schedule of CHAM- n/k takes the secret key $K \in \{0, 1\}^k$ and generates the $2k/w$ w -bit round keys $RK[0], RK[1], \dots, RK[2k/w - 1]$. Initially, divide the secret key into k/w w -bit words $K[0], K[1], \dots, K[k/w - 1]$, where $K = K[0]\|K[1]\|\dots\|K[k/w - 1]$. Then the round keys are generated as follows.

$$\begin{aligned} RK[i] &\leftarrow K[i] \oplus \text{ROL}_1(K[i]) \oplus \text{ROL}_8(K[i]), \\ RK[(i + k/w) \oplus 1] &\leftarrow K[i] \oplus \text{ROL}_1(K[i]) \oplus \text{ROL}_{11}(K[i]), \end{aligned}$$

where $0 \leq i < k/w$.

The structures of the key schedule and round function of CHAM are depicted in Fig. 1, and the test vectors are provided in Appendix A.

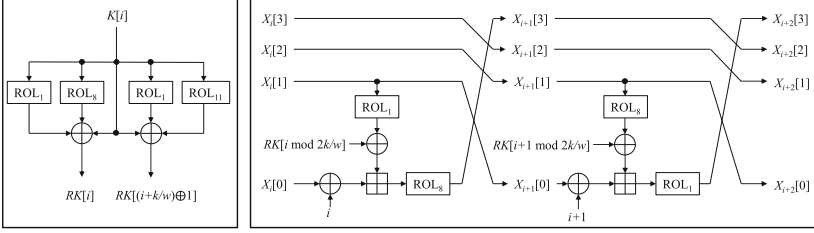


Fig. 1. Key schedule (left) and two consecutive round functions beginning with the even i -th round (right).

2.3 Design Principles

Design Paradigm and Design Approach. Our goal is to design a cipher that shows better efficiency than SIMON and SPECK on both (resource-constrained) hardware and software. So we design in the ARX design paradigm so that our cipher could have advantages in various resource-constrained environments. However, it is still not easy to prove nor to theoretically bound the minimum number of rounds with respect to the security of an ARX block cipher (although there is an exception [28]). Hence, we decided to design our algorithms (round function and key schedule) with a focus on efficiency and lightweight aspect and to perform a comprehensive and detailed cryptanalysis relying on experimental methods.

Round Functions. In the instruction sets of AVR and MSP introduced in Sect. 4, there is no single instruction for a rotation of arbitrary amounts. Only byte-swap and 1-bit rotation can be used for implementing $ROL_{r_i}()$. So, if we want to implement 3-bit rotation, we have to implement 1-bit rotation for 3 times. Hence, we choose every rotation amount as close as possible to 0 or 8 for the round functions (as well as the key schedule) to achieve a better performance. At the same time, in order to reduce the number of rounds while keeping security, we select several combinations of rotation amounts for a round function and consider combinations of the round functions for the whole encryption.

At first, we compare several cases of using a single round function and those of using two slightly different round functions. Let's denote rotation amounts of two consecutive rounds as a 4-tuple (a, b, c, d) such that a and c are rotation amounts for the values before the round key XOR and b and d are rotation amounts for outputs of the additions. For example, CHAM is of type $(1, 8, 8, 1)$. We estimate the maximum numbers of rounds of differential characteristics with probabilities greater than 2^{-n} when $w = 16$. Table 4 shows the results.

In Table 4, the type $(1, 8, 9, 1)$ looks like the best choice, however the type $(1, 8, 9, 1)$ requires more operations than the type $(1, 8, 8, 1)$ for every two rounds, so the overall efficiency of the type $(1, 8, 8, 1)$ is better. Also, we test the case of using 4-round alternating structures but could not find any better case than our selection in the overall efficiency point of view.

Table 4. Security evaluation results of several types of round functions.

Type	(0, 9, 0, 9)	(1, 8, 1, 8)	(1, 8, 1, 9)	(1, 8, 8, 1)	(1, 8, 9, 1)
Diff. ch. round	58	47	40	36	33

Additionally, we use round indexes as round constants not only to defend the slide attack and the rotational attack and but also to reduce the extra registers for storing constants on both hardware and software implementations.

Key Schedule. Key schedule is a procedure to calculate round keys from a given secret key. Many block ciphers (AES, SIMON/SPECK [8], and so on) have their own key schedules that consist of functions calculating each round keys from the current key state and update the key state. And key schedules of some other ciphers (PICCOLO [43], LED [29], and so on) consist of functions which calculate every round key directly from the secret key. Although the former key schedules could be also implemented to calculate every round key directly from the secret key, the latter key schedules are more suitable for such implementation. We call this implementation method as *stateless-on-the-fly* implementation and a key schedule like latter ones as *stateless-on-the-fly* key schedule.

It is clear that a stateless-on-the-fly implementation of a key schedule does not require areas (flip-flops) for storing key states on hardware, so it can be replaced by some selecting logics. For a k -bit secret key, at least $k \times s_f$ gates are required for storing key state while roughly at most $(k - 1) \times s_m$ gates are needed for some selecting logic, where s_f and s_m are gate counts of D flip-flop and 2-to-1 MUX, respectively. Therefore, the stateless-on-the-fly implementation helps to save the hardware areas when $s_m < s_f$. For an example of CHAM-64/128 in IBM130 [8], we can theoretically save 259 gates by adopting stateless-on-the-fly implementation for serial implementation, since $s_m = 2.25$ and $s_f = 4.25$. In practice, more gates could be saved for stateless-on-the-fly implementation because compiler could employ 3-to-1 or 4-to-1 MUXes instead of 2-to-1 MUXes to optimize the area while D flip-flop cannot be replaced by the other cell.

The key schedules of CHAM consist of a linear function Φ from $\{0, 1\}^w$ to $\{0, 1\}^{2w}$. Let I_w be the $w \times w$ identity matrix and I_w^l be a $w \times w$ matrix whose i -th row is defined by l times left rotation of i -th row of I_w for $i = 1, 2, \dots, w$. Then, the function Φ can be expressed by multiplication with the following binary matrix A defined by

$$A = (A_1 | A_2)^T = (I_w \oplus I_w^1 \oplus I_w^8 \mid I_w \oplus I_w^1 \oplus I_w^{11})^T.$$

Let \mathcal{A} be a set of all matrices $(I_w^{l_0} \oplus I_w^{l_1} \oplus I_w^{l_2} \mid I_w^{l_3} \oplus I_w^{l_4} \oplus I_w^{l_5})^T$, where $l_0 = l_3 = 0, l_0 \neq l_1 \neq l_2, l_3 \neq l_4 \neq l_5$ and $0 < l_1, l_2, l_4, l_5 < w$. Let

$$\mathcal{A}_j = \{M \in \mathcal{A} \mid \min_{x \in \{0, 1\}^w \setminus \{0\}} (w_H(M \cdot x)) = j\}.$$

We make certain that in the case of $w = 16$, $\mathcal{A} = \mathcal{A}_2 \cup \mathcal{A}_4 \cup \mathcal{A}_6$.

The matrix A is the case of $l_0 = 0, l_1 = 1, l_2 = 8, l_3 = 0, l_4 = 1,$ and $l_5 = 11$. It is chosen from \mathcal{A}_6 considering the number of instructions for implementations on AVR and MSP and cryptographic properties in the related-key model. Note that the matrix with $l_0 = 0, l_1 = 1, l_2 = 8, l_3 = 0, l_4 = 1,$ and $l_5 = 9$ is in \mathcal{A}_4 and shows worse property for the related-key differential cryptanalysis. And the matrix with $l_0 = 0, l_1 = 1, l_2 = 8, l_3 = 0, l_4 = 1,$ and $l_5 = 10$ also shows worse property than A in the viewpoint of the related-key differential cryptanalysis, even though it belongs \mathcal{A}_6 .

Additionally, we let the round keys be iteratively re-applied to reduce the size of memory for storing the round keys in software. And we give the round key index a tweak so that a w -bit word of a secret key cannot regularly act on every k/w rounds.

3 Hardware Implementation

We implemented CHAM in Verilog and synthesized using Synopsys Design Compiler 2008.09-SP5 without the *compile_ultra* feature. For the synthesis, the 100-kHz frequency constraint is imposed. Clocks for plaintext flip-flops are gated and scan flip-flops are not used. To evaluate and compare the hardware cost and performance, we use the UMC 90 nm, UMC 180 nm CMOS generic process library (UMC90, UMC180) provided by the Faraday Technology Corporation and IBM's 8RF 130 nm process (IBM130).

Figures 2 and 3 present the round-based and the bit-serial hardware architecture of CHAM, respectively. In both architectures, there are no flip-flops for

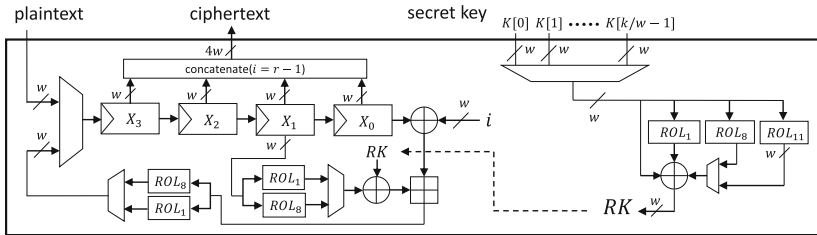


Fig. 2. Round-based hardware architecture of CHAM.

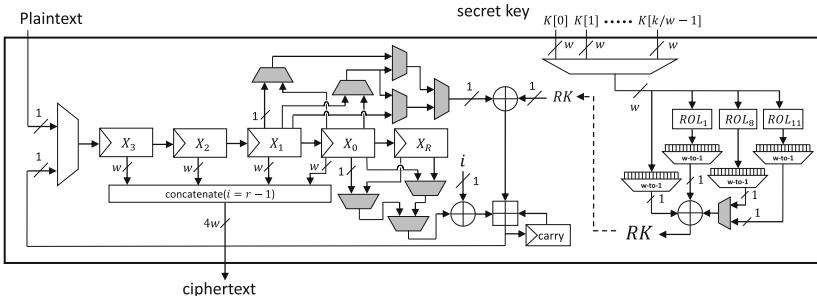


Fig. 3. Serialized hardware architecture of CHAM.

storing the secret key because the secret key is injected for each clock count. Instead of the flip-flops, a k/w -to-1 w -bit MUX is required for the round-based architecture, and a k -to-1 1-bit MUX is required for the bit-serial architecture. Since the size of the flip-flops is larger than that of the MUXes, we can save as much as the difference between the two sizes.

Table 5 shows the results of comparison to other lightweight block ciphers for each parameter and architecture in area (GE, gate equivalent) and throughput

Table 5. Comparison of the hardware implementation of CHAM and other ciphers.

n	k	Cipher	Bit-serial		Round-based		Tech.	Ref.
			Area (GE)	Throughput	Area (GE)	Throughput		
64	128	Twine	–	–	1,866	178.0	90n	[47]
		High	–	–	3,048	188.3	250n	[31]
		Gift	930	–	1,345	–	STM90	[4]
		Midori	–	–	1,542	–	STM90	[3]
		Midori	–	–	1,638	–	STM65	[3]
		Skinny	1,399	8.1	1,696	177.8	UMC180	[11]
		LED	1,265	3.4	–	–	–	[29]
		†LED	700	3.4	–	–	–	[29]
		Present	1,391	11.4	1,884	200.0	UMC180	[42]
		Piccolo	758	12.1	1,197	193.9	130n	[43]
		Simon	944	4.2	1,403	133.3	‡IBM130	[55]
		Simeck	924	4.2	1,365	133.3	‡IBM130	[55]
		Simon	958	4.2	1,417	133.3	IBM130	[8]
		Speck	996	3.4	1,658	206.5	IBM130	[8]
		CHAM	665	5.0	826	80.0	IBM130	Ours
CHAM	859	5.0	1,110	80.0	UMC180	Ours		
CHAM	727	5.0	985	80.0	UMC90	Ours		
128	128	Gift	1,213	–	1,997	–	STM90	[4]
		Midori	–	–	2,522	–	STM90	[3]
		Midori	–	–	2,714	–	STM65	[3]
		Skinny	1,840	14.7	2,391	320.0	UMC180	[11]
		LEA	2,302	4.2	3,826	76.2	UMC130	[30]
		AES	–	–	2,400	57.0	UMC180	[40]
		Simon	1,234	2.9	2,090	182.9	IBM130	[8]
		Speck	1,280	3.0	2,727	376.5	IBM130	[8]
		CHAM	1,057	5.0	1,499	160.0	IBM130	Ours
		CHAM	1,296	5.0	1,899	160.0	UMC180	Ours
		CHAM	1,084	5.0	1,691	160.0	UMC90	Ours
		128	256	Skinny	2,655	12.3	3,312	266.7
Simon	1,782			2.6	2,776	168.4	IBM130	[8]
Speck	1,840			2.8	3,284	336.8	IBM130	[8]
CHAM	1,180			4.2	1,622	133.3	IBM130	Ours
CHAM	1,481			4.2	2,087	133.3	UMC180	Ours
CHAM	1,256			4.2	1,864	133.3	UMC90	Ours

†: Hard-wired key implementation.

‡: Not exactly same to the non-daggered IBM130.

(Kbps@100 KHz). As the table shows, every instance in our family can be implemented in much smaller area than SIMON with the corresponding parameter. Precisely, the area of the sequential logic for CHAM is much smaller than that of SIMON because the secret key is not stored in flip-flops. Moreover, in most cases (except for the round-based CHAM-64/128), the throughputs per area's are also better than those of SIMON.

4 Software Implementation

We compare software performances of our cipher and existing algorithms on three typical microcontrollers suitable for lightweight devices. In [10, 26, 53], Atmega128 (AVR), MSP430F1611 (MSP), and SAM3X8E (ARM) are widely used as target 8, 16, and 32-bit microcontrollers, respectively. We also choose them as our target platforms. Their features are briefly presented in Appendix B.1.

Software performances are measured based on two metrics, *rank* and FOM, under two usage scenarios, *Fixed-key* and *Communication*. The metrics and scenarios are introduced in [9, 26, 27], and described in Appendices B.3 and B.4.

All our software implementations of CHAM are done in assembly language to draw better performance as much as possible. Some of the ideas are presented in Appendix B.2. Note also that, in our work, all the best results of CHAM are obtained from 4-round unrolled implementations except for CHAM-128/128 on the ARM. In that case, 8-round unroll results the best FOM value.

Table 6 presents the results of a performance comparison on the AVR and MSP platforms using the rank metric for the fixed-key scenario. Either SPECK or CHAM is always ranked at the top on both AVR and MSP, for all of the

Table 6. Performance comparison using the rank metric on AVR and MSP, under the fixed-key scenario.

Size (n/k)	Algorithm	AVR				MSP			
		ROM	RAM	cpb	Rank	ROM	RAM	cpb	Rank
64/128	SPECK [10]	218	0	154	29.8	204	0	98	50.0
	CHAM	202	3	172	27.9	156	8	118	49.3
	SIMON [10]	290	0	253	13.6	280	0	177	20.2
	SPARX	448	2	224	9.9	366	14	136	18.7
	HIGHT [9]	336	0	311	9.6	–	–	–	–
128/128	CHAM	362	16	148	17.1	280	20	125	25.0
	SPECK [10]	460	0	171	12.7	438	0	105	21.7
	AES [10]	970	18	146	6.8	–	–	–	–
	LEA	754	17	203	6.3	646	24	147	9.8
128/256	CHAM	396	16	177	13.2	312	20	148	19.2
	SPECK [9]	476	0	181	11.6	–	–	–	–

parameters of the block size and key size. They outperform the remaining ciphers. The results for SPARX [28] and LEA are drawn from performance reports posted on the FELICS [26] project’s website.

Table 7 shows the performances based on the rank metric under the one-way communication scenario, emphasizing variable key encryption without decryption. The results of CHAM on the AVR and MSP, obtained by optimization¹ level two, show slightly better performance than that of SPECK. The ranks of SPECK are derived from the data reported in the FELICS website, and the best ones are shown in the table.

On the ARM platform, CHAM-64/128 shows relatively poor result due to its 16-bit word size.² On the other hand, CHAM-128/128, based on a 32-bit word size, presents relatively decent result compared to SPECK-64/128.³

Table 7. Performance comparison using the rank metric in the one-way communication scenario. EKS and Enc represent the key schedule and encryption.

Platform	Algorithm	ROM		RAM		Cycles		cpb	Rank
		EKS	Enc	Stack	Data	EKS	Enc		
AVR	CHAM-64/128	72	280	11	184	309	23,664	187	7.2
	SPECK-64/128	178	240	12	260	1,401	19,888	166	6.3
MSP	CHAM-64/128	68	210	16	184	275	16,715	133	11.1
	SPECK-64/128	126	180	16	260	1,242	14,155	120	9.7
ARM	SPECK-64/128	52	164	36	260	516	6,323	53	23.2
	CHAM-128/128	44	210	48	192	95	8,846	70	19.5
	CHAM-64/128	124	272	48	184	170	16,944	134	8.7

Table 8 shows performance comparison based on the FOM metric under the communication scenario. SPECK-64/128 reveals the best value, followed by CHAM-128/128.⁴ Considering its block size, CHAM-128/128 can be regarded as a very good performer. The FOM of CHAM-64/128 is degraded by relatively poor performance on ARM. However, due to the excellence on AVR and MSP, its FOM is still ahead of all other competitors, except for SPECK. Note that all the values other than CHAM are presented on the FELICS website.

¹ The FELICS platform provides a unified implementation environment which generates performance figures automatically. The FELICS software framework, written in C language, permits users to implement only the core parts of encryption, decryption and their key schedules. Due to the common operational C source codes, the performance results are affected by the compiler’s optimization option.

² A SIMD implementation might enhance performance. Since the ARMv7-M architecture provides very limited instructions for SIMD arithmetics, it seems to be very difficult to get non-trivial performance gain from SIMD approach.

³ The performance of SPECK-128/128 is not yet reported in the FELICS website.

⁴ In the comparison, we exclude Chaskey algorithm because it is not considered as a block cipher.

Table 8. Performance comparison using the FOM metric in the communication scenario. The key schedule, encryption and decryption steps are all included.

Algorithm	AVR			MSP			ARM			FOM
	ROM	RAM	cpb	ROM	RAM	cpb	ROM	RAM	cpb	
SPECK-64/128	874	302	351	572	296	253	444	308	129	5.00
CHAM-128/128	1,230	262	337	926	258	298	528	272	144	5.47
CHAM-64/128	844	225	394	578	220	290	606	244	319	6.52
RECTANGLE-64/128	1,118	353	506	844	402	361	654	432	289	7.80
SPARX-64/128	1,198	392	512	966	392	287	1,200	424	319	8.57

5 Security Analysis

In this section, we summarize the security analysis results of CHAM and describe how we determine the number of rounds for each algorithm. We then present detailed cryptanalysis results of a number of well-known attacks applicable to our ciphers.

5.1 Summary of the Security Analysis

Table 9 shows the maximum numbers of rounds of characteristics that can be used for each attack. Based on these results, we determine the total number of rounds of each cipher with the following considerations

- Round extension of characteristics
- Round extension during key-recovery phase
- Security margin

Table 9. The numbers of rounds of the best discovered characteristics for each cipher and cryptanalysis.

CHAM-	DC	RDC	LC	BC	RBC	IDC	ZCLC	DLC	RDLC	IC	RXDC
64/128	36	34	34	35	41	18	21	34	36	16	16
128/128	45	33	40	47	36	15	18	45	39	16	23
128/256	45	40	40	47	47	15	18	45	45	16	23

(R)DC: (Related-key)Differential Cryptanalysis, LC: Linear Cryptanalysis, (R)BC: (Related-key)Boomerang Cryptanalysis, IDC: Impossible Differential Cryptanalysis ZCLC: Zero-Correlation Linear Cryptanalysis, (R)DLC: (Related-key)Differential-Linear Cryptanalysis, IC: Integral Cryptanalysis, RXDC: Rotational-XOR-Differential Cryptanalysis.

By applying the probability gathering technique used in [45] to our differential cryptanalysis, an attacker may find differentials longer than the differential

characteristics in Table 10 in Appendix C.1. But we check experimentally and confirm that the attacker can have at most four rounds longer differentials. Also, we verified that a boomerang and a rectangle characteristics can also be extended at most four rounds despite their use of two differential characteristics. Although such extensions can be applied only to the cryptanalyses based on difference or linear approximation, we assume every type of characteristic can be extended by four rounds regarding this technique.

In the key-recovery phase of an attack, if $k = 2n$, then one can attack k/w more rounds by guessing $k/2$ bits of the secret key before filtering. Otherwise, one cannot utilize this type of round extension. After filtering, one can add at most three rounds during key guessing and determining steps. Hence, we claim that the maximal numbers of rounds for which one can mount a key-recovery attack are at most $4 + 2(k/w - 4) + 3$ more than the numbers of rounds in Table 9, regardless of actual characteristics.

To estimate the numbers of rounds conservatively, we assume that an attacker can mount successful attacks using the characteristics we have found, even if the actual attacks are infeasible. Finally, we determine the numbers of rounds (instances of r), as shown in Table 3 with these considerations together with a security margin greater than 30% for full rounds⁵.

5.2 Cryptanalysis Results

We cryptanalyze our ciphers with the following well-studied techniques, and the results are summarized in Table 9. We use state-of-the-art techniques, modified for ARX structure ciphers [7, 17, 45, 49]. Using these results, we estimate the necessary numbers of rounds to ensure proper security. This subsection includes the results of (RK⁶) Differential, Linear, and (RK) Boomerang cryptanalysis and other detailed cryptanalysis results are provided in Appendix C due to the page limit.

(RK) Differential and Linear Cryptanalysis. We searched for (RK) differential characteristics [16] and linear approximations of each cipher using an automated algorithm known as the *threshold search* suggested in [17]. It is considered as an appropriate variant of Matsui’s branch-and-bound algorithm [39] for ARX ciphers. Using this threshold search approach, we found differential characteristics with a probability of $p > 2^{-n}$ for each cipher. These results are given in Table 10. We denote a (RK) differential characteristic by (Δ_{key}) , $\Delta_{\text{in}} \rightarrow \Delta_{\text{out}}$, where Δ_{key} is the difference of the secret key, and Δ_{in} and Δ_{out} are the input and output differences, respectively.

For a linear cryptanalysis [38], we found linear approximations with bias $\epsilon > 2^{-n/2}$ for each cipher. These results are given in Table 11. Γ_{in} and Γ_{out} denote the input and output masks, respectively. We calculate the correlation

⁵ 30% is a relatively high ratio for a security margin compared to those associated with other ciphers.

⁶ RK stands for “related-key”.

of the linear approximation by determining the bias of every addition using Wallén’s formula [52] and applying Piling-Up Lemma [38].

(RK) Boomerang Cryptanalysis. The (RK) boomerang characteristics [15, 50] are constructed with two short (RK) differential characteristics. These characteristics are the most threatening ones for our ciphers. Examples of short (RK) differential characteristics for constructing (RK) boomerang characteristics are shown in Table 12. A (RK) boomerang characteristic is valid when $pq > 2^{-n/2}$, where p and q are the probabilities of the two differential characteristics ϕ_1 and ϕ_2 , respectively, used for constructing the (RK) boomerang characteristic. Note that (RK) rectangle cryptanalysis [13, 15] works using the same characteristics.

6 Conclusion

In this paper, we have presented CHAM, a family of lightweight block ciphers that supports widely used block/key lengths. It is suitable for highly resource-constrained devices, especially area-constrained hardware with the help of the *stateless-on-the-fly* key schedule. Efficiency evaluations and comparisons with other lightweight block ciphers on various platforms are provided to demonstrate the merit of our algorithms. A variety of security analyses with extensive searching have convinced us that it is secure against known attacks. We also believe that it will be resistant to future attacks due to its high security margins. Moreover, the simplicity and flexibility of CHAM allow one to design ciphers with other block sizes and key lengths. Of course, a rigorous security analysis should be followed.

A Test Vectors

Test vectors are represented in hexadecimal with the prefix ‘0x’.

CHAM-64/128

```
secret Key : 0x0100 0x0302 0x0504 0x0706 0x0908 0x0b0a 0x0d0c 0x0f0e
plaintext  : 0x1100 0x3322 0x5544 0x7766
ciphertext : 0x453c 0x63bc 0xdcfa 0xbf4e
```

CHAM-128/128

```
secret Key : 0x03020100 0x07060504 0x0b0a0908 0x0f0e0d0c
plaintext  : 0x33221100 0x77665544 0xbbaa9988 0xffeeddcc
ciphertext : 0xc3746034 0xb55700c5 0x8d64ec32 0x489332f7
```

CHAM-128/256

```
secret Key : 0x03020100 0x07060504 0x0b0a0908 0x0f0e0d0c
             0xf3f2f1f0 0xf7f6f5f4 0xfbfaf9f8 0xfffffdff
plaintext  : 0x33221100 0x77665544 0xbbaa9988 0xffeeddcc
ciphertext : 0xa899c8a0 0xc929d55c 0xab670d38 0x0c4f7ac8
```

B Some Details About Software Implementation

B.1 Target Platforms

Atmega128 belongs to Atmel’s AVR microcontroller family with an 8-bit RISC architecture. It has 32 general-purpose registers and 133 instructions. It is equipped with 128 KBytes of flash and 4 KBytes of RAM. MSP430F1611, a microcontroller from Texas Instruments, adopts a 16-bit RISC architecture with 16 registers (12 of them are general-purpose registers) and 51 instructions, including the emulated ones. It has 48 KBytes of flash and 10 KBytes of RAM. The ARM Cortex-M3 is a 32-bit processor core based on the ARMv7-M architecture, with 12 general-purpose registers. The core is adopted in the Atmel SAM3X8E microcontroller installed on the well-known Arduino Due development board. SAM3X8E is equipped with 512 KBytes of flash and 96 KBytes of SRAM.

B.2 Implementating Bit-Wise Rotation

In ARX design like CHAM, rotations by certain bit sizes might be costly to implement on our 8 and 16-bit platforms. Efficient implementation of rotation is crucial to both high throughput and smaller memory. With this consideration, CHAM adopts ROL_8 , which can be performed for free on AVR platform.

The MSP430 provides a byte-swapping instruction, which is equivalent to ROL_8 for a 16-bit word. It is slightly tricky for a 32-bit word on the MSP430. Similarly to [20], ROL_8 can be carried out in seven instructions, as in Code 1 below.⁷ The code require an additional temporary register to hold a 16-bit data.

ARMv7-M provides a powerful instruction, barrel shifter, which can rotate a 32-bit word by any bit-size. Moreover, the instruction can perform a certain kind of operation additionally after the rotation. This fact gives rise to a good performance of CHAM-128/128. However, it appears that no single instruction of ARMv7-M can perform bit-wise rotation for a *16-bit* word. This explains the relatively low performance of CHAM-64/128 on ARMv7-M, as can be seen in Tables 7 and 8.

Code 1. MSP430 code for 8-bit left rotation; register pairs Rh and Rl store a 32-bit integer, ABCD, where each letter represents an 8-bit integer part; register Rt is temporary.

mov	Rl, Rt	:	Rh=(B,A),	Rl=(D,C),	Rt=(D,C)		ABCD
xor.b	Rh, Rl	:		Rl=(B^D,0)			
xor	Rh, Rl	:		Rl=(D,A)			
xor.b	Rt, Rh	:	Rh=(B^D,0)				
xor	Rt, Rh	:	Rh=(B,C)				
swpb	Rl	:		Rl=(A,D)			OODA
swpb	Rh	:	Rh=(C,B)				BCOO

⁷ We implement ROR₈, a right rotation for decryption, in eight instructions.

B.3 Performance Metrics

Lightweight IoT devices are usually considered to have constrained resources. This is why throughput alone does not fully describe the performance of an algorithm. A smaller code size and less RAM usage are also important factors to consider. In [9], the authors argue the same context and introduce the metric of *rank* as an overall performance indicator. It is defined as

$$\text{rank} = (10^6/\text{cpb})/(\text{ROM} + 2 \times \text{RAM}),$$

where *cpb* refers to the cycles per byte consumed for a task, and *ROM* and *RAM* are the byte sizes of the memory of each type. By definition, the larger rank is better. Note also that RAM is considered to be twice as costly as ROM.

The FOM (figure of merit) metric, defined recently in the FELICS [27], averages performances on AVR, MSP and ARM. For each implementation *i* on a device *d*, we measure memory usages $v_{\text{ROM}}^{i,d}$, $v_{\text{RAM}}^{i,d}$, and time cost $v_{\text{cost}}^{i,d}$. Among all the implementations of all the ciphers, the minimums of ROM, RAM and cost are also determined (possibly each from different implementations). Denote each minimum by m_{ROM}^d , m_{RAM}^d , and m_{cost}^d . Then, the performance parameter p_d for a cipher on a device *d* is defined by

$$p_d = \min_i \left(v_{\text{ROM}}^{i,d}/m_{\text{ROM}}^d + v_{\text{RAM}}^{i,d}/m_{\text{RAM}}^d + v_{\text{cost}}^{i,d}/m_{\text{cost}}^d \right).$$

Finally, the figure of merit for a cipher is defined by the average of three p_d 's,

$$\text{FOM} = (p_{\text{AVR}} + p_{\text{MSP}} + p_{\text{ARM}}) \times \frac{1}{3}.$$

The definition indicates the smaller FOM is better.⁸

B.4 Usage Scenarios

A block cipher suite usually consists of three distinctive algorithms: the key schedule, encryption and decryption. However, in lightweight applications, decryption tends to lose its role due to well-designed modes of operations for block ciphers. The combined performance of the key schedule and encryption is somewhat sensitive to their usage scenarios. For an easy comparison of our cipher with the results in the literature, we adopt two scenarios: simple encryption with a fixed key and data communication with variable keys.

Fixed-key scenario: In this scenario, a cipher is used for authenticating devices. There are no key schedules or decryption steps. Round keys are fixed in the device, i.e., specifically placed in the code area. Hence, their size is added to the code size. This scenario is used in Table 6.

⁸ It can be pointed out that the definition of the FOM has a drawback that whenever a new minimum is found by a better implementation of any cipher, the whole FOMs of all ciphers should be updated.

Communication scenario: In this scenario, a cipher is assumed to be used for data communication. It is defined as Scenario 1 in the FELICS [26]. Originally, the scenario contains encryption, decryption together with their key schedules, where 128 bytes of data are encrypted and decrypted in the CBC mode. Since encryption part is more important for lightweight application, we define *one-way* communication scenario by omitting the decryption part, which is used in Table 7. The scenario in its original meaning is also used in Table 8.

C Cryptanalysis Results

C.1 Tables of characteristics for (RK) Differential, Linear, and (RK) Boomerang Cryptanalysis

Tables 10, 11, and 12 show characteristics for (RK) Differential, Linear, and (RK) Boomerang Cryptanalysis, respectively.

Table 10. The best (RK) differential characteristics found. Only the input and output differences are shown. We assume that every operation is independent. The subscript x indicates a hexadecimal expression.

Model	n/k	Round/ p	Characteristic
Single-key model	64/128	$36/2^{-63}$	$(0004_x, 0408_x, 0A00_x, 0000_x) \rightarrow (0005_x, 8502_x, 0004_x, 0A00_x)$
	128/*	$45/2^{-125}$	$(01028008_x, 08200080_x, 04000040_x, 42040020_x)$ $\rightarrow (00000000_x, 00110004_x, 04089102_x, 00080010_x)$
Related-key model	64/128	Key diff.	$(0000_x, 0000_x, 0000_x, 0000_x, 0040_x, 0000_x, 0000_x, 4000_x),$
		$34/2^{-61}$	$(20A0_x, 1050_x, 4000_x, 2020_x) \rightarrow (0141_x, 8080_x, 4841_x, 830A_x)$
	128/128	Key diff.	$(00000000_x, 00000000_x, 00000000_x, 40000000_x),$
		$33/2^{-125}$	$(00410500_x, 00210080_x, 80102000_x, 40002041_x)$ $\rightarrow (12810001_x, 8A500940_x, 08001503_x, 06000220_x)$
128/256	Key diff.	$(00000000_x, 00000000_x, 00000000_x, 00000000_x,$ $00000000_x, 40000000_x, 00000000_x, 00000000_x)$	
	$40/2^{-127}$	$(42800040_x, 20010420_x, 00800104_x, 08408082_x)$ $\rightarrow (04405080_x, 80040892_x, 01000010_x, 80a00008_x)$	

Table 11. The best linear approximations found. We assume that every operation is independent.

n/k	Round	$\langle \Gamma_{in}, \Gamma_{out} \rangle$	ϵ
64/128	34	$\langle (0000_x, 1000_x, 10D9_x, 8C20_x), (CF06_x, 0202_x, 0100_x, 0009_x) \rangle$	2^{-31}
128/*	40	$\langle (00000000_x, 08001000_x, 40891038_x, 3000C800_x),$ $(06082000_x, 00001001_x, 42040030_x, 20020000_x) \rangle$	2^{-63}

Table 12. Examples of the best (RK) boomerang characteristics that we found.

Model	n/k	Round/ p or q	Characteristic
Single-key model	64/128	$17/2^{-15}$	$(8200_x, 0100_x, 0001_x, 8000_x) \rightarrow (0400_x, 0004_x, 0502_x, 0088_x)$
		$18/2^{-16}$	$(8200_x, 0100_x, 0001_x, 8000_x) \rightarrow (0004_x, 0502_x, 0088_x, 0000_x)$
	128/128	$23/2^{-30}$	$(08104000_x, 04002000_x, 00000020_x, 00000010_x)$ $\rightarrow (40010000_x, 20408100_x, 00000001_x, 02000080_x)$
		$24/2^{-33}$	$(02000000_x, 41000000_x, 20410000_x, 10008000_x)$ $\rightarrow (00000000_x, 00040001_x, 81020400_x, 00000004_x)$
Related-key model	64/128	Key diff. $20/2^{-15}$	$(0000_x, 0000_x, 0080_x, 0000_x, 0000_x, 8000_x, 0000_x, 0000_x)$ $(0084_x, 0000_x, 0000_x, 0000_x) \rightarrow (0400_x, 0105_x, 8080_x, 0100_x)$
		Key diff. $21/2^{-16}$	$(0000_x, 0000_x, 0000_x, 0000_x, 0080_x, 0000_x, 0000_x, 8000_x)$ $(0801_x, 8400_x, 0084_x, 0000_x) \rightarrow (0000_x, 0400_x, 0105_x, 8080_x)$
	128/128	Key diff. $18/2^{-31}$	$(00000000_x, 00000000_x, 00000000_x, 00800000_x)$ $(00000801_x, 80000400_x, 00800004_x, 00000000_x)$ $\rightarrow (00020484_x, 01000005_x, 08020000_x, 04010A00_x)$
		Key diff. $18/2^{-31}$	$(00000000_x, 00000000_x, 00000000_x, 00800000_x)$ $(00000801_x, 80000400_x, 00800004_x, 00000000_x)$ $\rightarrow (00020484_x, 01000005_x, 08020000_x, 04010A00_x)$
	128/256	Key diff. $23/2^{-27}$	$(00000000_x, 00000000_x, 00000000_x, 00000000_x)$ $(00000000_x, 40000000_x, 00000000_x, 00000000_x)$ $(08410002_x, 04008000_x, 00040080_x, 00020040_x)$ $\rightarrow (02020000_x, 01010240_x, 00000202_x, 04000004_x)$
		Key diff. $24/2^{-32}$	$(00000000_x, 00000000_x, 00000000_x, 00000000_x)$ $(00000000_x, 40000000_x, 00000000_x, 00000000_x)$ $(08410002_x, 04008000_x, 00040080_x, 00020040_x)$ $\rightarrow (01010240_x, 00000202_x, 04000004_x, 02008002_x)$

C.2 Impossible Differential Cryptanalysis and Zero-Correlation Linear Cryptanalysis

Impossible differential cryptanalysis [12] uses a differential characteristic that can never occur. A zero-correlation linear approximation [7] is the counter-part of the impossible differential characteristic in the linear cryptanalysis field. Examples of the best impossible differential characteristics and zero-correlation linear approximations as found here are given in Table 13.

C.3 (RK) Differential-Linear Cryptanalysis

A (RK) differential-linear approximation [14] is constructed with a short (RK) differential characteristic and a short linear approximation. A (RK) differential-linear approximation which has a correlation of $pc^2 > 2^{-n/2}$ can be used for a (RK) differential-linear attack, where p is the probability of the differential characteristic ϕ and c is the correlation of the linear approximation ψ . Examples showing how to build these (RK) differential-linear approximations are given in Table 14.

Table 13. Examples of the best impossible differential characteristics and zero correlation linear approximations found here.

Type	n/k	Round	Characteristic
Impossible differential	64/128	10	$(1^1 0^{15}, 0^{16}, 0^{16}, 0^{16}) \leftrightarrow (x^6 1^1 0^1 x^8, x^{14} 1^1 x^1, x^{16}, x^{13} 1^1 0^1 x^1)$
		8	$\not\leftrightarrow (x^{16}, x^{16}, x^{16}, x^8 1^1 0^7) \leftrightarrow (0^8 1^1 0^7, 0^{16}, 0^{16}, 0^{15} 1^1)$
	128/*	8	$(0^{32}, 0^{32}, 0^{32}, 1^1 0^{31}) \leftrightarrow (0^{32}, x^{32}, x^{32}, x^{30} 1^1 x^1)$
		7	$\not\leftrightarrow (x^{32}, x^{32}, x^{32}, x^{24} 1^1 0^7) \leftrightarrow (0^{32}, 0^{24} 1^1 0^7, 0^{32}, 0^{32})$
Zero correlation linear approximation	64/128	11	$(0^{15} 1^1, 1^1 0^{15}, 0^{16}, 0^{15} 1^1) \leftrightarrow (x^{16}, x^8 0^6 1^1 x^1, x^{16}, x^{16})$
		10	$\not\leftrightarrow (1^1 x^{15}, x^8 1^1 x^7, x^{16}, x^{16}) \leftrightarrow (0^{13} 1^1 0^2, 0^{16}, 0^{16}, 0^{16})$
	128/*	9	$(0^{32}, 0^{32}, 0^{31} 1^1, 1^1 0^{31}) \leftrightarrow (x^{32}, 0^{15} 1^1 x^8 0^8, x^{32}, x^{32})$
		9	$\not\leftrightarrow (0^{32}, 1^1 x^{31}, x^{32}, x^{32}) \leftrightarrow (0^{16} 1^1 0^{15}, 0^{32}, 0^{32}, 0^{32})$

Characteristics are denoted by 4-tuples of w -bit sequences separated by comma. 0^j , 1^j , and x^j are j -bit sequences of 0, 1, and free bit, respectively.

Table 14. Examples showing how to build the best (RK) differential-linear approximations found here.

Model	n/k	(RK) diff.-linear approx.		ϕ		ψ	
		Round	pc^2	Rounds	p	Round	c^2
Single-key model	64/128	34	2^{-31}	20	2^{-23}	14	2^{-8}
	128/*	45	2^{-63}	24	2^{-33}	21	2^{-30}
Related-key model	64/128	36	2^{-28}	22	2^{-18}	14	2^{-10}
	128/128	39	2^{-60}	17	2^{-26}	22	2^{-34}
	128/256	45	2^{-61}	23	2^{-27}	22	2^{-34}

C.4 Integral Cryptanalysis

Integral cryptanalysis [35] uses sets of chosen plaintexts of which a part is held constant and the other part varies through all possibilities. Considering ADD-balance [30], we found the following 16-round integral characteristic for all of our ciphers.

$$\begin{aligned}
& (A, C, C, C) \rightarrow (C, C, C, A) \rightarrow (C, C, A, C) \rightarrow (C, A, C, C) \rightarrow (A, C, C, A) \rightarrow (C, C, A, A) \rightarrow (C, A, A, C) \\
& \rightarrow (A, A, C, A) \rightarrow (A, C, A, \mathcal{B}_{\lll 1}^+) \rightarrow (C, A, \mathcal{B}_{\lll 1}^+, A) \rightarrow (A, \mathcal{B}_{\lll 1}^+, A, A) \rightarrow (\mathcal{B}_{\lll 1}^+, A, A, U) \\
& \rightarrow (A, A, U, U) \rightarrow (A, U, U, \mathcal{B}_{\lll 8}^+) \rightarrow (U, U, \mathcal{B}_{\lll 8}^+, U) \rightarrow (U, \mathcal{B}_{\lll 8}^+, U, U) \rightarrow (\mathcal{B}_{\lll 8}^+, U, U, U)
\end{aligned}$$

C , A , $\mathcal{B}_{\lll l}^+$, U represent a constant word, an active word, an ADD-balanced word when rotated to the right by l bits, and an unknown word, respectively. The above 16-round distinguisher means that if the first word of a plaintext is active, which takes all w -bit values at one time, and the other words of the plaintext are constants, then the first word of the output after 16 rounds is ADD-balanced when rotated to the right by 8 bits.

The bit-based division property [49] is an improvement of the division property [48] for non S-box-based ciphers. In [46], Sun et al. improved the integral cryptanalysis result of LEA slightly by applying the bit-based division property. Based on this result and owing to the similarity between LEA and our ciphers, we expect that the bit-based division property will not seriously improve our integral cryptanalysis.

C.5 Biclique Cryptanalysis

Wang et al. [51] showed that for variants of the Feistel structure, interleaving related-key differential trails cannot construct bicliques [5]. Hence, we consider the bicliques from independent related-key differentials, as our ciphers have a variant of the type-3 generalized Feistel structure. We calculate the total complexity C_{full} for a key recovery attack with independent bicliques using the following equation,

$$C_{full} = 2^{k-2d}(C_{biclique} + C_{precomp} + C_{recomp}),$$

where $C_{biclique}$, $C_{precomp}$, and C_{recomp} denote the complexities for building-biclique, pre-computation, and re-computation, respectively. Note that a trivial biclique for each cipher can be derived easily from related-key differentials. The specific complexities are shown in Table 15. The re-check complexity of a false positive is omitted in the above equation because it is negligible.

Table 15. Complexity of the biclique cryptanalysis for each parameter.

n/k	Biclique round	$C_{biclique}$	$C_{precomp}$	C_{recomp}	C_{full}
64/128	15	$2^{14.6}$	$2^{16.6}$	$2^{31.5}$	$2^{127.5}$
128/128	7	$2^{29.5}$	$2^{32.7}$	$2^{63.7}$	$2^{127.7}$
128/256	15	$2^{30.3}$	$2^{32.6}$	$2^{63.6}$	$2^{255.6}$

C.6 Rotational Cryptanalysis

The initial version of a rotational cryptanalysis [33] can be easily defended by constant-XOR's. However, the recently proposed rotational-XOR cryptanalysis [2] can be well-applied to ARX ciphers with constant XOR's. So, we carefully applied the rotational-XOR cryptanalysis to our algorithm and the results are shown in the Table 16. Characteristics are initial and final δ 's. Refer to [2] for attack conditions and the definition of δ .

C.7 Other Attacks

Applying round constants keeps our ciphers secure against slide attacks [18]. An algebraic attack [24] is not effective for our ciphers due to the high nonlinearity of such a case.

Table 16. The best rotational-XOR differential characteristics found.

n/k	Round/ p	Characteristic (δ)
64/128	$16/2^{-62.225}$	$(0000_x, 0000_x, 0000_x, 8002_x) \rightarrow (0004_x, 020e_x, 0805_x, 0810_x)$
128/*	$23/2^{-125.545}$	$(10000004_x, 08000000_x, 02080003_x, 40040000_x)$ $\rightarrow (00100004_x, 08080400_x, 00220008_x, a1110d00_x)$

References

- Albrecht, M.R., Driessen, B., Kavun, E.B., Leander, G., Paar, C., Yalçın, T.: Block ciphers – focus on the linear layer (feat. PRIDE). In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 57–76. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_4
- Ashur, T., Liu, Y.: Rotational cryptanalysis in the presence of constants. IACR Trans. Symmetric Cryptol. **2016**(1), 57–70 (2016)
- Banik, S., Bogdanov, A., Isobe, T., Shibutani, K., Hiwatari, H., Akishita, T., Regazzoni, F.: Midori: a block cipher for low energy. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9453, pp. 411–436. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48800-3_17
- Banik, S., Pandey, S.K., Peyrin, T., Sasaki, Y., Sim, S.M., Todo, Y.: GIFT: a small present. In: Fischer, W., Homma, N. (eds.) CHES 2017. LNCS, vol. 10529, pp. 321–345. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66787-4_16
- Bogdanov, A., Khovratovich, D., Rechberger, C.: Biclique cryptanalysis of the full AES. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 344–371. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_19
- Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C.: PRESENT: an ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74735-2_31
- Bogdanov, A., Rijmen, V.: Linear hulls with correlation zero and linear cryptanalysis of block ciphers. Des. Codes Cryptogr. **70**(3), 369–383 (2014). <https://doi.org/10.1007/s10623-012-9697-z>
- Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: The simon and speck families of lightweight block ciphers. IACR Cryptology ePrint Archive 2013, p. 404 (2013)
- Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: The SIMON and SPECK block ciphers on AVR 8-bit microcontrollers. In: Eisenbarth, T., Öztürk, E. (eds.) LightSec 2014. LNCS, vol. 8898, pp. 3–20. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16363-5_1
- Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: Simon and speck: block ciphers for the internet of things. IACR Cryptology ePrint Archive 2015, p. 585 (2015)
- Beierle, C., et al.: The SKINNY family of block ciphers and its low-latency variant MANTIS. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9815, pp. 123–153. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53008-5_5

12. 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). https://doi.org/10.1007/3-540-48910-X_2
13. 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). https://doi.org/10.1007/3-540-44987-6_21
14. Biham, E., Dunkelman, O., Keller, N.: Enhancing differential-linear cryptanalysis. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 254–266. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-36178-2_16
15. 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). https://doi.org/10.1007/11426639_30
16. Biham, E., Shamir, A.: Differential Cryptanalysis of the Data Encryption Standard. Springer, New York (1993). <https://doi.org/10.1007/978-1-4613-9314-6>. ISBN: 978-1-4613-9316-0, 978-1-4613-9314-6
17. Biryukov, A., Velichkov, V.: Automatic search for differential trails in ARX ciphers. In: Benaloh, J. (ed.) CT-RSA 2014. LNCS, vol. 8366, pp. 227–250. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-04852-9_12
18. Biryukov, A., Wagner, D.: Slide attacks. In: Knudsen, L. (ed.) FSE 1999. LNCS, vol. 1636, pp. 245–259. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48519-8_18
19. Borghoff, J., et al.: PRINCE – a low-latency block cipher for pervasive computing applications. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 208–225. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34961-4_14
20. Buhrow, B., Riemer, P., Shea, M., Gilbert, B., Daniel, E.: Block cipher speed and energy efficiency records on the MSP430: system design trade-offs for 16-bit embedded applications. In: Aranha, D.F., Menezes, A. (eds.) LATINCRYPT 2014. LNCS, vol. 8895, pp. 104–123. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16295-9_6
21. 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). https://doi.org/10.1007/978-3-642-04138-9_20
22. Canteaut, A., Lallemand, V., Naya-Plasencia, M.: Related-key attack on full-round PICARO. In: Dunkelman, O., Keliher, L. (eds.) SAC 2015. LNCS, vol. 9566, pp. 86–101. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-31301-6_5
23. Chen, J., Teh, J.S., Su, C., Samsudin, A., Fang, J.: Improved (related-key) attacks on round-reduced KATAN-32/48/64 based on the extended boomerang framework. In: Liu, J.K., Steinfeld, R. (eds.) ACISP 2016. LNCS, vol. 9723, pp. 333–346. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-40367-0_21
24. Courtois, N.T., Pieprzyk, J.: Cryptanalysis of block ciphers with overdefined systems of equations. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 267–287. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-36178-2_17
25. Dai, Y., Chen, S.: Cryptanalysis of full PRIDE block cipher. *Sci. China Inf. Sci.* **60**, 052108 (2017). <https://doi.org/10.1007/s11432-015-5487-3>
26. Dinu, D., Biryukov, A., Großschädl, J., Khovratovich, D., Le Corre, Y., Perrin, L.: FELICS - fair evaluation of lightweight cryptographic systems. In: NIST Workshop on Lightweight Cryptography 2015 National Institute of Standards and Technology (2015)

27. Dinu, D., Le Corre, Y., Khovratovich, D., Perrin, L., Großschädl, J., Biryukov, A.: Triathlon of lightweight block ciphers for the Internet of things. IACR Cryptology ePrint Archive, p. 209 (2015)
28. Dinu, D., Perrin, L., Udovenko, A., Velichkov, V., Großschädl, J., Biryukov, A.: Design strategies for ARX with provable bounds: SPARX and LAX. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016. LNCS, vol. 10031, pp. 484–513. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53887-6_18
29. Guo, J., Peyrin, T., Poschmann, A., Robshaw, M.: The LED block cipher. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 326–341. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23951-9_22
30. Hong, D., Lee, J.-K., Kim, D.-C., Kwon, D., Ryu, K.H., Lee, D.-G.: LEA: a 128-bit block cipher for fast encryption on common processors. In: Kim, Y., Lee, H., Perrig, A. (eds.) WISA 2013. LNCS, vol. 8267, pp. 3–27. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-05149-9_1
31. Hong, D., et al.: HIGHT: a new block cipher suitable for low-resource device. In: Goubin, L., Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 46–59. Springer, Heidelberg (2006). https://doi.org/10.1007/11894063_4
32. Jean, J., Nikolić, I., Peyrin, T., Wang, L., Wu, S.: Security analysis of PRINCE. In: Moriai, S. (ed.) FSE 2013. LNCS, vol. 8424, pp. 92–111. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43933-3_6
33. Khovratovich, D., Nikolić, I.: Rotational cryptanalysis of ARX. In: Hong, S., Iwata, T. (eds.) FSE 2010. LNCS, vol. 6147, pp. 333–346. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13858-4_19
34. Knudsen, L., Leander, G., Poschmann, A., Robshaw, M.J.B.: PRINTCIPHER: a block cipher for IC-printing. In: Mangard, S., Standaeert, F.-X. (eds.) CHES 2010. LNCS, vol. 6225, pp. 16–32. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15031-9_2
35. Knudsen, L., Wagner, D.: Integral cryptanalysis. In: Daemen, J., Rijmen, V. (eds.) FSE 2002. LNCS, vol. 2365, pp. 112–127. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45661-9_9
36. Kolay, S., Mukhopadhyay, D.: Khudra: a new lightweight block cipher for FPGAs. In: Chakraborty, R.S., Matyas, V., Schaumont, P. (eds.) SPACE 2014. LNCS, vol. 8804, pp. 126–145. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-12060-7_9
37. Koo, B., Hong, D., Kwon, D.: Related-key attack on the full HIGHT. In: Rhee, K.-H., Nyang, D.H. (eds.) ICISC 2010. LNCS, vol. 6829, pp. 49–67. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24209-0_4
38. Matsui, M.: Linear cryptanalysis method for DES cipher. In: Helleseht, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 386–397. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48285-7_33
39. Matsui, M.: On correlation between the order of S-boxes and the strength of DES. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 366–375. Springer, Heidelberg (1995). <https://doi.org/10.1007/BFb0053451>
40. Moradi, A., Poschmann, A., Ling, S., Paar, C., Wang, H.: Pushing the limits: a very compact and a threshold implementation of AES. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 69–88. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20465-4_6
41. Piret, G., Roche, T., Carlet, C.: PICARO – a block cipher allowing efficient higher-order side-channel resistance. In: Bao, F., Samarati, P., Zhou, J. (eds.) ACNS 2012. LNCS, vol. 7341, pp. 311–328. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31284-7_19

42. Poschmann, A.: Lightweight cryptography - cryptographic engineering for a pervasive world. Number 8 in IT Security. Europäischer Universitätsverlag, Published: Ph.D. thesis, Ruhr University Bochum (2009)
43. Shibutani, K., Isobe, T., Hiwatari, H., Mitsuda, A., Akishita, T., Shirai, T.: *Piccolo*: an ultra-lightweight blockcipher. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 342–357. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23951-9_23
44. Shirai, T., Shibutani, K., Akishita, T., Moriai, S., Iwata, T.: The 128-bit blockcipher CLEFIA (extended abstract). In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 181–195. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74619-5_12
45. Song, L., Huang, Z., Yang, Q.: Automatic differential analysis of ARX block ciphers with application to SPECK and LEA. In: Liu, J.K., Steinfeld, R. (eds.) ACISP 2016. LNCS, vol. 9723, pp. 379–394. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-40367-0_24
46. Sun, L., Wang, W., Liu, R., Wang, M.: MILP-aided bit-based division property for ARX-based block cipher, Cryptology ePrint Archive, Report 2016, p. 1101 (2016)
47. Suzaki, T., Minematsu, K., Morioka, S., Kobayashi, E.: *TWINE*: a lightweight block cipher for multiple platforms. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 339–354. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-35999-6_22
48. Todo, Y.: Integral cryptanalysis on full MISTY1. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 413–432. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47989-6_20
49. Todo, Y., Morii, M.: Bit-based division property and application to SIMON family. In: Peyrin, T. (ed.) FSE 2016. LNCS, vol. 9783, pp. 357–377. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-52993-5_18
50. Wagner, D.: The boomerang attack. In: Knudsen, L. (ed.) FSE 1999. LNCS, vol. 1636, pp. 156–170. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48519-8_12
51. Wang, Y., Wu, W., Yu, X., Zhang, L.: Security on LBlock against biclique cryptanalysis. In: Lee, D.H., Yung, M. (eds.) WISA 2012. LNCS, vol. 7690, pp. 1–14. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35416-8_1
52. Wallén, J.: On the differential and linear properties of addition master’s thesis. Helsinki University of Technology, Laboratory for Theoretical Computer Science (2003)
53. Wenzel-Benner, C., Gräf, J.: *XBX*: eXternal benchmarking eXtension for the SUPERCOP crypto benchmarking framework. In: Mangard, S., Standaert, F.-X. (eds.) CHES 2010. LNCS, vol. 6225, pp. 294–305. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15031-9_20
54. Yang, Q., Hu, L., Sun, S., Song, L.: Related-key impossible differential analysis of full *Khudra*. In: Ogawa, K., Yoshioka, K. (eds.) IWSEC 2016. LNCS, vol. 9836, pp. 135–146. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-44524-3_8
55. Yang, G., Zhu, B., Suder, V., Aagaard, M.D., Gong, G.: The Simeck family of lightweight block ciphers. In: Güneysu, T., Handschuh, H. (eds.) CHES 2015. LNCS, vol. 9293, pp. 307–329. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48324-4_16



Improved Meet-in-the-Middle Attacks on Reduced Round Kuznyechik

Mohamed Tolba and Amr M. Youssef^(✉) 

Concordia Institute for Information Systems Engineering,
Concordia University, Montréal, QC, Canada
youssef@ciise.concordia.ca

Abstract. Kuznyechik is an SPN block cipher that has been chosen recently to be standardized by the Russian federation as a new GOST cipher. The cipher employs a 256-bit key which is used to generate ten 128-bit round keys. The encryption procedure updates the 16-byte state by iterating the round function for nine rounds. In this work, we improve the previous 5-round Meet-in-the-Middle (MitM) attack on Kuznyechik by presenting a 6-round attack using the MitM with differential enumeration technique. Unlike previous distinguishers which utilize only the structural properties of the Maximum Distance Separable (MDS) linear transformation layer of the cipher, our 3-round distinguisher is computed based on the exact values of the coefficients of this MDS transformation. More specifically, first, we identified the MDS matrix that is utilized in this cipher. Then, we find all the relations that relate between subset of the inputs and outputs of this linear transformation. Finally, we utilized one of these relations in order to find the best distinguisher that can optimize the time complexity of the attack. Also, instead of placing the distinguisher in the middle rounds of the cipher as in the previous 5-round attack, we place it at the first 3 rounds which allows us to convert the attack from the chosen ciphertext model to the chosen plaintext model. Then, to extend the distinguisher by 3 rounds, we performed the matching between the offline and online phases around the linear transformation instead of matching on a state byte.

Keywords: Cryptanalysis · Meet-in-the-middle attacks
Substitution permutation network · Block ciphers · Kuznyechik
MDS transformations

1 Introduction

Kuznyechik [19] (Grasshopper in Russian) is a substitution permutation network (SPN) block cipher which has been recently selected to be standardized by the Russian federation as a new GOST cipher [2]. The current GOST R 34.12-2015 standard defines the new cipher, Kuznyechik, in addition to the old block cipher GOST 28147-89 [1] which is now named Magma. The cipher employs a 256-bit key which is used to generate ten 128-bit round keys. The encryption procedure updates the 16-byte state by iterating the round function for nine rounds. While

the encryption process of Kuznyechik follows the SPN design architecture, its key schedule employs a Feistel network to generate the round keys.

In the single-key attack model, Kuznyechik has been analyzed in [4, 6]. In [6], they launched multiset-algebraic attack against 6 and 7 rounds of Kuznyechik, but the 7-round attack requires the whole code book. The data, time, and memory complexities of the 6-round attack are 2^{120} chosen plaintexts, $2^{146.5}$ encryptions, and 2^{128} 128-bit blocks, respectively. Furthermore, Biryukov *et al.* [7] revealed a hidden structure in the S-box used in the cipher. A fault analysis attack on the cipher was also presented in [3]. In this paper, we focus on improving the previous MitM attack on the cipher. The MitM attack was first proposed by Diffie and Hellman in 1977 where they applied it to the cryptanalysis of Data Encryption Standard (DES) [14]. In the classical version of this analysis, we split the block cipher E into two sub-ciphers such that $E = G_{K_2} \circ F_{K_1}$, where K_1 and K_2 are two distinct key sets which are used in F and G , respectively. While this MitM attack requires a low data complexity and is considered as one of the major cryptanalysis techniques of block ciphers, finding two distinct key sets, K_1 and K_2 , that cover a large number of rounds is quite challenging, especially in block ciphers that use nonlinear key schedules. The three-subset MitM attack proposed by Bogdanov and Rechberger [8] solves this problem by splitting the key into three-subsets K_1 , K_2 , and K_c such that the key sets K_1 and K_2 may have common key bits that define the set K_c . The attack is then repeated for each possible value of the key bits in K_c . This approach succeeded in attacking the full KTANTAN cipher [8]. Later on, Tolba and Youssef [21] generalized the idea of Bogdanov and Rechberger by allowing the key to be partitioned into $n \geq 3$ subsets which are not restricted to be independent.

Another line of research on the MitM attacks was triggered by Demirci and Selçuk when they were able to attack 8 rounds of both AES-192 and AES-256 [10]. In this attack, the cipher is split into three sub-ciphers such that $E = E_2 \circ E_{mid} \circ E_1$, where E_{mid} exhibits a distinguishing property that is evaluated offline independently of the middle rounds keys. Then, in the online phase, the keys used in E_1 and E_2 are guessed and checked whether they verify the distinguishing property or not. The main downside of this attack is the high memory required to save the precomputation table. Later on, Dunkelman *et al.* [15] proposed two techniques to tackle the issue of the high memory requirement, namely, the differential enumeration and the use of multisets, which helped reduce the memory requirement but not enough to attack AES-128. Afterwards, Derbez *et al.* [12] reduced the memory requirement further by using a rebound-like idea and succeeded in attacking AES-128. Finally, Li *et al.* proposed a key-dependent sieving technique [17] to further reduce the memory complexity of Derbez's attack and presented an attack on 9-round AES-192 and 8-round PRINCE. The MitM attack is not only applied to SPN block ciphers such as AES but also to Feistel ciphers as exemplified by the attacks presented by Guo *et al.* [16] and Lin and Wu [18]. It is worth noting that despite its high memory requirement, the MitM attack based on Demirci and Selçuk technique proves to be quite successful as presented by the recent work against the SPN structure PRINCE [13] and the Feistel constructions TWINE [5], and Piccolo [20].

In this paper, we present 6-round MitM attack on Kuznyechik¹. This attack utilizes a 3-round truncated differential distinguisher that is based on the efficient enumeration and multiset techniques. In particular, in this distinguisher, the match between the offline and online computations is accomplished using a linear relation that relates 5 input bytes and 12 output bytes of the linear transformation layer. In the offline phase, we compute the left hand side of the relation and store the obtained results in a table. Then, in the online phase, we work out the right hand side of the relation and compare it with the stored results in order to filter out the wrong keys. This idea, which is inspired by the work in [11], enables us to extend the 3-round distinguisher by 3 rounds to launch our 6-round MitM attack. Our attack has a time complexity of 2^{231} encryptions, a memory complexity of 2^{218} 128-bit, and a data complexity of 2^{113} chosen plaintexts.

The rest of the paper is organized as follows. Section 2 provides the notations used throughout the paper and a brief description of Kuznyechik. In Sect. 3, we present our attack on 6 rounds of Kuznyechik. Finally, the paper is concluded in Sect. 4.

2 Specifications of Kuznyechik

2.1 Notations

The following notation is used throughout the rest of the paper:

- K : The master key.
- K_i : The 128-bit round key used in the i^{th} round.
- x_i, y_i, z_i : The 16-byte state before the substitution, linear, and key addition operations, respectively, at round i .
- $x_i[j]$: The j^{th} byte of the state x_i , $j = 0, 1, \dots, 15$.
- $x_i[j : l]$: The bytes from j to l of x_i , $j < l$.
- $\Delta x_i, \Delta x_i[j]$: The difference at state x_i , and byte $x_i[j]$, respectively.
- $\cdot, +$: The multiplication and addition in $GF(2^8)$ using the irreducible polynomial $p(x) = x^8 + x^7 + x^6 + x + 1$.
- $||$: The concatenation operation.

2.2 Specifications

Kuznyechik [2, 19] employs the SPN design approach. It has a block length of 128-bit and a secret key of 256 bits. The 256-bit secret key is used to generate 10 128-bit round keys through the key schedule algorithm. The internal state is updated using the round keys by applying the round function 9 times, as illustrated in Fig. 1. The round function has the following operations [4]:

- Nonlinear bijective transformation (S): A nonlinear byte bijective mapping.

¹ In Appendix A, we also show how the attack presented in [4] can be tweaked to work under the chosen plaintext model with less time complexity.

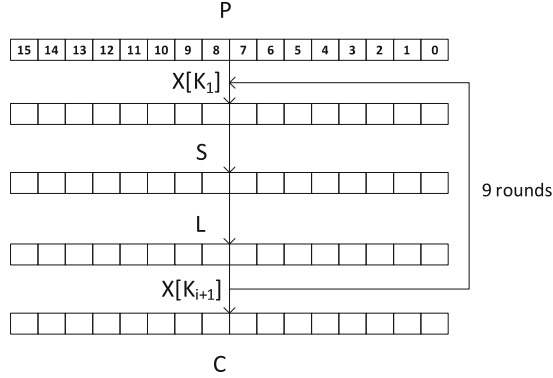


Fig. 1. Encryption scheme

- Linear Transformation (L): An optimal diffusion operation that operates on a 16-byte input and has an optimal branch number = 17. This operation can be defined as $L(a) = R^{16}(a)$, i.e., by iterating the transformation R for 16 times on $a \in \{0, 1\}^{128}$ where

$$R(a) = R(a_{15} || \dots || a_0) = l(a_{15} || \dots || a_0) || a_{15} || \dots || a_1, \quad (1)$$

$$\begin{aligned} l(a_{15}, \dots, a_0) = & 0x94 \cdot a_{15} + 0x20 \cdot a_{14} + 0x85 \cdot a_{13} + 0x10 \cdot a_{12} \\ & + 0xC2 \cdot a_{11} + 0xC0 \cdot a_{10} + a_9 + 0xFB \cdot a_8 + a_7 \\ & + 0xC0 \cdot a_6 + 0xC2 \cdot a_5 + 0x10 \cdot a_4 + 0x85 \cdot a_3 \\ & + 0x20 \cdot a_2 + 0x94 \cdot a_1 + a_0. \end{aligned} \quad (2)$$

- XOR layer (X): Mixes round keys with the encryption state.

Additionally, an XOR layer is added after the last round. Therefore, the encryption algorithm that is used to encrypt the plaintext P to obtain the ciphertext C , can be expressed as follows:

$$C = X[K_{10}] \circ (L \circ S \circ X[K_9]) \circ \dots \circ (L \circ S \circ X[K_1])(P).$$

In our attack, we swap the order of the linear operations L , X , and hence we use an equivalent key EK_r instead of K_r such that $EK_r = L^{-1}(K_r)$.

Key Schedule. The 256-bit secret key is used to generate 10 sub-round keys of length 128 bits each through a 32-round Feistel network. First the master key K is split into two keys K_1, K_2 such that $K = K_1 || K_2$. These keys are used as the first two round keys. Then, these two round keys are updated through the Feistel network that applies the same round function of the encryption on the right branch of the Feistel structure. Finally, as depicted in Fig. 2, each pair of subsequent round keys is extracted after eight rounds of execution. The XOR operation in the round function is done using constants C_i , $i = 1, 2, \dots, 32$, defined as $C_i = L(i)$.

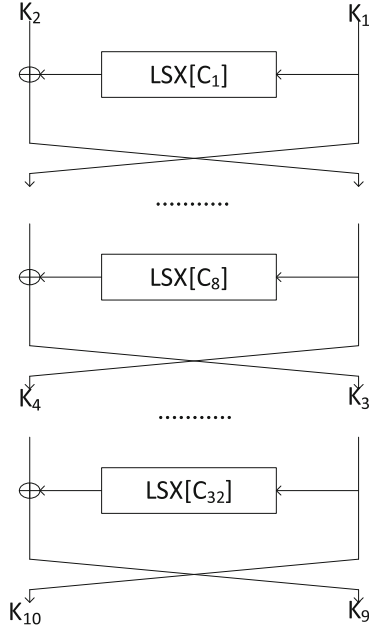


Fig. 2. Key schedule

3 A Differential Enumeration MitM Attack

In the MitM attack, the block cipher E is split into three sub-ciphers E_1 , E_{mid} and E_2 such that $E = E_2 \circ E_{mid} \circ E_1$ where the middle sub-cipher E_{mid} exhibits a distinguishing property evaluated from a particular set of messages. The distinguishing property is evaluated in the *offline phase* independent of middle round keys. Afterwards, in the *online phase*, the round keys used in the analysis rounds of the sub-ciphers E_1 and E_2 are guessed and checked whether they satisfy the distinguishing property or not. The round keys that satisfy the distinguishing property are considered as key candidates and the remaining keys are discarded. The chosen property in our attacks is a truncated differential characteristic whose input is a δ -set [9] captured by Definition 1 and its output is a parameterized function of the input. To identify the key candidates, we use the multisets (see Definition 2) of the outputs corresponding to the δ -set as a distinguishing property.

Definition 1 (δ -set). A δ -set for a byte-oriented cipher is a set of 256 state values that are all different in one byte (the active byte) and equal in the remaining bytes (the inactive bytes).

Definition 2 (*Multisets of bytes*). A multiset generalizes the set concept by allowing elements to appear more than once. In our case, a multiset of 256 bytes can take as many as $\binom{2^8+2^8-1}{2^8} \approx 2^{506.17}$ different values.

Our attack depends on the following proposition:

Proposition 1 (*Differential Property of S*). Given two nonzero differences Δi and Δo in \mathbb{F}_{256} , the equation: $S(x)+S(x+\Delta i) = \Delta o$ has one solution on average. This property also applies to S^{-1} .

3.1 MitM Attack on 6-Round Kuznyechik

The distinguishing property used in our attack is based on a relation between 5 input bytes and 12 output bytes of the linear transformation L , which can be captured by the following proposition:

Proposition 2. The linear transformation L that transform the input $a = a_{15}||a_{14}||\dots||a_0$ to the output $b = b_{15}||b_{14}||\dots||b_0$ has the following property:

$$\begin{aligned} & a_{11} + 0x94 \cdot a_{12} + 0x20 \cdot a_{13} + 0x85 \cdot a_{14} + 0x10 \cdot a_{15} = \\ & 0xC2 \cdot b_0 + 0xC0 \cdot b_1 + b_2 + 0xFB \cdot b_3 + b_4 + 0xC0 \cdot b_5 + \\ & 0xC2 \cdot b_6 + 0x10 \cdot b_7 + 0x85 \cdot b_8 + 0x20 \cdot b_9 + 0x94 \cdot b_{10} + b_{11}. \end{aligned} \tag{3}$$

Proposition 2 is obtained by first using Eqs. (1) and (2) to calculate the equivalent 16×16 MDS matrix, A , that transforms the input a to the output b . Thus we have²

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \\ b_8 \\ b_9 \\ b_{10} \\ b_{11} \\ b_{12} \\ b_{13} \\ b_{14} \\ b_{15} \end{pmatrix} = \begin{pmatrix} 1 & 94 & 20 & 85 & 10 & C2 & C0 & 1 & FB & 1 & C0 & C2 & 10 & 85 & 20 & 94 \\ 94 & A5 & 3C & 44 & D1 & 8D & B4 & 54 & DE & 6F & 77 & 5D & 96 & 74 & 2D & 84 \\ 84 & 64 & 48 & DF & D3 & 31 & A6 & 30 & E0 & 5A & 44 & 97 & CA & 75 & 99 & DD \\ DD & 0D & F8 & 52 & 91 & 64 & FF & 7B & AF & 3D & 94 & F3 & D9 & D0 & E9 & 10 \\ 10 & 89 & 48 & 7F & 91 & EC & 39 & EF & 10 & BF & 60 & E9 & 30 & 5E & 95 & BD \\ BD & A2 & 48 & C6 & FE & EB & 2F & 84 & C9 & AD & 7C & 1A & 68 & BE & 9F & 27 \\ 27 & 7F & C8 & 98 & F3 & 0F & 54 & 8 & F6 & EE & 12 & 8D & 2F & B8 & D4 & 5D \\ 5D & 4B & 8E & 60 & 1 & 2A & 6C & 9 & 49 & AB & 8D & CB & 14 & 87 & 49 & B8 \\ B8 & 6E & 2A & D4 & B1 & 37 & AF & D4 & BE & F1 & 2E & BB & 1A & 4E & E6 & 7A \\ 7A & 16 & F5 & 52 & 78 & 99 & EB & D5 & E7 & C4 & 2D & 6 & 17 & 62 & D5 & 48 \\ 48 & C3 & 2 & 0E & 58 & 90 & E1 & A3 & 6E & AF & BC & C5 & 0C & EC & 76 & 6C \\ 6C & 4C & DD & 65 & 1 & C4 & D4 & 8D & A4 & 2 & EB & 20 & CA & 6B & F2 & 72 \\ 72 & E8 & 14 & 7 & 49 & F6 & D7 & A6 & 6A & D6 & 11 & 1C & 0C & 10 & 33 & 76 \\ 76 & E3 & 30 & 9F & 6B & 30 & 63 & A1 & 2B & 1C & 43 & 68 & 70 & 87 & C8 & A2 \\ A2 & D0 & 44 & 86 & 2D & B8 & 64 & C1 & 9C & 89 & 48 & 90 & DA & C6 & 20 & 6E \\ 6E & 4D & 8E & EA & A9 & F6 & BF & 0A & F3 & F2 & 8E & 93 & BF & 74 & 98 & CF \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ a_{10} \\ a_{11} \\ a_{12} \\ a_{13} \\ a_{14} \\ a_{15} \end{pmatrix}$$

Then by performing Gauss elimination on the augmented matrix $(A||b)$, we obtain

² All the matrix coefficients $\in GF(2^8)$ and are expressed in hexadecimal notation.

$$\begin{pmatrix}
 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 94 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 20 & 94 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 85 & 20 & 94 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 10 & 85 & 20 & 94 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 C2 & 10 & 85 & 20 & 94 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 C0 & C2 & 10 & 85 & 20 & 94 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 1 & C0 & C2 & 10 & 85 & 20 & 94 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 FB & 1 & C0 & C2 & 10 & 85 & 20 & 94 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 1 & FB & 1 & C0 & C2 & 10 & 85 & 20 & 94 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
 C0 & 1 & FB & 1 & C0 & C2 & 10 & 85 & 20 & 94 & 1 & 0 & 0 & 0 & 0 & 0 \\
 C2 & C0 & 1 & FB & 1 & C0 & C2 & 10 & 85 & 20 & 94 & 1 & 0 & 0 & 0 & 0 \\
 10 & C2 & C0 & 1 & FB & 1 & C0 & C2 & 10 & 85 & 20 & 94 & 1 & 0 & 0 & 0 \\
 85 & 10 & C2 & C0 & 1 & FB & 1 & C0 & C2 & 10 & 85 & 20 & 94 & 1 & 0 & 0 \\
 20 & 85 & 10 & C2 & C0 & 1 & FB & 1 & C0 & C2 & 10 & 85 & 20 & 94 & 1 & 0 \\
 94 & 20 & 85 & 10 & C2 & C0 & 1 & FB & 1 & C0 & C2 & 10 & 85 & 20 & 94 & 1
 \end{pmatrix}
 \begin{pmatrix}
 b_0 \\
 b_1 \\
 b_2 \\
 b_3 \\
 b_4 \\
 b_5 \\
 b_6 \\
 b_7 \\
 b_8 \\
 b_9 \\
 b_{10} \\
 b_{11} \\
 b_{12} \\
 b_{13} \\
 b_{14} \\
 b_{15}
 \end{pmatrix}
 =
 \begin{pmatrix}
 1 & 94 & 20 & 85 & 10 & C2 & C0 & 1 & FB & 1 & C0 & C2 & 10 & 85 & 20 & 94 \\
 0 & 1 & 94 & 20 & 85 & 10 & C2 & C0 & 1 & FB & 1 & C0 & C2 & 10 & 85 & 20 \\
 0 & 0 & 1 & 94 & 20 & 85 & 10 & C2 & C0 & 1 & FB & 1 & C0 & C2 & 10 & 85 \\
 0 & 0 & 0 & 1 & 94 & 20 & 85 & 10 & C2 & C0 & 1 & FB & 1 & C0 & C2 & 10 \\
 0 & 0 & 0 & 0 & 1 & 94 & 20 & 85 & 10 & C2 & C0 & 1 & FB & 1 & C0 & C2 \\
 0 & 0 & 0 & 0 & 0 & 1 & 94 & 20 & 85 & 10 & C2 & C0 & 1 & FB & 1 & C0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 1 & 94 & 20 & 85 & 10 & C2 & C0 & 1 & FB & 1 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 94 & 20 & 85 & 10 & C2 & C0 & 1 & FB \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 94 & 20 & 85 & 10 & C2 & C0 & 1 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 94 & 20 & 85 & 10 & C2 & C0 & 1 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 94 & 20 & 85 & 10 & C2 & C0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 94 & 20 & 85 & 10 & C2 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 94 & 20 & 85 & 10 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 94 & 20 & 85 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 94 & 20 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 94 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
 \end{pmatrix}
 \begin{pmatrix}
 a_0 \\
 a_1 \\
 a_2 \\
 a_3 \\
 a_4 \\
 a_5 \\
 a_6 \\
 a_7 \\
 a_8 \\
 a_9 \\
 a_{10} \\
 a_{11} \\
 a_{12} \\
 a_{13} \\
 a_{14} \\
 a_{15}
 \end{pmatrix}
 \quad (4)$$

It should be noted that this Gauss elimination step allows us to obtain many linear relations that relate subsets of the input and output bytes of the linear transformation, however, the relation identified by Proposition 2, which corresponds to the bold line in Eq. (4) above, helped us to reduce the total time complexity of the attack.

As depicted in Fig. 3, the distinguisher starts at x_1 and ends at y_4 . The δ -set is chosen at byte 15 and the multiset is computed from the left hand side of Eq. (3) using bytes 11, 12, 13, 14, 15. Our distinguisher is based on the following proposition:

Proposition 3. *If a message m belongs to a pair of states conforming to the truncated differential characteristic shown in Fig. 3, then the multiset of differences $y_4[11] + 0x94 \cdot y_4[12] + 0x20 \cdot y_4[13] + 0x85 \cdot y_4[14] + 0x10 \cdot y_4[15]$ obtained from the δ -set constructed from m in $x_1[15]$ is fully determined by the following 27 bytes: $\Delta x_1[15], x_2, y_4[11 : 15]$ and $\Delta y_4[11 : 15]$.*

Proof. The proof is based on the efficient enumeration technique [12]. In the following, we show how the knowledge of 27 bytes is enough to propagate the δ -set at x_1 and compute the multiset $(y_4[11] + 0x94 \cdot y_4[12] + 0x20 \cdot y_4[13] + 0x85 \cdot y_4[14] + 0x10 \cdot y_4[15])$. Let (m, m') be a right pair that conforms to the truncated differential characteristic in Fig. 3. Since we are interested in the multiset instead of the ordered sequence and the S-box that is used is bijective, then we can propagate the difference $\Delta y_1[15]$ instead of $\Delta x_1[15]$. $\Delta y_1[15]$ can be propagated through the linear operations L , and X to compute Δx_2 . With the knowledge of x_2 , we can bypass the non-linear operation S . Then we can forward the knowledge through L, X to get Δx_3 . Similarly, in the backward direction, the knowledge of $y_4[11 : 15]$ allows the propagation of $\Delta y_4[11 : 15]$ through the non-linear mapping S^{-1} to get $\Delta x_4[11 : 15]$. Then, we propagate the difference $\Delta x_4[11 : 15]$ backward linearly through X, L^{-1} to determine Δy_3 . Using the

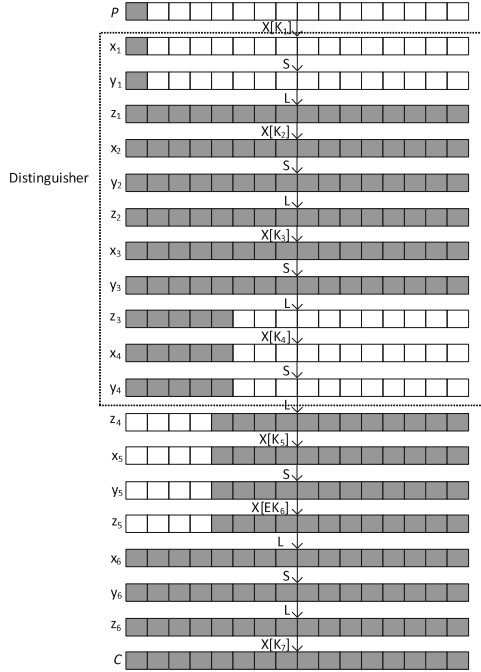


Fig. 3. Kuznyechik 6-round attack

differential property of Kuznyechik S-box, the expected number of solutions for x_3 is one. \square

The previous 3-round distinguisher is utilized to launch a 6-round attack on Kuznyechik by appending 3 rounds below it. In what follows, we describe the details of our attack which has two phases, namely the offline and online phases.

Offline Phase. In this phase, we build the distinguishing property that will be used in the online phase to filter the key space. As mentioned above, our distinguisher can be built using only 27 parameters given by Proposition 3. First, we iterate on the $2^{27 \times 8} = 2^{216}$ possible values for each of the 27 byte parameters, to deduce the internal state variable x_3 . Then, using the obtained value of x_3 , we propagate the δ -set to compute the multiset differences $y_4[11] + 0x94 \cdot y_4[12] + 0x20 \cdot y_4[13] + 0x85 \cdot y_4[14] + 0x10 \cdot y_4[15]$ and store them in a table. Consequently, we have 2^{216} multisets out of the $2^{467.6}$ theoretically possible ones (not $2^{506.17}$ because the number of ordered sequences associated to a multiset is not constant [12]).

Online Phase. In this phase, we have two steps. First, the *data collection* step where we want to ensure that we have one pair of messages that conform to the truncated differential characteristic in Fig. 3, which holds with probability p . This is achieved by collecting $1/p$ pair of messages. Second, the *key recovery* where we

first identify the key values that satisfy the truncated differential characteristic in Fig. 3. Then these key values with the collected data pairs are used to create the δ -set, compute the multiset in the online phase and compare it with the precomputation table to determine the valid key candidates.

Data Collection. We employed our δ -set in the plaintext side to operate in the chosen plaintext model. We also utilize plaintext structures to reduce the amount of required plaintexts. The utilized structure takes all the possible values in byte 15 while the remaining bytes take fixed value. Therefore, one structure generates $2^8 \times (2^8 - 1)/2 \approx 2^{15}$ possible pairs. The probability of the whole truncated differential characteristic of Fig. 3 can be calculated from the following probabilities: transition from x_6 to z_5 over L^{-1} ($16 \rightarrow 12$) of probability $2^{-4 \times 8} = 2^{-32}$ and transition from z_4 to y_4 over L^{-1} ($12 \rightarrow 5$) of probability $2^{-11 \times 8} = 2^{-88}$. Therefore, the total probability of the truncated differential characteristic is 2^{-120} . Hence, to find one pair of messages that conform to the truncated differential characteristic, we need to collect 2^{120} message pairs. Since each structure contains 2^{15} message pairs, 2^{105} structures will suffice to find the right pair. Therefore, the total number of queries to the encryption oracle is about 2^{113} .

Key Recovery. This step involves two sub steps. First, identifying the key suggestions that will be used in building and computing the δ -set and the multiset, respectively. Second, checking whether the identified keys in the first sub step are key candidates. The keys that are used in building the δ -set and computing the multiset are $EK_6[0 : 11], K_7$, i.e., we want to guess 28 bytes which will make the computational complexity exceeds the exhaustive search. Instead, we identify the number of key suggestions of the 28 bytes that correspond to each pair of messages. This can be achieved as follows: to deduce the values of the 16 bytes of K_7 , we guess 12 bytes of $\Delta y_5[0 : 11]$ and propagate these values linearly through X, L to get Δx_6 . The knowledge of the ciphertext allows us to compute Δy_6 . Using the differential property of the S-box, we evaluate y_6 . Then, we compute z_6 from y_6 through the linear operation L . The knowledge of the ciphertext and z_6 allows us to deduce the values of the 16 bytes of K_7 . For the next 12 bytes of $EK_6[0 : 11]$, the 5 bytes of $\Delta y_4, \Delta y_4[11 : 15]$ can take 2^{40} values, but only 2^8 values among them can follow the truncated differential characteristic and after applying L , Δz_4 has only 4 zero bytes. Therefore, to deduce the values of the 12 bytes $EK_6[0 : 11]$, we guess the 2^8 values for $\Delta y_4[11 : 15]$ and propagate them linearly forward through L, X to compute $\Delta x_5[0 : 11]$. The knowledge of $\Delta x_5[0 : 11]$ and $\Delta y_5[0 : 11]$ allows us to deduce the value of 12 bytes $y_5[0 : 11]$ using the differential property of the S-box. We can also propagate y_6 through S^{-1} and L^{-1} to get $z_5[0 : 11]$. The knowledge of $z_5[0 : 11]$ and $y_5[0 : 11]$ allows to deduce $EK_6[0 : 11]$. To summarize this part, we have $2^{13 \times 8} = 2^{104}$ key suggestions for the 28 key bytes $EK_6[0 : 11], K_7$.

Now, we can use the 2^{120} collected message pairs and the 2^{104} suggestions for the 28 key bytes $EK_6[0 : 11], K_7$ to identify the δ -set and compute the multiset constructed from $0xC2 \cdot z_4[0] + 0xC0 \cdot z_4[1] + z_4[2] + 0xFB \cdot z_4[3] + z_4[4] + 0xC0 \cdot z_4[5] + 0xC2 \cdot z_4[6] + 0x10 \cdot z_4[7] + 0x85 \cdot z_4[8] + 0x20 \cdot z_4[9] + 0x94 \cdot z_4[10] + z_4[11]$.

Consequently, the key suggestion is considered wrong if no match is found in the table. Otherwise, it is considered as a key candidate. The probability of a wrong key leading to a match in the table is $2^{216}/2^{467.6} = 2^{-251.6}$. Therefore, after this step we will have $2^{120+104-251.6} = 2^{-27.6}$ key candidates for the key bytes $EK_6[0 : 11], K_7$. In other words, only the right candidate will pass this filtering step. From the key schedule of Kuznyechik, the master key can be recovered by guessing K_8 . Then, we can propagate the knowledge of K_7 and K_8 until we evaluate K_1 and K_2 , and hence the master key $K = K_1 || K_2$. Therefore, retrieving the master key requires to exhaustively search the remaining key candidate of K_7 with the 2^{128} possible values of K_8 using two plaintext/ciphertext pairs.

Attack Complexity. The offline phase dominates the memory complexity of the attack in which we store 2^{216} multiset where each multiset is 512 bits. Therefore, the memory complexity of the attack is $2^{216} \times 512/128 = 2^{218}$ 128-bit blocks. The data collection step which determines the data complexity of the attack requires us to collect 2^{113} plaintext to generate 2^{120} message pairs. Hence the data complexity of the attack is 2^{113} chosen plaintexts. The time complexity of the attack has three main components. First, the time needed to store the precomputation table, is $2^{27 \times 8} \times 2^8 \times 3/6 = 2^{223}$ encryptions. Second, the time required to get the one key candidate for K_7 is $2^{120} \times 2^{13 \times 8} \times 2^8 \times 3/6 = 2^{231}$ encryptions. Third, the time required to retrieve the master key is $2 \times 2^{128} = 2^{129}$. Therefore, the time complexity of the attack is about 2^{231} encryptions.

4 Conclusion

In this work, we presented a Meet-in-the-Middle attack on 6-round reduced Kuznyechik. By exploiting the exact values of the coefficients of the MDS transformation of the cipher, our attack recovers the master key with data complexity of 2^{113} chosen plaintexts, time complexity of 2^{231} encryptions, and memory complexity of 2^{218} 128-bit blocks. This attack improves the previous results of the MitM with differential enumeration technique which can only reach 5 rounds. Compared to the best known attack which does not require the full code book on this cipher in the single-key model, and which also reaches 6 rounds, our attack improves the data complexity at the expense of requiring more time and memory complexities.

A Chosen Plaintext MitM Attack on 5-Round Kuznyechik

The authors in [4] implied that their attack can only work in the chosen ciphertext model. In this appendix, we show how we can tweak their attack to work in the plaintext model. Figure 4 illustrates our 3-round distinguisher which starts at x_1 and ends at y_4 . The δ -set is chosen at byte 15 and the multiset is computed at byte 15. Our distinguisher is based on the following proposition:

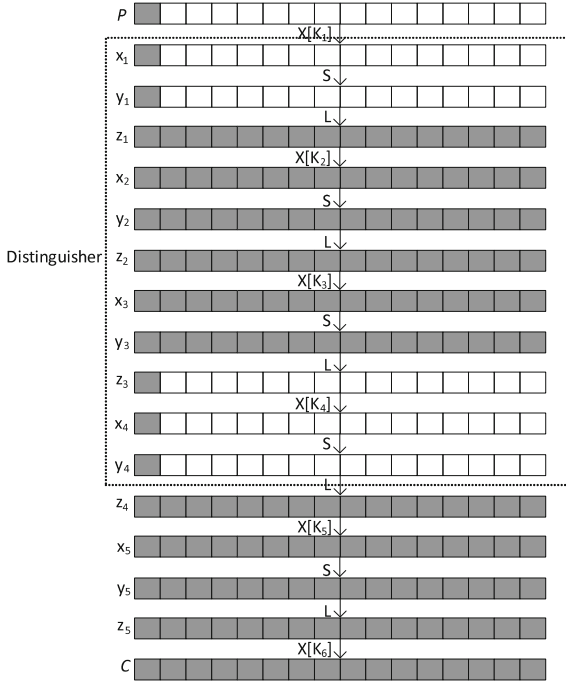


Fig. 4. Kuznyechik 5-round attack

Proposition 4. *If a message m belongs to a pair of states conforming to the truncated differential characteristic of Fig. 4, then the multiset of differences $\Delta y_4[15]$ obtained from the δ -set constructed from m in $x_1[15]$ is fully determined by the following 19 bytes: $\Delta x_1[15], x_2, y_4[15]$ and $\Delta y_4[15]$.*

Proposition 4 can be proved using the same approach used to prove Proposition 3.

Offline Phase. In this phase, we compute the multiset at $y_4[15]$ using the 19 byte parameters mentioned in Proposition 4, i.e., we have $2^{19 \times 8} = 2^{152}$ multiset out of $2^{467.6}$ theoretically possible ones.

Data Collection. The probability of the truncated differential characteristic can be evaluated as follows: transition from z_4 to y_4 over L^{-1} ($16 \rightarrow 1$) of probability $2^{-15 \times 8} = 2^{-120}$. Therefore, we need to collect 2^{120} message pairs to guarantee that there exist one message pair which conform to the truncated path. We use the same structure that is used in the 6-round attack. Hence, we need to query 2^{113} chosen plaintext.

Key Recovery. In order to build the δ -set and compute the multiset, we need to guess K_6 . The key suggestions for the 16 bytes K_6 can be obtained by guessing $\Delta y_4[15]$. Therefore, we have 2^8 values for the 16 bytes key K_6 .

The probability of finding a match in the table with the wrong key is $2^{152} / 2^{467.6} = 2^{-315.6}$. Therefore, the number of key candidates of K_6 after

launching the attack is $2^{120+8-315.6} = 2^{-187.6}$, i.e., our attack will find one right value for K_6 . Then, the master key can be recovered by guessing K_5 using two plaintext/ciphertext pairs.

Attack complexity. The memory complexity is $2^{152} \times 512/128 = 2^{154}$ 128-bit. The data complexity is 2^{113} chosen plaintext. The time complexity is $2^{19 \times 8} \times 2^8 \times 3/5 + 2^{120} \times 2^8 \times 2^8 \times 2/5 + 2 \times 2^{128} \approx 2^{159.3} + 2^{134.7} + 2^{129} \approx 2^{159.3}$ encryptions. Our attack has an online time complexity of $2^{134.7}$ encryptions. Therefore, this attack reduces the online time complexity of the previous attack [4] by a factor of $2^{5.6}$ with the same data and a non significant increase in the memory complexity.

References

1. GOST 28147-89. Information Processing Systems. Cryptographic Protection. Cryptographic Transformation Algorithm (in Russian)
2. The National Standard of the Russian Federation GOST R 34.11-2012. Russian Federal Agency on Technical Regulation and Metrology report (2015)
3. AlTawy, R., Duman, O., Youssef, A.M.: Fault analysis of kuznyechik. IACR Cryptology ePrint Archive, 2015/347 (2015). <https://eprint.iacr.org/2015/347.pdf>
4. AlTawy, R., Youssef, A.M.: A meet in the middle attack on reduced round Kuznyechik. IEICE Trans. **98**–A(10), 2194–2198 (2015)
5. Biryukov, A., Derbez, P., Perrin, L.: Differential analysis and meet-in-the-middle attack against round-reduced TWINE. In: Leander, G. (ed.) FSE 2015. LNCS, vol. 9054, pp. 3–27. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48116-5_1
6. Biryukov, A., Khovratovich, D., Perrin, L.: Multiset-algebraic cryptanalysis of reduced Kuznyechik, Khazad, and secret SPNs. IACR Trans. Symmetric Cryptol. **2016**(2), 226–247 (2017)
7. Biryukov, A., Perrin, L., Udovenko, A.: Reverse-engineering the S-Box of Streebog, Kuznyechik and STRIBOBr1. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9665, pp. 372–402. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49890-3_15
8. 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). https://doi.org/10.1007/978-3-642-19574-7_16
9. Daemen, J., Knudsen, L., Rijmen, V.: The block cipher Square. In: Biham, E. (ed.) FSE 1997. LNCS, vol. 1267, pp. 149–165. Springer, Heidelberg (1997). <https://doi.org/10.1007/BFb0052343>
10. Demirci, H., Selçuk, A.A.: A meet-in-the-middle attack on 8-round AES. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 116–126. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-71039-4_7
11. Derbez, P., Fouque, P.-A.: Exhausting Demirci-Selçuk meet-in-the-middle attacks against reduced-round AES. In: Moriai, S. (ed.) FSE 2013. LNCS, vol. 8424, pp. 541–560. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43933-3_28

12. Derbez, P., Fouque, P.-A., Jean, J.: Improved key recovery attacks on reduced-round AES in the single-key setting. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 371–387. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38348-9_23
13. Derbez, P., Perrin, L.: Meet-in-the-middle attacks and structural analysis of round-reduced PRINCE. In: Leander, G. (ed.) FSE 2015. LNCS, vol. 9054, pp. 190–216. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48116-5_10
14. Diffie, W., Hellman, M.E.: Special feature exhaustive cryptanalysis of the NBS data encryption standard. *Computer* **10**(6), 74–84 (1977)
15. Dunkelman, O., Keller, N., Shamir, A.: Improved single-key attacks on 8-round AES-192 and AES-256. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 158–176. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17373-8_10
16. Guo, J., Jean, J., Nikolić, I., Sasaki, Y.: Meet-in-the-middle attacks on generic feistel constructions. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8873, pp. 458–477. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45611-8_24
17. Li, L., Jia, K., Wang, X.: Improved meet-in-the-middle attacks on AES-192 and PRINCE. IACR Cryptology ePrint Archive, 2013/573 (2013). <https://eprint.iacr.org/2013/573.pdf>
18. Lin, L., Wu, W.: Improved meet-in-the-middle distinguisher on Feistel schemes. IACR Cryptology ePrint Archive, 2015/051 (2015). <https://eprint.iacr.org/2015/051.pdf>
19. Shishkin, V., Dygin, D., Lavrikov, I., Marshalko, G., Rudskoy, V., Trifonov, D.: Low-Weight and Hi-End: Draft Russian Encryption Standard, pp. 183–188 (2014)
20. Tolba, M., Abdelkhalek, A., Youssef, A.M.: Meet-in-the-middle attacks on reduced round piccolo. In: Güneysu, T., Leander, G., Moradi, A. (eds.) LightSec 2015. LNCS, vol. 9542, pp. 3–20. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-29078-2_1
21. Tolba, M., Youssef, A.M.: Generalized MitM attacks on full TWINE. *Inf. Process. Lett.* **116**, 128–135 (2016)



Security of Stateful Order-Preserving Encryption

Kee Sung Kim^(✉), Minkyu Kim, Dongsoo Lee,
Je Hong Park, and Woo-Hwan Kim

National Security Research Institute, Daejeon, South Korea
{keesung2137, mkkim, letrhee, jhpark, whkim5}@nsr.re.kr

Abstract. Most of the proposed order-preserving encryption (OPE) schemes in the early stage of development including the first provably secure one are *stateless* and work efficiently, but guarantee only weak security. Additionally, subsequent works have shown that an ideal security notion IND-OCPA can be achieved using statefulness, ciphertexts mutability, and interactivity between client and server. Though such properties hinder availability of IND-OCPA secure OPE schemes, the only definitively known result is the impossibility of constructing a feasible IND-OCPA secure OPE scheme without ciphertext mutability. In this work, we study the security that can be fulfilled by only statefulness, from a viewpoint different from the existing research. We first consider a new security notion, called δ -IND-OCPA, which is a natural relaxation of IND-OCPA. In comparison to IND-OCPA in which ciphertexts reveal no additional information beyond the order of the plaintexts, our notion can quantify the rate of plaintext bits that are leaked. To show achievability of our notion, we construct a new δ -IND-OCPA secure OPE scheme. The proposed scheme is stateful and non-interactive, but does not require ciphertext mutation. Through several experiments, we show that our construction is also feasible and that has an advantage in the correlation analysis compared with the IND-OCPA secure scheme.

Keywords: Order-preserving encryption · Outsourced database
Cloud computing

1 Introduction

Cloud storage services provide the ability to accumulate large amount of data and make it possible to share and synchronize them across devices. Considering the sensitivity of such data and the confidence in the cloud servers, a significant amount of data should be encrypted before being transferred to the cloud servers, and stored on those servers without decryption. While traditional encryption is a powerful tool of protecting the confidentiality of stored data, its all-or-nothing approach to data access destroys properties of plaintexts that are necessary for efficient server-side processing, such as order comparison. One particularly efficient solution for performing useful operations on encrypted data is to use an

order-preserving encryption (OPE) scheme [5, 12, 13, 16, 18]. OPE is a symmetric encryption that results in ciphertexts which preserve the numerical orders of the corresponding plaintexts. Although an OPE scheme cannot achieve a natural indistinguishable notion of semantic security [5], it allows order comparison operations, including matching, range, sorting, ranking, etc., directly on ciphertexts. So any possible environment that makes an SQL query on encrypted data such as e-mail, web applications, database security solutions, and cloud storage services can be applications of OPE.

The notion of ideal security for OPE, called Indistinguishability under ordered chosen-plaintext attack (IND-OCPA)¹ was first formalized by Boldyreva et al. [5], but the first concrete IND-OCPA secure scheme was proposed a few years later by Popa et al. [16]. Back in [5], the authors showed that stateless OPE schemes cannot achieve ideal security under reasonable assumptions. To construct a deployable and provably secure OPE scheme, they set up a weaker security goal called pseudorandom order-preserving function security under chosen-ciphertext attack (POPF-CCA) in which the oracle access to the encryption algorithm of an OPE scheme is indistinguishable from that to a random order-preserving function under chosen-ciphertext attacks. Later, it was proven that POPF-CCA secure OPE schemes reveal *at least* half of the most significant bits of plaintexts [6]. As an alternative to POPF-CCA, Teranishi et al. proposed another security goal for OPE of θ -lsb-KPA: guaranteeing the secrecy of a fraction θ of the least significant bits of a plaintext under known plaintext attacks [18].

Instead of striving for a weaker notion of security, Popa et al. utilized the client-side inner state (statefulness) and ciphertext mutability, and proved impossibility of achieving IND-OCPA by the statefulness alone [16]. Here, ciphertext mutability means that whenever a new plaintext is encrypted the existing ciphertexts can be updated. Because the states are encrypted and stored in a tree data structure at the server, the encryption process is *interactive*, requiring multiple rounds of communication between the client and the server. Such approach has a problem that the entire ciphertexts should be updated at a certain point in time depending on the distribution of plaintexts. Although Kerschbaum and Schröepfer [13] have shown that there is little update when plaintexts follow uniform distribution. However, it only take into accounts an ideal case that favorable to them, and there are many applications whose plaintext distribution is non-uniform. For example, we can consider the worst-case scenario in which the plaintext is inserted one by one according to their order.

Before describing the contribution of this paper, we briefly compare the characteristics of the existing OPE schemes in Table 1. As evaluation points, the columns “Client storage” and “Computation” denote the size of secret information and computational complexity of encrypting data at the client-side, respectively. The “Communication” column denotes the (worst-case) communication cost between client and server when encrypting a plaintext. The “Update” column denotes the cost of re-encrypting ciphertexts at the server-side DBMS, but

¹ See Appendix A.

Table 1. Comparison with the previous OPE schemes

Scheme	Client storage	#Round	Computation	Communication	Update	Ref.
BCLO	$O(1)$	-	$O(\log M)$ -HDS	$O(1)$	-	[5]
TYM	$O(1)$	-	$O(\log M)$ -HDS/BDS	$O(1)$	-	[18]
PLZ	$O(1)$	$O(\log M)$	$O(\log M)$ -Dec	$O(\log M)$	$O(M)$	[16]
KS	$O(M)$	-	$O(\log M)$ -TL/SAO	$O(M/\log M)$	$O(M)$	[13]
Ours	$O(M)$	-	$O(\log M)$ -TL/SAO	$O(1)$	-	

M : Message Space Size, SAO: Simple Arithmetic Operation,

HDS: Hypergeometric Distribution Sampling, BDS: Binomial Distribution Sampling,

Dec: Decryption, TL: Table Lookup

it generally occurs sporadically. For example, the re-encryption process of the OPE schemes proposed in [12, 13] are occurred every $O(\log M)$ times in the worst case, such as encrypting strictly increasing or decreasing data set. Known stateful OPE schemes [12, 13, 16] with ciphertext mutability provide ideal security but at least one of the efficiency factors is harmed. Note that such constructions have focused on achieving IND-OCPA security using ciphertext mutability, but the upper limit of the achievable security without this property is still unknown.

As an approach to reduce this gap, we first consider a new security notion of OPE, called δ -IND-OCPA. Our security notion is similar to the partial indistinguishability notion of [18] but considers adaptive adversaries [10] and provides stronger security goal that can quantify the rate of plaintext bits to be leaked. By such properties, we can show that δ -IND-OCPA implies θ -lsb-KPA [18] under any plaintext distribution (relation between θ and δ is explained later), and it is identical to IND-OCPA when $\delta = 1$. We also provide a stateful but non-mutable OPE (hereinafter referred to as “sOPE”) scheme which achieves this new security notion. Note that the stateless OPE scheme in [18] does not guarantee security goal in the adaptive CPA model, and so we use the client-side state to counter such stronger attack model. Finally, we implement our scheme to show its feasibility by comparing with the existing schemes [5, 13]. When a ciphertext re-encryption phase via update operation is occurred in the IND-OCPA secure scheme, we can confirm that our sOPE scheme is faster. We also show that our sOPE scheme has lower correlation between the ciphertext and plaintext than the IND-OCPA secure scheme for plaintexts with normal distribution.

As related works, Boneh et al. introduced a related primitive called order-revealing encryption (ORE), which is a generalization of OPE [7]. The ORE scheme does not preserve the numerical ordering of the plaintexts in ciphertexts, but there is a public comparison function that takes two ciphertexts and outputs the numeric ordering of the underlying plaintexts. Although their ORE scheme is IND-OCPA secure, it relies on multilinear maps, which accompany heavy computations and strong assumptions. Recently, N. Chenette et al. showed that a practical ORE scheme leaks some information about the relative distance between the underlying plaintexts [8].

2 Preliminaries

2.1 Notations

Let $\lambda \in \mathbb{N}$ denote the security parameter. Given positive integers α and β with $\alpha \leq \beta$, $[\alpha, \beta]$ denotes the set $\{\alpha, \alpha + 1, \dots, \beta\}$ and $[\alpha] = [1, \alpha]$. Let $|\llbracket \alpha, \beta \rrbracket|$ denote the number of elements in the interval.

A function $\nu : \mathbb{N} \rightarrow \mathbb{N}$ is negligible in k if for every positive polynomial $p(\cdot)$ there exists $K \in \mathbb{R}$ such that $\nu(k) < 1/p(k)$ for all $k \geq K$. We write $f(k) = \text{negl}(k)$ to mean that there exists a negligible function $\nu(\cdot)$ such that $f(k) \leq \nu(k)$ for all sufficiently large k . We write $\text{poly}(k)$ to denote a polynomial in k . Let $\lceil x \rceil$ denote the $\min\{n \in \mathbb{N} \mid n \geq x\}$ and $\lfloor x \rfloor$ denote the $\max\{n \in \mathbb{N} \mid n \leq x\}$. The output x of an algorithm \mathcal{A} is denoted by $x \leftarrow \mathcal{A}$.

2.2 Basic Primitive

A symmetric key encryption scheme SE consists of three polynomial-time algorithms $\text{SE} = (\text{Kg}, \text{Enc}, \text{Dec})$ such that SE.Kg takes a security parameter λ in unary and returns a secret key skey ; SE.Enc takes a key skey and a message m and returns a ciphertext c ; SE.Dec takes a key skey and a ciphertext c and returns m if skey was the key under which c was produced. Here, SE.Kg is probabilistic, but SE.Enc and SE.Dec are deterministic. Informally, a symmetric key encryption scheme is considered CPA secure if a ciphertext do not leak any information about the corresponding plaintext even to an adversary that can query to the encryption oracle.

2.3 Syntax and Correctness

A stateful order-preserving encryption scheme sOPE consists of the polynomial-time algorithms defined as follows:

- Kg , the key generation algorithm, is a probabilistic polynomial-time algorithm that, given a security parameter λ and a domain \mathcal{D} , returns a secret key okey that includes some states; $\text{okey} \leftarrow \text{sOPE.Kg}(1^\lambda, \mathcal{D})$.
- Enc , the encryption algorithm, is a deterministic polynomial-time algorithm that, given the secret key and a plaintext $m \in \mathcal{D}$, returns a ciphertext $c \in \mathcal{R}$ where \mathcal{R} is the range of sOPE ; $c \leftarrow \text{sOPE.Enc}(\text{okey}, m)$. Here, some states in okey are updated (inserted usually) during this encryption process.
- Dec , the decryption algorithm, is a deterministic polynomial-time algorithm that, given the secret key and a ciphertext $c \in \mathcal{R}$, returns a plaintext $m \in \mathcal{D}$; $m \leftarrow \text{sOPE.Dec}(\text{okey}, c)$.

For $\mathcal{D}, \mathcal{R} \subseteq \mathbb{N}$, an OPE scheme sOPE has to satisfy two requirements. One is the monotone increasing property: for all $(m_1, m_2) \in \mathcal{D} \times \mathcal{D}$ and $\text{okey} \leftarrow \text{sOPE.Kg}(1^\lambda, \mathcal{D})$,

$$m_1 < m_2 \iff c_1 < c_2 \text{ where } c_i \leftarrow \text{sOPE.Enc}(\text{okey}, m_i).$$

The other is the standard consistency constraint: for all plaintexts $m \in \mathcal{D}$ and $\text{okey} \leftarrow \text{sOPE.Kg}(1^\lambda, \mathcal{D})$,

$$m \leftarrow \text{sOPE.Dec}(\text{okey}, c) \text{ where } c \leftarrow \text{sOPE.Enc}(\text{okey}, m).$$

3 New Security Notion

Previous works showed that any IND-OCPA secure stateless or even stateful OPE scheme must have ciphertext size that is at least exponential to the size of the plaintext [5, 16]. As noted above, the first provably secure and feasible OPE scheme was designed to satisfy POPF-CCA. To clarify the OPE security guaranteed by such pseudorandom OPE schemes, the cryptographic properties of a random order-preserving function (ROPF) have been considered as an ideal object of POPF in [6]. One of things considered in such analysis is one-wayness under the KPA model. In more detail, known plaintext/ciphertext pairs split the plaintext and ciphertext spaces into subspaces, and so the analysis under each one-wayness definition reduces to that of an ROPF on the subspace.

In the CPA model, however, an adversary can even specify the target subspace and gradually constrain the range of the subspace. Then ciphertext indistinguishability, which is a stronger security notion than one-wayness, should guarantee that the advantage of any OCPA adversary to distinguish a pair of ciphertexts in the target subspace is negligible.

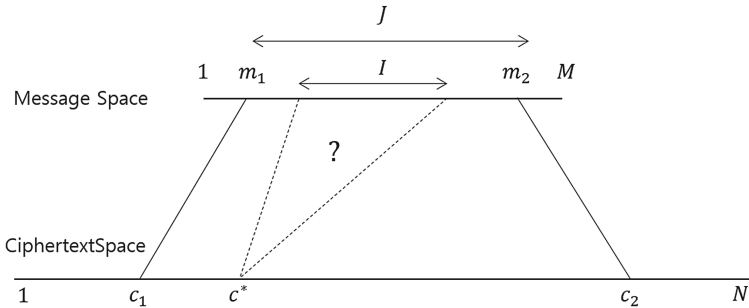


Fig. 1. Chosen plaintext attack on OPE

Figure 1 shows that a challenge ciphertext c^* is given to a OCPA adversary who obtained (m_1, c_1) and (m_2, c_2) from the encryption oracle. Here, our motivation started from the following question:

Is it possible to construct an OPE scheme ideally secure in at least l not J ?

An affirmative answer for this question has been shown by Teranish et al. [18]. They considered a find-then-guess type *partial* indistinguishability definition under the *non-adaptive* CPA model (called (k, θ) -FTG-O-nCPA), to ensure the

secrecy of the fraction of the least significant bits of a plaintext². Such secrecy is guaranteed by showing that a ciphertext of any element m_0^* in an interval I is indistinguishable from that of a uniformly random element m_1^* in I which is determined by k and θ . In more details, the interval I is set as $[\alpha, \beta]$ where α and β satisfy $|\alpha - m_1| \geq k\theta$ and $|m_2 - \beta| \geq k\theta$. Then it implies that $2k\theta + \theta \leq |J|$, $J = [m_1, m_2]$.

We extend this approach to the CPA model and present a new security notion, called δ -IND-OCPA. This notion is defined by the following game $\text{Game}_{\text{sOPE}}^{\delta\text{-ideal}}(\mathcal{A}, b)$ between a challenger \mathcal{C} and an adversary \mathcal{A} :

(Setup) \mathcal{C} runs $\text{okey} \leftarrow \text{sOPE.Kg}(\lambda)$ and chooses a random bit b .

(Query) At round $i \in [1, n]$, \mathcal{A} queries i -th message pair (m_i^0, m_i^1) to \mathcal{C} as follows:

1. \mathcal{A} chooses a message m_i^0 and sends it to \mathcal{C} .
2. If $m_i^0 = m_j^0$ for some $j \in [1, i-1]$, \mathcal{C} returns $l_i = \{m_j^1\}$ to \mathcal{A} .
3. Otherwise, \mathcal{C} finds the largest (resp. smallest) m_x^0 (resp. m_y^0) in $\text{QM} = \{m_0^0, m_1^0, \dots, m_{i-1}^0, m_{n+1}^0\}$ where $m_x^0 < m_i^0$ (resp. $m_y^0 > m_i^0$) and $m_0^0 = m_0^1 = 0$, $m_{n+1}^0 = m_{n+1}^1 = M+1$. Then, \mathcal{C} returns an interval l_i to \mathcal{A} satisfying

$$l_i \subseteq J_i = [m_x^1 + 1, m_y^1 - 1] \quad \text{and} \quad \log |l_i| \geq \delta \log |J_i|.$$

4. \mathcal{A} chooses $m_i^1 \in l_i$ arbitrarily and sends it to \mathcal{C} .
5. \mathcal{C} returns $c_i^b \leftarrow \text{sOPE.Enc}(\text{okey}, m_i^b)$ to \mathcal{A} .

Here, the left and right messages should have the same order relation, i.e., for all $1 \leq i, j \leq n$, $m_i^0 < m_j^0$ if and only if $m_i^1 < m_j^1$.

(Guess) \mathcal{A} outputs b' , its guess for b .

Definition 1. A stateful order-preserving encryption scheme sOPE is δ -IND-OCPA secure if for any probabilistic polynomial-time adversary \mathcal{A} ,

$$\left| \Pr[\text{Game}_{\text{sOPE}}^{\delta\text{-ideal}}(\mathcal{A}, 1) = 1] - \Pr[\text{Game}_{\text{sOPE}}^{\delta\text{-ideal}}(\mathcal{A}, 0) = 1] \right| \leq \text{negl}(\lambda).$$

Our δ -IND-OCPA notion means that plaintext information on I can be hidden to any polynomial-time adversaries. From $\log |I| \geq \delta \log |J|$, we know that δ means the ratio of the bit information that the underlying OPE scheme can protect to the maximum bit information guaranteed by the ideal security notion. As described in Fig. 1, $\delta = 1$ means that an adversary cannot get any information about a plaintext $m^* \in I (= J)$ from a ciphertext c^* , and it thus collapses to the case of IND-OCPA. Additionally, δ -IND-OCPA implies (k, θ) -FTG-O-nCPA when δ is taken as $\log \theta / \log(2k\theta + \theta)$ from $\theta = |I|$ and $2k\theta + \theta \leq |J|$.

As mentioned in [18], the proposed stateless OPE scheme with (k, θ) -FTG-O-nCPA security does not preserve ciphertext indistinguishability under the adaptive CPA model. Therefore, we now design a new OPE scheme satisfying δ -IND-OCPA notion. To counter the enhanced capabilities of the adversary, our main design strategy is to adopt statefulness property.

² Teranish et al. [18] proved that (k, θ) -FTG-O-nCPA implies θ -lsb-KPA that ensures the secrecy of the least significant $\log \theta$ bits of a plaintext under the known plaintext attack. Here, θ is determined by plaintext distribution and has a maximum value in the uniform distribution.

4 Construction

In this section, we show that the security notion δ -IND-OCPA is achievable by maintaining states of OPE. Similar to the constructions of [13, 16], the encryption algorithm of our OPE scheme can be separated into the conventional symmetric key encryption part and the order-preserving encoding part. It means that a ciphertext c consists of (c_1, c_2) where c_1 is an order-preserving encoding value and c_2 is an encrypted value using the symmetric key cipher. This section focuses on how to generate an encoded value c_1 that preserves order from a given plaintext. For convenience of explanation, we occasionally regard this order-preserving encoding part c_1 as the whole ciphertext c although it abuses notation.

4.1 Building Blocks

The basic concept of our construction is that each partitioned plaintext and ciphertext space divided by a new pair of plaintext and ciphertext maintains the ratio of the original plaintext space size to ciphertext space size. The following example explains why our idea is crucial to achieve δ -IND-OCPA security.

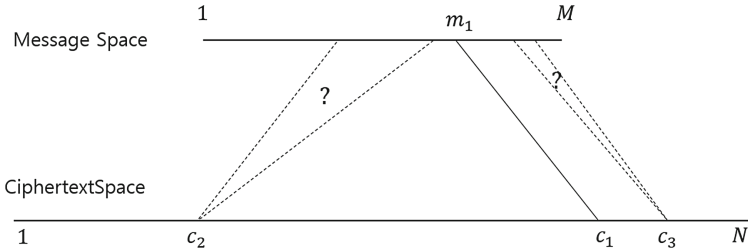


Fig. 2. The basic design idea of our scheme

Assume that a plaintext/ciphertext pair (m_1, c_1) divides the entire space into two subspaces, as depicted at Fig. 2. Here, the ratio between the partitioned plaintext and ciphertext space sizes divided by (m_1, c_1) is different. More precisely, the following relation holds.

$$\frac{\log(c_1 - 1)}{\log(m_1 - 1)} > \frac{\log N}{\log M} > \frac{\log(N - c_1)}{\log(M - m_1)}. \quad (1)$$

The left inequality in relation (1) shows that the ratio of the size of possible ciphertext space to the size of plaintext space $[1, m_1]$ is increased. Thus, it becomes more difficult for an adversary to infer plaintext information from a ciphertext c_2 than c_3 . On the other hand, the right inequality in (1) shows that the size of the ciphertext candidates to be reduced. If the latter case occurs in the encryption process, either the OCPA-advantage of an adversary may be increased by adaptive queries on such partitioned plaintext/ciphertext space

or the size of the interval I returned by the challenger \mathcal{C} in the δ -IND-OCPA game may be reduced. To overcome this, it is required to assign a ciphertext c for a given plaintext m to satisfy $\frac{\log(c-1)}{\log(m-1)} > \frac{\log N}{\log M}$ and $\frac{\log(N-c)}{\log(M-m)} > \frac{\log N}{\log M}$. It means that the ratio of between the partitioned plaintext and ciphertext space sizes induced from any plaintext and ciphertext pair (m, c) should be maintained as large as $\frac{\log N}{\log M}$. Below, we will show that an OPE scheme designed by this intuition satisfies the δ -IND-OCPA security. Figure 3 shows that any element in $[(m-1)^2+1, m(2M-m)]$ for a certain message m satisfies this condition. From $\frac{\log(m-1)^2}{\log(m-1)} = \frac{\log(M^2-m(2M-m))}{\log(M-m)} = \frac{\log M^2}{\log M} = 2$, we can assign any c in $[(m-1)^2+1, m(2M-m)]$ as a ciphertext of m . Algorithm 1, named Find, describes our intuition in more detail.

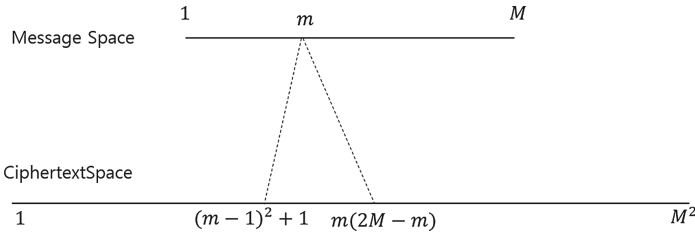


Fig. 3. Possible ciphertext space for a message m

Algorithm 1. Find(M, N, x) $\rightarrow y$

Input: M : size of plaintext space, $N(= M^d)$: size of ciphertext space, x : plaintext

Output: y : ciphertext

```

1: flag  $\leftarrow 0$ ;  $x_0 \leftarrow 1$ ;  $i \leftarrow 0$ 
2: if  $x > \lceil M/2 \rceil$  then
3:    $x \leftarrow M - x + 1$ ; flag  $\leftarrow 1$ 
4: end if
5: repeat
6:    $i \leftarrow i + 1$ 
7:    $y_i \leftarrow \min\{\lceil N/2 \rceil, N - (M - x_{i-1})^d\}$ 
8:    $x_i \leftarrow \min\{\lceil y_i^{1/d} \rceil, \lceil M/2 \rceil\}$ 
9:   if  $0 < \lceil M/2 \rceil - x_i < x_1$  then
10:     $x_i \leftarrow \lceil M/2 \rceil - x_1$ 
11:   end if
12: until  $x \leq x_i$ 
13: if flag = 0 or  $y_i = \lceil N/2 \rceil$  then
14:   output  $y_i$ 
15: else
16:   output  $N - y_i + 1$ 
17: end if

```

Figure 4 describes an embodiment of Algorithm 1 for a particular input value $(M, N, x) = (200, 40000, 170)$. The input value $x = 170$ is symmetrically shifted to $x' = 31$ based on the median of the plaintext space, and then $y_1 = 399$ and $x_1 = 20$ are sequentially computed in the first loop. We can easily check that $|\lceil [1, c-1] \rceil| > |\lceil [1, m-1] \rceil|^2$ and $|\lceil [c+1, 20000] \rceil| > |\lceil [m+1, 200] \rceil|^2$ if $\forall m \in [1, 20]$ and

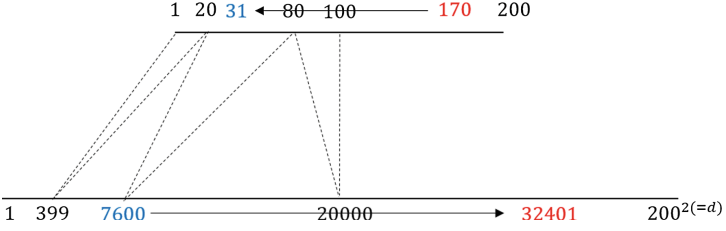


Fig. 4. An example of the algorithm

$c = 399$. Second loop computes $y_2 = 7600$ and $x_2 = 80$, where the correction of line 10 is occurred. Since $x_2 = 80 > 31 = x'$, 31 is associated with $y' = 7600$ and so Find outputs $y = N - y_2 + 1 = 32401$ which is symmetrically shifted based on the median of the ciphertext space. At this point, 399 is the only ciphertext candidate for all messages in $[1, 20]$, which guarantees indistinguishability on $[1, 20]$. Since $\log 20 / \log 200 > 0.5$ and it is the smallest interval, we can say that more than half of $\log M$ bits can be hidden in this example.

Lemma 1. *The running time of the Find algorithm is $O(\log M)$.*

Proof. First, we show that $2x_i < x_{i+1}$ when $d \geq 2$ and $x_i < 2M/5$. In Algorithm 1, x_{i+1} is defined as $(M^d - (M - x_i)^d)^{1/d}$ and it is a monotone increasing function for the variable d . We then know

$$x_{i+1} = (M^d - (M - x_i)^d)^{1/d} \geq (M^2 - (M - x_i)^2)^{1/2} = (2Mx_i - x_i^2)^{1/2}.$$

By assumption, we get $(2Mx_i - x_i^2)^{1/2} > 2x_i$, and thus $x_{i+1} > 2x_i$. And if $x_i > 2M/5$, then $x_{i+1} > M/2$. So the running time of the Find algorithm is $O(\log M)$. On the other hand, for $0 < x_i < M/2$, $x_{i+1} - x_i = (2Mx_i - x_i^d)^{1/d} - x_i$ takes its minimum at $x_i = 1$ (namely, $i = 0$), and lines 8–10 in Algorithm 1 guarantee this.

4.2 Proposed Scheme

In this subsection, we describe a new δ -IND-OCPA secure OPE scheme sOPE based on the Find algorithm. For a given plaintext space $[M]$, a client chooses a parameter $0 < \delta < 1$, and defines the ciphertext space as $[N]$ where $N = \lceil M^d \rceil$ and $d = 1/(1 - \delta)$. For better security, a larger ciphertext space is required, but there is a trade-off between security and efficiency. In Sect. 5.2 we provide a guideline for choosing a proper d based on our correlation analysis.

As noted above, a ciphertext c of sOPE consists of (c_1, c_2) where c_1 is an order-preserving encoding part and c_2 is a symmetric key encryption part. We assume that a secure symmetric key encryption scheme $\text{SE} = (\text{SE.Kg}, \text{SE.Enc}, \text{SE.Dec})$ is used for generating c_2 .

Algorithm 2. $\text{sOPE.Kg}(\lambda, [M]) \rightarrow \text{okey}$

- 1: $\text{KeyTable} \leftarrow \{(0, 0), (M + 1, N + 1)\}$
 - 2: $\text{skey} \leftarrow \text{SE.Kg}(\lambda)$
 - 3: Output $\text{okey} \leftarrow (\text{KeyTable}, \text{skey})$
-

Key Generation: Algorithm 2 shows the key generation algorithm of sOPE. Here, a key table KeyTable is used to keep some states.

Encryption: The encryption process based on the Find algorithm is quite straightforward. If an input m has previously been encrypted, namely $(m, y) \in \text{KeyTable}$ for some y , y can be used as the ciphertext c_1 of m again. Otherwise, c_1 is generated as described in Algorithm 3 and Fig. 5.

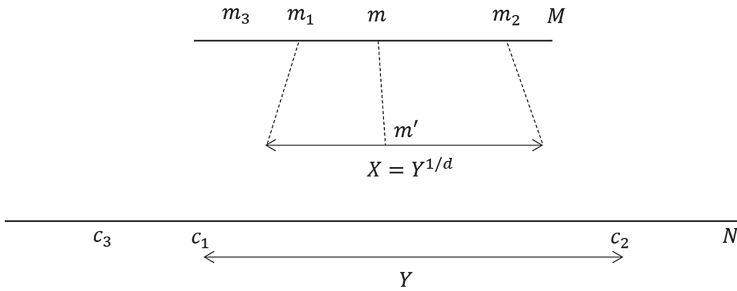


Fig. 5. How to set m'

Algorithm 3. $\text{sOPE.Enc}(\text{okey}, m) \rightarrow c$

- 1: $c_2 \leftarrow \text{SE.Enc}(\text{skey}, m)$
 - 2: **if** There exists (x, y) in KeyTable where $x = m$ **then**
 - 3: Output $c \leftarrow (y, c_2)$
 - 4: **else**
 - 5: Search $(x_1, y_1), (x_2, y_2) \leftarrow \text{KeyTable}$ where $x_1 < m < x_2$ and $|x_2 - x_1|$ is the smallest.
 - 6: $X \leftarrow \lfloor (y_2 - y_1 - 1)^{1/d} \rfloor$; $Y \leftarrow y_2 - y_1 - 1$
 - 7: $m' \leftarrow \lfloor (y_2 - y_1 - 1)^{1/d} (m - x_1) / (x_2 - x_1 - 1) \rfloor$
 - 8: $\text{KeyTable} \leftarrow \text{KeyTable} \cup \{(m, y_1 + y)\}$ where $y \leftarrow \text{Find}(X, Y, m')$
 - 9: Output $c \leftarrow (y_1 + y, c_2)$
 - 10: **end if**
-

From lines 5–6 of Algorithm 3, we get $Y = y_2 - y_1 - 1$ and $X = \lfloor Y^{1/d} \rfloor$ if we have (x_1, y_1) and (x_2, y_2) in KeyTable . Though it seems natural to set X to $x_2 - x_1 - 1$, note that this is not secure because some information of x_1 and x_2 can be deduced from X . Since the size of plaintext candidates X is adjusted

from $|x_2 - x_1 - 1|$ to $X = \lfloor Y^{1/d} \rfloor$, we have to adjust m to m' accordingly. Finally, c_1 is computed as $y_1 + y$ where $y \leftarrow \text{Find}(X, Y, m')$.

A client needs to maintain **KeyTable** of size at most M . By encrypting **KeyTable** and storing it on the server, we can make the secret key size $O(1)$. However, $O(\log M)$ communication costs with the server are required, as in [16], to perform line 5 of **sOPE.Enc**.

Decryption: Note that c_1 is not required for the decryption process. It is completed by simply running **SE.Dec** for c_2 .

4.3 Security Analysis

Theorem 1. *For the plaintext space $[M]$ and ciphertext space $[M^d]$, the proposed scheme **sOPE** is $(1 - \frac{1}{d})$ -IND-OCPA secure if the underlying symmetric key encryption scheme **SE** is CPA-secure.*

Proof. An adversary \mathcal{A} and a challenger \mathcal{C} proceed to the δ -IND-OCPA security game in the following process.

- (**Setup**) \mathcal{C} runs the key generation algorithm **sOPE.Kg**($\lambda, [M]$) twice, and obtains two keys $\text{okey1} = (\text{KeyTable1}, \text{skey1})$ and $\text{okey2} = (\text{KeyTable2}, \text{skey2})$.
- (**Query**) For a given message m_i^0 issued by \mathcal{A} at round $i \in [1, n = \text{poly}(\lambda)]$, \mathcal{C} determines l_i and c_i as follows:
 - If $m_i^0 = m_j^0$ for some $j \in [1, i - 1]$, set $l_i = \{m_j^1\}$ and $c_i = c_j$
 - Otherwise, \mathcal{C} first computes $c_i \leftarrow \text{sOPE.Enc}(\text{okey1}, m_i^0)$. Let (x_1, y_1) and (x_2, y_2) be two tuples obtained in the process of computing c_i (line 5 of Algorithm 3), and let $X = \lfloor (y_2 - y_1 - 1)^{1/d} \rfloor$, $Y = y_2 - y_1 - 1$, $x = \lfloor (y_2 - y_1 - 1)^{1/d} (m_i^0 - x_1) / (x_2 - x_1 - 1) \rfloor$. Note that given $y = \text{Find}(X, Y, x)$ \mathcal{C} can easily find an interval $[x_{i-1}, x_i]$ such that $y = \text{Find}(X, Y, x')$ for any $x' \in [x_{i-1}, x_i]$, and input parameters X and Y of **Find** algorithm depend on only *previous* ciphertexts. Thus, if all of ciphertexts in the states **KeyTable1** and **KeyTable2** are the same, \mathcal{C} can easily find an interval l_i satisfying $\text{sOPE.Enc}(\text{okey1}, m_i^0) = \text{sOPE.Enc}(\text{okey2}, m_i^1)$ for any $m_i^1 \in l_i$. \mathcal{C} returns such l_i to \mathcal{A} and receives $m_i^1 \in l_i$ from \mathcal{A} . Then, \mathcal{C} finally returns $c_i = \text{sOPE.Enc}(\text{okey1}, m_i^0) = \text{sOPE.Enc}(\text{okey2}, m_i^1)$ to \mathcal{A} .

To complete the proof, it is sufficient to show that the encryption of m_i^1 randomly selected in l_i are indistinguishable from the encryption of m_i^0 . Let E_1 and E_2 be the events where \mathcal{A} gets some information about plaintext through c_1 and c_2 of the ciphertext $c = (c_1, c_2)$, respectively. The advantage of \mathcal{A} for the security game is defined as follows. ($\delta = (d - 1)/d$)

$$\begin{aligned} \Pr[\text{Game}_{\text{sOPE}}^{\delta\text{-ideal}}(\mathcal{A}, 1) = 1] - \Pr[\text{Game}_{\text{sOPE}}^{\delta\text{-ideal}}(\mathcal{A}, 0) = 1] &= \Pr[E_1 \vee E_2] \\ &\leq \Pr[E_1] + \Pr[E_2]. \end{aligned}$$

By construction of l_i , we can easily check that all plaintext m_i^1 belonging to l_i and m_i^0 have the same ciphertext c_1 . This means $\Pr[E_1] = 0$. On the other

hand, $\Pr[E_2]$ is determined by the CPA security of SE, and by assumption, it is negligible. Now, we describe how to set δ from the expansion factor d . It is $(M^d - (M - 1)^d)^{1/d}$ when $||$ is the smallest as in the analysis of Sect. 4.

$$\frac{\log ||}{\log |J|} \geq \frac{\log(M^d - (M - 1)^d)^{1/d}}{\log M} \geq \frac{\log(M^{d-1})}{d \log M} = \frac{(d - 1)}{d}.$$

From this equation, we can obtain $\log || \geq \delta \log |J|$ where $\delta = (d - 1)/d$. By combining all these results, our proposed scheme can provide $(d - 1)/d$ -IND-OCPA security.

5 Evaluation

The evaluation of our scheme targets the correlation between plaintext and ciphertext and the performance comparison of encryption phase for the different OPE schemes.

5.1 General Setting

The client and the DBMS server were implemented and performed on a single desktop computer with a Intel Core i7 4790 3.6 GHz CPU, 16 GB of DDR3 RAM, and Samsung SSD 850 PRO 512 GB running Linux Mint 18(64-bit). The main programming language is C++14 and we also use Python to take advantage of the scripts we needed for testing. As a DBMS located at the server, we use MariaDB [3] and use MySQL Connector/C++ to embed DBMS directly in our application. Independently, to manage a state in the client, we use SQLite [4]. As a symmetric key cipher, we use AES-128 operating in ECB mode which is supported by the CPU instruction.

To implement our sOPE scheme, we use GMP [2] as a big number backend of the Boost multiprecision library [1]. And we use the conversion functions provided by GMP for conversion between binary and big integer. We also implemented BCLO [5] and KS [13] schemes for comparison. For BCLO scheme, we used the implementation from CryptDB [17] but replaced AES operation blocks with the equivalent AES-NI instructions. For KS scheme, we implemented using the description in [13]. But we modify update procedure as detailed in Appendix C. Such modifications on the KS scheme allow us to evaluate the efficiency of our scheme through more fair comparison, and we think it is an independently interesting result.

For our experiments, we use four types of datasets for plaintext inputs. One of them follows the uniform distribution and other three follow the normal distribution. To vary the distribution in the normal distribution, three ρ values 0.5, 1, and 2 are used. We constitute a dataset using the random number library of C++11.

5.2 Correlation Analysis

Each OPE scheme can be described as a monotonically increasing function, so a simple statistical cryptanalysis technique is to model the encryption as a linear function. To estimate the effectiveness of such statistical attack, Kerschbaum and Schröepfer [13], and Kerschbaum [12] compute the Pearson correlation coefficients for their OPE schemes. The correlation coefficient r measures a linear relation between two sets of random variables. Since r is expressed as a value between -1 and 1 , $r = 1$ (or $r = -1$) indicates linear correlation, and $r = 0$ indicates no correlation, a smaller r which is closer to 0 is better for the security of an OPE scheme.

In our experiment to measure correlation coefficients, we set the number l of bits in the plaintext domain by $l = 32$, i.e. $M = 2^{32}$, that is usually expected to be used frequently, and the number n of plaintexts by $n = 2^{20}$. Instead we vary the expansion factor d by choosing $d \in \{2, 3, 4, 5, 6\}$, i.e. $N = 2^{32d}$. We run 100 experiments for any combination of parameters. The correlation coefficients are computed by using `numpy.corrcoef` in Python.

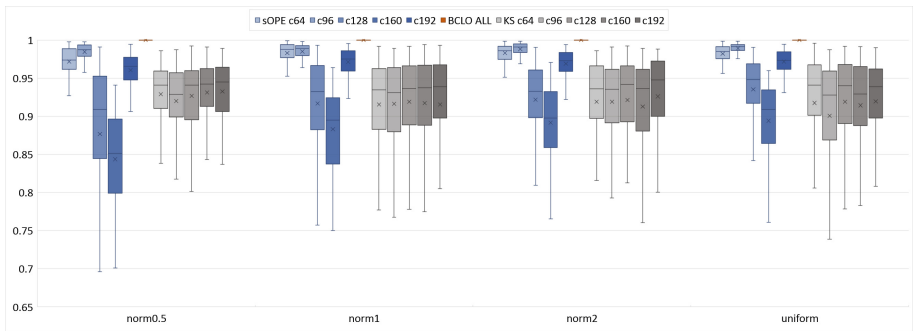


Fig. 6. Correlation coefficients for 32-bit normal and uniform datasets

Figure 6 depicts a distribution chart of the correlation coefficients for uniform and normal datasets. Each box plot shows the correlation coefficients of an OPE scheme for each value of d and each distribution. In a box plot, computed correlation coefficients are divided into quartiles, and a box is drawn between the first and third quartiles, with an additional line drawn along the second quartile to mark the median. A symbol \times inside a box marks the average. And the minimums and maximums outside the first and third quartiles are depicted with lines.

Under uniform distribution, the correlation coefficients for both BCLO and KS schemes already measured in [13]. Although parameter settings are slightly different, our experiment shows similar results. Even under normal distribution, no noticeable change of the patterns of correlation coefficients of both schemes was found. For our sOPE scheme, it can be expected to show a good correlation

coefficient because of d -th powering and d -th root operations. And Fig. 6 shows that such our guess is highly plausible. However, unlike the other two schemes, our sOPE scheme shows that the fluctuation of correlation coefficients for each distribution is relatively wide depending on the size of d .

In our experiment, the proposed scheme behaves better under normal distribution than uniform distribution. In normally distributed datasets, it behaves better when ρ value decreases. This means that even if plaintexts are clustered in a specific area, our sOPE scheme properly distributes corresponding ciphertexts. In particular, such characteristic can be an advantage when considering that the personal information usually follows the normal distribution.

5.3 Performance

Most of the OPE applications assume that the DBMS is located on the server and the client encrypts the data and stores it in the server-side DB. For some OPE schemes including Popa et al. [16] the performance of the network connecting client and server has a significant impact on the encryption time. But our evaluation targets are limited to the schemes that do not require communication with the server when encrypting. Instead, they use an internal state which is managed by the client-side DBMS or have no internal state at all. The KS scheme and our sOPE scheme are the former case, and the BCLO scheme is the latter case. In this case, the encryption procedure is performed only on the client side using the internal state (if exists), and only the final result, i.e. ciphertexts, is shared by the client and the server. So we measure the performance of the encryption procedure, including the associated DBMS query processing. To compute several variations, we vary two parameters: the number l of bits in the plaintext domain and the expansion factor d . For the size of the plaintext domain we choose $l \in \{16, 24, 32\}$, and we use the expansion factors $d = 2$ and 4. We set the number n of plaintexts by $n = 2^{20}$ and measure average time spent for one plaintext. We run five experiments for any combination of parameters to measure the average.

Figure 7 shows the performance results of three OPE schemes for uniform dataset. As a note, the performance difference according to the distribution of the data was not significant, so we omit results on other distribution datasets. For the BCLO scheme, it shows the performance of pure encryption procedure because the scheme does not use SQLite. Since we fix $n = 2^{20}$ that is sufficient margin to the size of ciphertext, performance difference of the KS scheme was not significant. On the other hand, we can confirm that the change of parameters, especially the size of the ciphertext domain, is reflected in the computation. When both sizes of plaintext and ciphertext are small, our scheme is fast, and for other cases, the KS scheme is fast. For one reason we think of this result is that it is possible to perform the d -th powering and d -th root operation without big-integer operations at values within 64 bits.

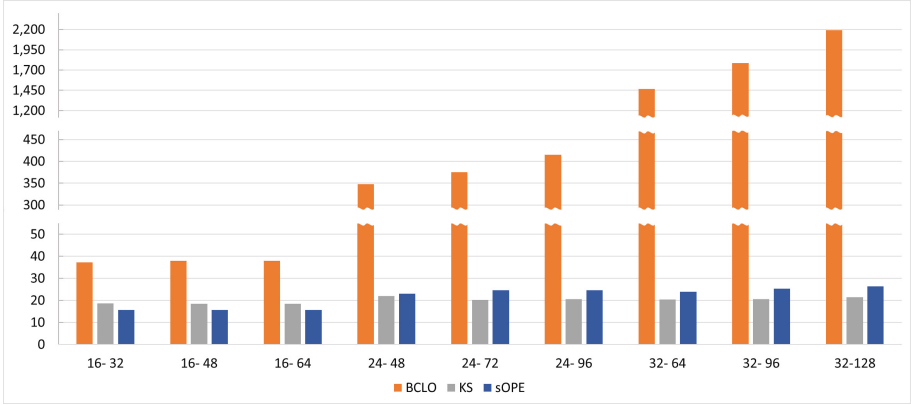


Fig. 7. Encryption speed comparison (unit: μs)

6 Concluding Remarks

In this paper, we propose a new security model δ -IND-OCPA, which is a relaxed version of IND-OCPA and show the relationship to other security models. We also show that δ -IND-OCPA can be achievable by constructing a feasible stateful OPE scheme.

Recently, several leakage-abuse attacks [9, 11, 15] against OPE schemes have been proposed. Those attacks show how leakage information such as access pattern, rank, density, and distribution can be used to statistically recover a significant amount of information about plaintext in OPE-protected DB and so demonstrates even IND-OCPA is insufficient. Though we have not analyzed in detail, the scheme we constructed is also expected to be vulnerable to such attacks in certain practical scenarios. For example, a generic attack in [14] assumes the adversary’s ability to utilize access pattern leakage of OPE schemes and a dense dataset. Therefore, if one plans to apply OPE to an application, we recommend to perform proper preliminary analysis to assure whether such attacks are problematic or not.

Acknowledgement. This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korean government (MSIT) (No. R0101-16-0301).

A Ideal Security

The IND-OCPA notion is a generalization of semantic security, and states that no efficient adversary can distinguish between the encryptions of any two sequences of messages, provided that the ordering of the messages in the two sequences is identical. We recall the formal definition here. Specifically, IND-OCPA is defined as the following game between a challenger \mathcal{C} and an adversary \mathcal{A} , where \mathcal{A} is a probabilistic polynomial-time algorithm:

(Setup) \mathcal{C} runs $\text{okey} \leftarrow \text{OPE.Kg}(1^\lambda, \mathcal{D})$ and chooses a random bit b .

(Query) At round $i \in [1, n]$, \mathcal{A} queries adaptively the i -th message pair (m_i^0, m_i^1) to \mathcal{C} , and then \mathcal{C} returns $c_i^b \leftarrow \text{OPE.Enc}(\text{okey}, m_i^b)$ to as its answer. Here, the left and right messages should have the same order relation, i.e., for all $1 \leq i, j \leq n$, $m_i^0 < m_j^0$ iff $m_i^1 < m_j^1$.

(Guess) \mathcal{A} outputs b' , its guess for b .

We say that an OPE scheme guarantees the IND-OCPA security if the probability of $b' = b$ is $1/2 + \text{negl}(\lambda)$.

B Further Correlation Coefficients Evaluation

To confirm consistency in the pattern of change, we further vary the number n of inputs from 2^{10} to 2^{20} by 2 times interval and compute the correlation coefficients for uniform and normal datasets. Figures 8 and 9 depict correlation coefficients of 32-bit normally distributed (with $\rho = 0.5$) and uniformly distributed datasets, respectively. In other two normal distribution cases, the same trend is shown in Fig. 8 based on Fig. 6, and the result is omitted. Conclusively, they show the same pattern of change depicted in Fig. 6 for each case.

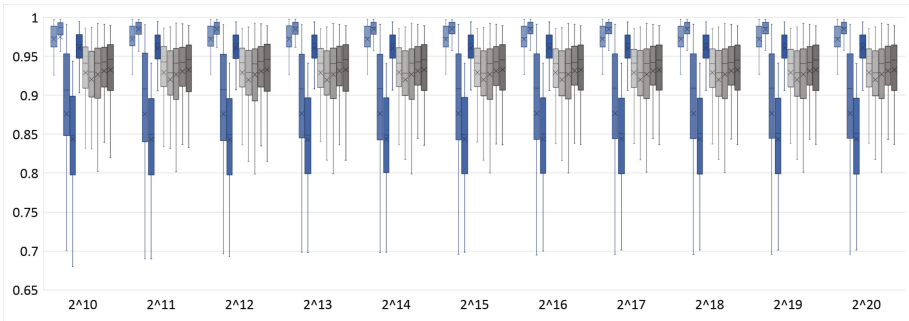


Fig. 8. Correlation coefficients for 32-bit normal datasets with $\rho = 0.5$

In general, it is predicted that as d increases, the correlation coefficient of our sOPE scheme will be lowered due to the influence of d -th powering or d -th root operation. But in this experiment we can observe that it decreases until the specific d , and then increases again. Figures 6, 8 and 9 show that such specific d is 5 when the size of the plaintext domain is 32. Considering that the larger d , the less efficient or our sOPE scheme, this particular d is considered *optimal*. We further examined for the other cases of $l \in \{16, 24, 32, 48\}$, and found that the optimal expansion factor d for each l is 3, 4, 5, and 6, respectively.

Despite that we need more experiments to reliably recommend a choice of d , we observe that our sOPE scheme with optimal choice of d performs better than the ideal-secure KS scheme under normal distribution. And the performance of both our sOPE scheme with optimal d and KS scheme are similar under uniform distribution.

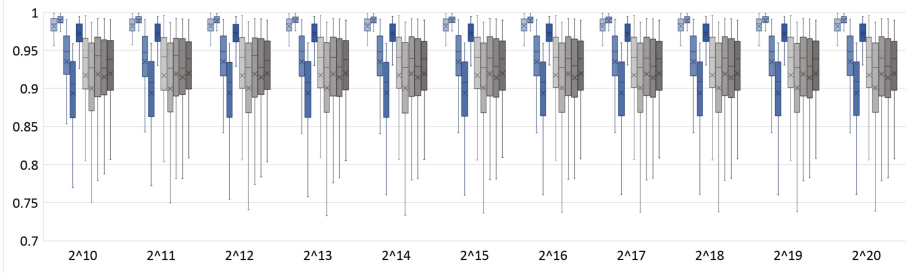


Fig. 9. Correlation coefficients for 32-bit uniform datasets

C Implementation of the KS Scheme

The update algorithm of the KS scheme updates the state managed by the client. It newly generates a ciphertext for all plaintexts up to the present in order by creating a balanced tree. The updated ciphertexts need to be sent to the server-side DB. Apart from the communication overhead that is generally pointed out, such procedure shows realistic problems in our experiment. To implement the update algorithm in MariaDB (including MySQL), we need to combine a specific UDF (User-Defined Function) and table manipulation procedures. Such approach to invoking cryptographic operations to the DBMS can be seen more specifically in [17]. But native MariaDB (or MySQL) functions to manipulate table have a data size constraint. Though there may be a way to suppress update operations through parameter setting as stated in [13], but in this experiment, we tried to solve the problem through implementation optimization.

In the KS scheme, the state that is represented as a binary tree plays the role of the key. And the encryption procedure for a new plaintext using this key can be interpreted as assigning a new node in the binary tree. In the update process, all plaintext encrypted up to the present must be sorted first. And a new balanced binary tree based only on the order of the sorted plaintexts is derived through the recursive procedure. This binary tree is used as a key for encrypting the next plaintext. Since the DBMS in the server can sort ciphertexts up to the present in order, it can perform an update operation itself by creating the same balanced binary tree with the client. Thus, we can update ciphertexts stored in the server-side DB without sending renewal ciphertexts. Instead, it is sufficient to send only basic sink information for the same balanced binary tree creation.

References

1. Boost C++ Libraries. <http://www.boost.org/>
2. The GNU Multiple Precision Arithmetic Library. <https://gmplib.org/>
3. The MariaDB Foundation. <https://mariadb.org/>
4. SQLite. <https://sqlite.org/>

5. Boldyreva, A., Chenette, N., Lee, Y., O'Neill, A.: Order-preserving symmetric encryption. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 224–241. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01001-9_13
6. Boldyreva, A., Chenette, N., O'Neill, A.: Order-preserving encryption revisited: improved security analysis and alternative solutions. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 578–595. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_33
7. Boneh, D., Lewi, K., Raykova, M., Sahai, A., Zhandry, M., Zimmerman, J.: Semantically secure order-revealing encryption: multi-input functional encryption without obfuscation. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 563–594. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_19
8. Chenette, N., Lewi, K., Weis, S.A., Wu, D.J.: Practical order-revealing encryption with limited leakage. In: Peyrin, T. (ed.) FSE 2016. LNCS, vol. 9783, pp. 474–493. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-52993-5_24
9. Grubbs, P., Sekniqi, K., Bindschaedler, V., Naveed, M., Ristenpart, T.: Leakage-abuse attacks against order-revealing encryption. In: 2017 IEEE Symposium on Security and Privacy (SP), pp. 655–672. IEEE Press, New York (2017). <https://doi.org/10.1109/SP.2017.44>
10. Katz, J., Yung, M.: Characterization of security notions for probabilistic private-key encryption. *J. Cryptol.* **19**(1), 67–96 (2006). <https://doi.org/10.1007/s00145-005-0310-8>
11. Kellaris, G., Kollios, G., Nissim, L., O'Neill, A.: Generic attacks on secure outsourced database. In: 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 1329–1340. ACM Press, New York (2016). <https://doi.org/10.1145/2976749.2978386>
12. Kerschbaum, F.: Frequency-hiding order-preserving encryption. In: 22nd ACM SIGSAC Conference on Computer and Communications Security, pp. 656–667. ACM Press, New York (2015). <https://doi.org/10.1145/2810103.2813629>
13. Kerschbaum, F., Schröpfer, A.: Optimal average-complexity ideal-security order-preserving encryption. In: 2014 ACM SIGSAC Conference on Computer and Communications Security, pp. 275–286. ACM Press, New York (2014). <https://doi.org/10.1145/2660267.2660277>
14. Lacharité, M.-S., Minaud, B., Paterson, K.G.: Improved reconstruction attacks on encrypted data using range query leakage. In: 2018 IEEE Symposium on Security and Privacy (SP), pp. 19–36. IEEE Press, New York (2018). <https://doi.org/10.1109/SP.2018.00002>
15. Naveed, M., Kamara, S., Wright, C.V.: Inference attacks on property-preserving encrypted databases. In: 22nd ACM SIGSAC Conference on Computer and Communications Security, pp. 644–655. ACM Press, New York (2015). <https://doi.org/10.1145/2810103.2813651>
16. Popa, R.A., Li, F.H., Zeldovich, N.: An ideal-security protocol for order-preserving encoding. In: 2013 IEEE Symposium on Security and Privacy (SP), pp. 463–477. IEEE Press, New York (2013). <https://doi.org/10.1109/SP.2013.38>
17. Popa, R.A., Redfield, C.M.S., Zeldovich, N., Balakrishnan, H.: CryptDB: protecting confidentiality with encrypted query processing. In: Twenty-Third ACM Symposium on Operating Systems Principles, pp. 85–100. ACM Press, New York (2011). <https://doi.org/10.1145/2043556.2043566>
18. Teranishi, I., Yung, M., Malkin, T.: Order-preserving encryption secure beyond one-wayness. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8874, pp. 42–61. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45608-8_3

Homomorphic Encryption



Cryptanalysis of Tran-Pang-Deng Verifiable Homomorphic Encryption

Shuaijianni Xu^{1,2}, Yan He^{1,2}, and Liang Feng Zhang¹(✉)

¹ School of Information Science and Technology, ShanghaiTech University,
Shanghai 201210, People's Republic of China
{xushjn, heyan, zhanglf}@shanghaitech.edu.cn

² Shanghai Institute of Microsystem and Information Technology,
Chinese Academy of Sciences, Shanghai 200050, People's Republic of China

Abstract. Tran, Pang and Deng (AsiaCCS'16) proposed two verifiable computation schemes on outsourced encrypted data in the cloud computing scenario. One of them enables the delegation of linear functions and the other is constructed for multivariate quadratic polynomials. In the quadratic function case, it was claimed that their scheme is the first to guarantee both confidentiality of input data and authenticity of computations *without using fully homomorphic encryption (FHE)*. In this paper we present a cryptanalysis which shows that their scheme cannot guarantee confidentiality of input data. We start with a technical lemma on pseudorandom functions that have a range of Abelian group and then provides a simple attack which allows the adversary to successfully break the scheme with probability close to 1.

Keywords: Cryptanalysis · Verifiable homomorphic encryption
Pseudorandom function

1 Introduction

The past years have witnessed an increasing amount of attention spent on the problem of securely outsourcing computation (a.k.a. delegating computation or verifiable computation), due to the popularity of cloud computing and the proliferation of mobile devices. The computationally weak clients such as smartphones and netbooks can outsource the storage of numerous data and expensive computations on the data to a powerful cloud server.

A main security issue arising in outsourcing is how to ensure the cloud server performs the delegated computation correctly. The cloud server may have strong financial incentives to run a quick but incorrect computation to free up computing resources. A number of models and schemes for securely outsourcing computation have been developed in the past decade. For example, the verifiable computation of Gennaro et al. [19] allows the client to outsource the computation of a function f on any input x and then verify the result of the server's work. The bottom line is that the client's work of preparing delegation and

doing verification should be substantially more efficient than computing $f(x)$ from scratch. Verifiable computation schemes enabling the delegation of an arbitrary function f (encoded as a Boolean circuit) [1, 4, 15, 25] are mostly short of efficiency due to their dependence on expensive cryptographic primitives (such as fully homomorphic encryption (FHE) and garbled circuit) and the necessity of representing f as a Boolean circuit. A program of constructing efficient verifiable computation schemes has been initiated by [6] and resulted in a number of constructions for restricted classes of functions [10, 17, 24]. It was shown that several related cryptographic primitives such as homomorphic MAC [3, 9, 11, 20] and homomorphic signature [2, 7, 8, 12–14, 16] imply verifiable computation, in both privately verifiable setting and publicly verifiable setting.

Another security issue is how to protect the data of the client (e.g. the input x) from an untrusted cloud server. After all, a cloud server not trusted to perform computation correctly can hardly be trusted with the knowledge of the data. How to keep the confidentiality of input data was not addressed in the above constructions except [4, 15, 19], which are based on FHE and short of efficiency. Fiore et al. [18] and Joo and Yun [21] enable the delegation of restricted functions such as multivariate quadratic polynomials with both the confidentiality of input data and the authenticity of computation achieved. Although more efficient, they were again based on FHE or adapted versions of FHE. Lai et al. [22] combined data encryption and homomorphic message authenticator (either homomorphic signature or homomorphic MAC) via Encrypt-and-MAC [5] and obtained verifiable homomorphic encryption schemes that resolve both security issues. Their schemes takes FHE as the data encryption and still lack efficiency. The constructions of [2, 8, 12, 13, 16] imply efficient verifiable homomorphic encryptions but only enable the delegation of linear functions. Tran et al. [26] proposed a verifiable homomorphic encryption scheme and claimed that theirs is the first scheme for delegating nonlinear functions (more precisely, multivariate quadratic polynomials) on encrypted data, guaranteeing both the confidentiality of input data and the authenticity of computation, without using FHE. It was claimed that their data privacy was based on the semantical security of the additively homomorphic Paillier encryption scheme [23] and the hardness of solving discrete logarithm problem in \mathbb{Z}_q^* , the multiplicative group of integers modulo a safe prime q .

In this paper, we present a cryptanalysis of their verifiable homomorphic encryption scheme and show that it cannot keep the client’s data semantically secure against a malicious cloud server. We observe that (see Sect. 2.4 for more details) their scheme transforms any data block $m \in \mathbb{Z}_{q-1}$ into a quadruple $(c_0, c_1, y_0, Y_1) = ((g')^m \cdot u^N \bmod N^2, m - k_\tau \bmod N, \frac{m - r_\tau}{s} \bmod q - 1, g^{r_\tau} \bmod q)$. While c_0 and c_1 are ciphertexts of m under the Paillier’s encryption and one-time pad encryption (with refreshed keys), respectively, the remaining components do not form a semantically secure ciphertext of m . We show that y_0 and Y_1 reveal essential information of m . Our work calls for the first verifiable computation scheme that achieves both confidentiality of input data and authenticity of computations for nonlinear functions, *without using FHE*.

2 Tran-Pang-Deng Verifiable Homomorphic Encryption

We first review the definitions, notations and relevant constructions from [26]. Let $x \leftarrow S$ be the operation of assigning to x an element selected uniformly at random from a set S . The notation $x \leftarrow A(\cdot)$ denotes the operation of running a procedure A with the given input and assigning the output to x . For any security parameter λ , let $\text{negl}(\lambda)$ denote a negligible function in λ , i.e., for every real number $c > 0$, there is an integer $\lambda_0 > 0$ such that $\text{negl}(\lambda) < \lambda^{-c}$ for all $\lambda > \lambda_0$; let $\text{poly}(\lambda)$ denote a polynomial function in λ . For any integer $n > 0$, let $[1, n]$ denote the set $\{1, 2, \dots, n\}$.

2.1 Pseudorandom Function

A pseudorandom function (PRF) is an efficiently computable keyed function family $F : \mathcal{K} \times \mathcal{X} \rightarrow \mathcal{Y}$ with the property that the input-output behavior of a random instance F_k ($k \in \mathcal{K} \subseteq \{0, 1\}^\lambda$) of the family is computationally indistinguishable from that of a random function with the same domain \mathcal{X} and range \mathcal{Y} , i.e., for all probabilistic polynomial-time distinguishers \mathcal{D} , there is a negligible function $\text{negl}(\cdot)$ such that

$$|\Pr[\mathcal{D}^{F_k(\cdot)} = 1] - \Pr[\mathcal{D}^{f(\cdot)} = 1]| \leq \text{negl}(\lambda)$$

where the first probability is taken over the uniform choice of $k \leftarrow \mathcal{K}$ and the random coins of \mathcal{D} , and the second probability is taken over the uniform choice of $f \leftarrow \{R|R : \mathcal{X} \rightarrow \mathcal{Y}\}$ and the random coins of \mathcal{D} .

2.2 Arithmetic Circuits

An arithmetic circuit over a field \mathbb{F} is a directed acyclic graph. Each node in the graph is called a *gate*. A gate with input degree 0 is an *input gate*, which is labeled by either a variable from a set of $\chi = (x_1, \dots, x_n) \in \mathbb{F}^n$ or a constant $\alpha \in \mathbb{F}$. A gate with in-degree and out-degree greater than 0 is an *internal gate*, which is either an addition gate with a ‘+’ label, or a multiplication gate with a ‘×’ label. A gate with out-degree 0 is an output gate. The scheme of [26] allows a variable to undergo only one multiplication with another variable, but unlimited multiplications with constants and additions with other variables. Such a circuit can always be transformed into one in which every internal gate has two inputs. Therefore, they only consider circuits with one output gate, and in which each internal gate has in-degree 2. The final result of the circuit is the output of the output gate.

2.3 Labeled Programs

A labeled program \mathcal{P} is defined by a tuple $(f, \tau_1, \dots, \tau_n)$ where $f : \mathbb{F}^n \rightarrow \mathbb{F}$ is a circuit as defined above, and each label $\tau_i \in \{0, 1\}^*$ (where $\{0, 1\}^*$ is the set of all finite bit strings) uniquely identifies the i -th input node of f . Labeled programs

may be composed as follows. Given labeled programs $\mathcal{P}_1, \dots, \mathcal{P}_t$ and a circuit $f : \mathbb{F}^t \rightarrow \mathbb{F}$, the composed programs $\mathcal{P}^* = f(\mathcal{P}_1, \dots, \mathcal{P}_t)$ evaluates a circuit f on the outputs of $\mathcal{P}_1, \dots, \mathcal{P}_t$. The labeled inputs of \mathcal{P}^* correspond to the distinct labeled inputs of $\mathcal{P}_1, \dots, \mathcal{P}_t$. Let $\mathcal{I}_\tau = (f_{id}, \tau)$ denote the identity program with input label $\tau \in \{0, 1\}^*$, where $f_{id} : \mathbb{F} \rightarrow \mathbb{F}$ is the canonical identity function. Notice that any program $\mathcal{P} = (f, \tau_1, \dots, \tau_n)$ can be written as a composition of identity programs $\mathcal{P} = f(\mathcal{I}_{\tau_1}, \dots, \mathcal{I}_{\tau_n})$.

2.4 Verifiable Homomorphic Encryption

Informally, a verifiable homomorphic encryption (VHE) is a symmetric-key homomorphic encryption which enables verifiable computation on outsourced encrypted data. In a VHE scheme, a user with secret key SK can encrypt n messages m_1, \dots, m_n into n independent ciphertexts c_1, \dots, c_n . Given ciphertexts c_1, \dots, c_n and an admissible function f , anyone can compute ciphertext $c = f(c_1, \dots, c_n)$. The user then decrypts c to get m with the secret key SK , and check whether $m = f(m_1, \dots, m_n)$.

The generic VHE scheme (called HEMAC in [26]) is a tuple $\mathcal{H} = (\text{KeyGen}, \text{Enc}, \text{Eval}, \text{Dec})$ of probabilistic polynomial time algorithms defined as follows:

- $\text{KeyGen}(1^\lambda)$ takes as input a security parameter λ , and outputs a secret key SK and a public parameter PP .
- $\text{Enc}(SK, \tau, m)$ takes as input a secret key SK , a label $\tau \in \{0, 1\}^*$ and a datum $m \in \mathcal{M}$, where \mathcal{M} is the data space. It outputs a ciphertext C .
- $\text{Eval}(PP, \mathcal{P}, \mathbf{C})$ takes as input the public parameter PP , a label program $\mathcal{P} = (f, \tau_1, \dots, \tau_n)$ and a vector of ciphertexts $\mathbf{C} = (C_1, \dots, C_n)$. It outputs a new ciphertext C .
- $\text{Dec}(SK, \mathcal{P}, C)$ takes as input the secret key SK , a labeled program $\mathcal{P} = (f, \tau_1, \dots, \tau_n)$ and a ciphertext C . It outputs a datum $m \in \mathcal{M}$ or an error symbol \perp .

The HEMAC scheme \mathcal{H} has to satisfy the requirements of semantic security and unforgeability.

Attack 1. The semantic security of HEMAC is formalized with the following security game between a challenger and an adversary \mathcal{A} .

Setup. The challenger runs $\text{KeyGen}(1^\lambda)$ to obtain a pair of secret key and public parameter (SK, PP) . It gives public parameter PP to adversary \mathcal{A} , and keeps secret key SK itself. The challenger also initializes a list $T = \emptyset$ for tracking the queries from \mathcal{A} .

Queries. Adversary \mathcal{A} adaptively issues encryption queries to the challenger, each of the form (τ, m) where $\tau \in \{0, 1\}^*$ and $m \in \mathcal{M}$. The challenger then performs the following:

- if τ does not exist in T , the challenger computes $C \leftarrow \text{Enc}(SK, \tau, m)$, updates the list $T = T \cup \{\tau\}$, and gives ciphertext C to \mathcal{A} .

– if τ is found in T , the challenger rejects the query.

Challenge. Adversary \mathcal{A} submits a label $\tau^* \in \{0, 1\}^*$ and two data items $m_0, m_1 \in \mathcal{M}$, such that τ^* is not already in list T . The challenger selects a random bit $b \in \{0, 1\}$, computes $C^* \leftarrow \text{Enc}(SK, \tau^*, m_b)$, and sends C^* to \mathcal{A} .

Output. Adversary \mathcal{A} outputs $b' \in \{0, 1\}$ representing its guess for b . \mathcal{A} wins the game if $b' = b$.

The advantage $ss\text{-adv}[\mathcal{A}, \mathcal{H}]$ of adversary \mathcal{A} with respect to the HEMAC scheme in this game is defined as

$$\left| \Pr[b' = b] - \frac{1}{2} \right|$$

where the probability is taken over the random bits used by the challenger and adversary \mathcal{A} .

Definition 1. The HEMAC scheme \mathcal{H} is semantically secure if, for all probabilistic polynomial time adversary \mathcal{A} , the advantage $ss\text{-adv}[\mathcal{A}, \mathcal{H}]$ is negligible.

The HEMAC scheme should be existentially unforgeable under adaptive chosen message and query verification attacks. However, unforgeability is not the subject of this work and will not be elaborated here.

Tran et al. proposed two schemes in [26], one is for linear functions (Sect. 4 of [26]), the other is for quadratic functions (Sect. 5 of [26]). In this paper, we mainly focus on the second scheme which is denoted as q -HEMAC. Let $f : \mathbb{F}^n \rightarrow \mathbb{F}$ be an arithmetic circuit with addition gates and multiplication gates such that any multiplication gate, for which none of its two inputs are constants, can only be followed by addition gates or multiplication gates with as least one constant as input. Without loss of generality, f can be identified with a quadratic multivariate polynomial:

$$f(x_1, \dots, x_n) = \sum_{i,j \in [1,n]} \alpha_{i,j} x_i x_j + \sum_{l \in [1,n]} \alpha_l x_l + \alpha \tag{1}$$

for some constants $\alpha_i, \alpha_j, \alpha_l, \alpha \in \mathbb{F}$ and x_i, x_j, x_l taking arbitrary values in \mathbb{F} .

As our attack is on the semantical security of q -HEMAC, we are only interested in the algorithms $\text{KeyGen}(\cdot)$ and $\text{Enc}(\cdot)$ of q -HEMAC:

– $\text{KeyGen}(1^\lambda)$. Let p_1, p_2 be prime numbers with roughly $\lambda/2$ bits, where λ is a security parameter. Run the Paillier key generation algorithm [23] to generate public parameters $N = p_1 p_2$ and g' . Choose a random seed $K \leftarrow \mathbb{Z}_N$ for the pseudo-random function $F'_K : \{0, 1\}^* \rightarrow \mathbb{Z}_N$. Next, let q be a large safe prime number with roughly λ bits such that $q < N$. Let \mathbb{Z}_q^* be a multiplicative cyclic group of order $q - 1$ on which the discrete logarithm problem is hard, and let g be a generator of \mathbb{Z}_q^* . Choose a random seed $R \leftarrow \mathbb{Z}_{q-1}$ for the pseudo-random function $F_R : \{0, 1\}^* \rightarrow \mathbb{Z}_{q-1}$, and a random number $s \leftarrow \mathbb{Z}_{q-1}^*$. Publish the public key $PK = (N, g', g, q)$, and retain the secret key $SK = (p_1, p_2, K, R, s)$. The data space is $\mathcal{M} = \mathbb{Z}_{q-1}$ and $\mathbb{F} = \mathbb{Z}_{q-1}$.

- $\text{Enc}(SK, \tau, m)$. Given secret key $SK = (p_1, p_2, K, R, s)$, proceed as follows to encrypt a datum $m \in \mathbb{Z}_{q-1}$ with label $\tau \in \{0, 1\}^*$. First, compute an encryption key $k_\tau = F'_K(\tau)$ and a pseudo-random value $r_\tau = F_R(\tau)$. Then, choose a random number $u \leftarrow \mathbb{Z}_q^*$, and output a level-1 ciphertext $C = (c_0, c_1, y_0, Y_1) \in \mathbb{Z}_{N^2}^* \times \mathbb{Z}_N \times \mathbb{Z}_{q-1} \times \mathbb{Z}_q^*$ where:

$$\begin{aligned} c_0 &= g'^m \cdot u^N \bmod N^2, \\ c_1 &= m - k_\tau \bmod N, \\ y_0 &= \frac{m - r_\tau}{s} \bmod (q - 1), \\ Y_1 &= g^{r_\tau} \bmod q. \end{aligned} \tag{2}$$

In the following sections we shall show that the output of $\text{Enc}(SK, \tau, m)$ will reveal much information about m and thus render the semantical security of q -HEMAC completely broken.

3 A Technical Lemma on Pseudorandom Functions

Our cryptanalysis starts with a technical lemma on the PRFs with a range \mathcal{Y} of Abelian groups. Without loss of generality, we suppose that \mathcal{Y} is additive. Our technical lemma informally says that, given any subset $\mathcal{T} \subseteq \mathcal{Y}$ with

$$\frac{|\mathcal{T}|}{|\mathcal{Y}|} \geq \frac{1}{\text{poly}(\lambda)}, \tag{3}$$

where λ is a security parameter (such as the one in Sect. 2.4) and $|\mathcal{Y}| = 2^{O(\lambda)}$, one can easily find two inputs $x, x' \in \mathcal{X}$ such that $F_k(x') - F_k(x) \in \mathcal{T}$.

Lemma 1. *Let $F : \mathcal{K} \times \mathcal{X} \rightarrow \mathcal{Y}$ be a PRF where the range \mathcal{Y} is an additive Abelian group. Let $\mathcal{T} \subseteq \mathcal{Y}$ and $|\mathcal{T}|/|\mathcal{Y}| \geq \frac{1}{\alpha(\lambda)}$ for a polynomial function $\alpha(\lambda)$. Suppose that there is a polynomial time algorithm deciding whether $y \in \mathcal{T}$ for all $y \in \mathcal{Y}$. Then there is a probabilistic polynomial time algorithm with oracle access to $F_k(\cdot)$ (where $k \leftarrow \mathcal{K}$) and outputs two inputs $x, x' \in \mathcal{X}$ such that $F_k(x') - F_k(x) \in \mathcal{T}$ with probability $\geq 1 - \text{negl}(\lambda)$.*

Proof. Let \mathcal{A} be a probabilistic algorithm with oracle access to $F_k(\cdot)$ for $k \leftarrow \mathcal{K}$ and

- for $t = \lambda \cdot \alpha(\lambda)$, choose a $(t + 1)$ -subset $\{x_0, x_1, \dots, x_t\}$ of the set \mathcal{X} ;
- for $i = 0$ to t : feed $F_k(\cdot)$ with x_i and get $F_k(x_i)$;
- for $i = 1$ to t : if $F_k(x_i) - F_k(x_0) \in \mathcal{T}$, then output 0;
- otherwise, output 1.

The algorithm \mathcal{A} clearly runs in polynomial time. For every $i \in [1, t]$, let E_i be the event that $F_k(x_i) - F_k(x) \notin \mathcal{T}$. Let $E = E_1 \wedge E_2 \wedge \dots \wedge E_t$, i.e., the event that \mathcal{A} fails and outputs \perp . It remains to show that $\Pr[E] \leq \text{negl}(\lambda)$, where the probability is taken over $k \leftarrow \mathcal{K}$ and the random choices of $x_0, x_1, \dots, x_t \leftarrow \mathcal{X}$.

In order to show that $\Pr[E] \leq \text{negl}(\lambda)$, we construct a distinguisher \mathcal{D} for the PRF F . Given access to an oracle \mathcal{O} , which either implements $F_k(\cdot)$ for $k \leftarrow \mathcal{K}$ or a truly random function $f : \mathcal{X} \rightarrow \mathcal{Y}$, \mathcal{D} proceeds as below:

- for $t = \lambda \cdot \alpha(\lambda)$, choose a $(t + 1)$ -subset $\{x_0, x_1, \dots, x_t\}$ of the set \mathcal{X} ;
- for $i = 0$ to t : feed $\mathcal{O}(\cdot)$ with x_i and get $\mathcal{O}(x_i)$;
- for $i = 1$ to t : if $\mathcal{O}(x_i) - \mathcal{O}(x_0) \in \mathcal{T}$, then output 0;
- otherwise, output 1.

Because F is a PRF, we must have that $|\Pr[\mathcal{D}^{\mathcal{O}(\cdot)}(\lambda) = 1 | \mathcal{O} = F_k] - \Pr[\mathcal{D}^{\mathcal{O}(\cdot)}(\lambda) = 1 | \mathcal{O} = f]| \leq \text{negl}(\lambda)$, where $k \leftarrow \mathcal{K}$ and $f \leftarrow \{R | R : \mathcal{X} \rightarrow \mathcal{Y}\}$ are uniformly chosen. That is,

$$\left| \Pr[\mathcal{D}^{F_k(\cdot)}(\lambda) = 1] - \Pr[\mathcal{D}^{f(\cdot)}(\lambda) = 1] \right| \leq \text{negl}(\lambda).$$

Note that $\Pr[\mathcal{D}^{F_k(\cdot)}(\lambda) = 1] = \Pr[E]$. Hence

$$\Pr[E] \leq \Pr[\mathcal{D}^{f(\cdot)}(\lambda) = 1] + \text{negl}(\lambda). \quad (4)$$

$\Pr[\mathcal{D}^{f(\cdot)}(\lambda) = 1]$ is taken over the uniform choices of $f \in \{R | R : \mathcal{X} \rightarrow \mathcal{Y}\}$ and $\{x_0, x_1, \dots, x_t\} \subseteq \mathcal{X}$. The event $\mathcal{D}^{f(\cdot)}(\lambda) = 1$ occurs if and only if $f(x_i) - f(x_0) \notin \mathcal{T}$ for all $i \in [t]$. As the inputs $x_0, x_1, x_2, \dots, x_t$ are all distinct from each other, $\{f(x_i) - f(x_0)\}_{i=1}^t$ are t uniformly distributed random variables over \mathcal{Y} and totally independent. For every $i \in [t]$,

$$\Pr[f(x_i) - f(x_0) \in \mathcal{T}] = \frac{|\mathcal{T}|}{|\mathcal{Y}|} \geq \frac{1}{\alpha(\lambda)}.$$

It follows that

$$\begin{aligned} \Pr[\mathcal{D}^{f(\cdot)}(\lambda) = 1] &= \Pr[f(x_i) - f(x_0) \notin \mathcal{T} \text{ for all } i \in [t]] \\ &= \prod_{i=1}^t \Pr[f(x_i) - f(x_0) \notin \mathcal{T}] \\ &\leq \prod_{i=1}^t \left(1 - \frac{1}{\alpha(\lambda)}\right) \\ &\leq e^{-\lambda}. \end{aligned}$$

Due to Eq. (4), we have that $\Pr[E] \leq e^{-\lambda} + \text{negl}(\lambda) = \text{negl}(\lambda)$. \square

Remark. In the proof of Lemma 1, we could choose $t = \alpha(\lambda) \cdot \omega(\log \lambda)$ such that $\Pr[\mathcal{D}^{f(\cdot)}(\lambda) = 1] \leq e^{-\omega(\log \lambda)}$ is negligible.

4 The Proposed Attack

In this section, we show that the Tran-Pang-Deng verifiable homomorphic encryption q -HEMAC (2) cannot provide the claimed “confidentiality of input

data” with a simple attack. Recall that in (2) each data block $m \in \mathcal{M}$ is encrypted as a quadruple (c_0, c_1, y_0, Y_1) , where c_0 and c_1 are ciphertexts of m under the Paillier encryption and one-time pad encryption (keys for each data block refreshed constantly), respectively. While (c_0, c_1) leaks no significant information about m , we show that (y_0, Y_1) do not provide a secure encryption of m .

Theorem 1. *The scheme q -HEMAC is not semantically secure according to Definition 1.*

Proof. We denote by \mathcal{H} the scheme q -HEMAC. Consider the security game

Attack 1. We construct a PPT adversary \mathcal{A} such that $ss\text{-adv}[\mathcal{A}, \mathcal{H}]$ is non-negligible. Our adversary \mathcal{A} works as below with its challenger:

- **Setup.** The challenger runs $\mathcal{H}.\text{KeyGen}(1^\lambda)$ to obtain a secret key $SK = (p_1, p_2, K, R, s)$ and the public parameter $PP = (N, g', g, q)$, where p_1, p_2 are $\lambda/2$ -bit primes, $N = p_1 p_2$ and g' are public parameters for the Paillier encryption, $q = 2p + 1$ is a random λ -bit safe prime, g is a generator of \mathbb{Z}_q^* , $s \leftarrow \mathbb{Z}_{q-1}^*$, $K \leftarrow \mathbb{Z}_N$ and $R \leftarrow \mathbb{Z}_{q-1}$ are secret keys of the PRFs $F'_K : \{0, 1\}^* \rightarrow \mathbb{Z}_N$ and $F_R : \{0, 1\}^* \rightarrow \mathbb{Z}_{q-1}$, respectively. The challenger then gives PP to the adversary \mathcal{A} , and keeps SK to itself. The challenger also initializes a list $T = \emptyset$ for tracking the queries from \mathcal{A} .
- **Queries.** Let $t = 2\lambda$. The adversary \mathcal{A} chooses a $(t + 1)$ -subset $\{\tau_0, \tau_1, \dots, \tau_t\}$ of $\{0, 1\}^*$. The adversary also chooses a data block $m \in \mathcal{M}$ arbitrarily. Then for $i = 0$ to t :
 - \mathcal{A} queries its challenger with (τ_i, m) .
 - Note that the label $\tau_i \in \{0, 1\}^*$ was never reused. The challenger updates the list $T = T \cup \{\tau_i\}$ and replies with

$$C_i = \left(c_{0,i}, c_{1,i}, y_{0,i} = \frac{m - r_{\tau_i}}{s} \bmod (q - 1), Y_{1,i} = g^{r_{\tau_i}} \bmod q \right),$$

where $r_{\tau_i} = F_R(\tau_i)$.

- **Challenge.** The adversary \mathcal{A} submits a label $\tau^* \in \{0, 1\}^*$ and two data blocks $m_0^*, m_1^* \in \mathcal{M}$, such that $m_0^* \not\equiv m_1^* \pmod{q - 1}$ and $\tau^* \notin T$. The challenger selects a random bit $b \in \{0, 1\}$ and replies with

$$C^* = \left(c_0^*, c_1^*, y_0^* = \frac{m_b^* - r_{\tau^*}}{s} \bmod (q - 1), Y_1^* = g^{r_{\tau^*}} \bmod q \right),$$

where $r_{\tau^*} = F_R(\tau^*)$.

- **Output.** The adversary \mathcal{A} determines its output $b' \in \{0, 1\}$ as below.
 - For every $i \in \{1, \dots, t\}$, compute

$$\frac{r_{\tau_i} - r_{\tau_0}}{s} = \frac{m - r_{\tau_0}}{s} - \frac{m - r_{\tau_i}}{s}. \quad (5)$$

- If $\frac{r_{\tau_i} - r_{\tau_0}}{s} \notin \mathbb{Z}_{q-1}^*$ for all $i \in \{1, \dots, t\}$, then choose $b' \leftarrow \{0, 1\}$ uniformly at random and output b' .

- Otherwise, there is at least one integer $i \in \{1, \dots, t\}$ such that $\frac{r_{\tau_i} - r_{\tau_0}}{s} \in \mathbb{Z}_{q-1}^*$. Compute

$$\begin{aligned} \frac{s}{r_{\tau_i} - r_{\tau_0}} &= \left(\frac{r_{\tau_i} - r_{\tau_0}}{s} \right)^{-1} \pmod{q-1}, \\ g^s &= (g^{r_{\tau_i}} \cdot g^{-r_{\tau_0}})^{\frac{s}{r_{\tau_i} - r_{\tau_0}}} \pmod{q}, \\ g^{m_b^*} &= (g^s)^{\frac{m_b^* - r_{\tau^*}}{s}} \cdot g^{r_{\tau^*}} \pmod{q}. \end{aligned} \tag{6}$$

Compare $g^{m_b^*}$ with $g^{m_0^*}$ and $g^{m_1^*}$. Define

$$b' = \begin{cases} 0, & \text{if } g^{m_b^*} = g^{m_0^*}; \\ 1, & \text{if } g^{m_b^*} = g^{m_1^*} \end{cases}$$

and output b' .

It is trivial to see that the adversary \mathcal{A} is probabilistic and the running time of \mathcal{A} is $\text{poly}(\lambda)$. It remains to show that \mathcal{A} outputs the correct value of b with non-negligible probability, i.e., $ss\text{-adv}[\mathcal{A}, \mathcal{H}] = \Pr[b' = b]$ is non-negligible in λ . Let I be the event that there is at least one integer $i \in \{1, \dots, t\}$ such that $\frac{r_{\tau_i} - r_{\tau_0}}{s} \in \mathbb{Z}_{q-1}^*$. Clearly, when I occurs \mathcal{A} will be able to learn $g^{m_b^*}$ and then decide $b' = b$ with probability 1. On the other hand, if I does not occur, then the uniform guess b' will be equal to b with probability exactly half. Therefore,

$$\begin{aligned} ss\text{-adv}[\mathcal{A}, \mathcal{H}] &= \left| \Pr[b' = b] - \frac{1}{2} \right| \\ &= \left| \Pr[b' = b|I] \Pr[I] + \Pr[b' = b|\bar{I}] \Pr[\bar{I}] - \frac{1}{2} \right| \\ &= \left| 1 \cdot \Pr[I] + \frac{1}{2} \cdot \Pr[\bar{I}] - \frac{1}{2} \right| \\ &= \frac{1}{2} \Pr[I], \end{aligned} \tag{7}$$

where \bar{I} denotes the complement of I .

It is left to show that $\Pr[I]$ is non-negligible in λ . We shall shortly see that the expected result is an easy consequence of (the proof of) Lemma 1, our technical lemma on PRFs with a range of Abelian group. Let F be the PRF with key space $\mathcal{K} = \mathbb{Z}_{q-1}$, domain $\mathcal{X} = \{0, 1\}^*$ and range $\mathcal{Y} = \mathbb{Z}_{q-1}$ in q -HEMAC. Let $\mathcal{T} = \mathbb{Z}_{q-1}^* \subseteq \mathbb{Z}_{q-1}$ be the subset of \mathbb{Z}_{q-1} that consists of all integers $x \in \mathbb{Z}_{q-1}$ such that $\gcd(x, q-1) = 1$. As $q = 2p + 1$ is a safe prime, we have that $|\mathcal{T}| = \phi(q-1) = p$. Then

$$\frac{|\mathcal{T}|}{|\mathcal{Y}|} = \frac{p}{2p} = \frac{1}{\alpha(\lambda)}$$

for $\alpha(\lambda) = 2$. As in the proof of Lemma 1, the adversary \mathcal{A} chooses $t + 1$ distinct PRF inputs $\tau_0, \tau_1, \dots, \tau_t$. Lemma 1 and its proof show that the probability that

there is at least one integer $i \in \{1, \dots, t\}$ such that $F_R(\tau_i) - F_R(\tau_0) \in \mathcal{T}$ is $\geq 1 - e^{-\lambda} - \text{negl}(\lambda)$. As a result,

$$\Pr[I] \geq 1 - e^{-\lambda} - \text{negl}(\lambda),$$

which is non-negligible in λ . Putting all pieces together, we have that

$$ss\text{-adv}[\mathcal{A}, \mathcal{H}] = \frac{1}{2} \Pr[I] \geq \frac{1}{2} (1 - e^{-\lambda} - \text{negl}(\lambda)).$$

Therefore, the scheme q -HEMAC is not semantically secure according to Definition 1. \square

5 Conclusions

In this paper, we show that the q -HEMAC scheme of Tran et al. [26] is not semantically secure and thus provides no confidentiality of input data in the delegation of multivariate quadratic polynomials. Our attack is simple and refutes their claim of the first verifiable computation scheme without FHE for quadratic functions. In particular, the analysis of our attack shows that a malicious cloud server is able to learn significant information of the secret key (e.g., g^s). It is an interesting open problem to fix the broken scheme, without using FHE.

Acknowledgment. The authors would like to thank the anonymous referees for the helpful comments. The research was supported by NSFC (No. 61602304) and Pujiang Talent Program (No. 16PJ1406500).

References

1. Applebaum, B., Ishai, Y., Kushilevitz, E.: From secrecy to soundness: efficient verification via secure computation. In: Abramsky, S., Gavioille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) ICALP 2010. LNCS, vol. 6198, pp. 152–163. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14165-2_14
2. Attrapadung, N., Libert, B.: Homomorphic network coding signatures in the standard model. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 17–34. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19379-8_2
3. Backes, M., Fiore, D., Reischuk, R.M.: Verifiable delegation of computation on outsourced data. In: Sadeghi, A.-R., Gligor, V.D., Yung, M. (eds.) ACM CCS 2013, pp. 863–874. ACM Press (2013)
4. Barbosa, M., Farshim, P.: Delegatable homomorphic encryption with applications to secure outsourcing of computation. In: Dunkelman, O. (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 296–312. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27954-6_19
5. 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). https://doi.org/10.1007/3-540-44448-3_41

6. Benabbas, S., Gennaro, R., Vahlis, Y.: Verifiable delegation of computation over large datasets. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 111–131. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_7
7. Boneh, D., Freeman, D.M.: Homomorphic signatures for polynomial functions. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 149–168. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20465-4_10
8. Boneh, D., Freeman, D.M.: Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 1–16. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19379-8_1
9. Catalano, D., Fiore, D.: Practical homomorphic MACs for arithmetic circuits. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 336–352. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38348-9_21
10. Catalano, D., Fiore, D., Gennaro, R., Vamvourellis, K.: Algebraic (trapdoor) one-way functions and their applications. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 680–699. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36594-2_38
11. Catalano, D., Fiore, D., Gennaro, R., Nizzardo, L.: Generalizing homomorphic MACs for arithmetic circuits. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 538–555. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54631-0_31
12. 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). https://doi.org/10.1007/978-3-642-20465-4_13
13. Catalano, D., Fiore, D., Warinschi, B.: Efficient network coding signatures in the standard model. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 680–696. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30057-8_40
14. Catalano, D., Fiore, D., Warinschi, B.: Homomorphic signatures with efficient verification for polynomial functions. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 371–389. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_21
15. Chung, K.-M., Kalai, Y., Vadhan, S.: Improved delegation of computation using fully homomorphic encryption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 483–501. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7_26
16. Freeman, D.M.: Improved security for linearly homomorphic signatures: a generic framework. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 697–714. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30057-8_41
17. Fiore, D., Gennaro, R.: Publicly verifiable delegation of large polynomials and matrix computations, with applications. In: CCS 2012, pp. 501–512 (2012)
18. Fiore, D., Gennaro, R., Pastro, V.: Efficiently verifiable computation on encrypted data. In: ACM CCS 2014, pp. 844–855. ACM (2014)
19. Gennaro, R., Gentry, C., Parno, B.: Non-interactive verifiable computing: outsourcing computation to untrusted workers. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 465–482. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7_25
20. Gennaro, R., Wichs, D.: Fully homomorphic message authenticators. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. LNCS, vol. 8270, pp. 301–320. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-42045-0_16

21. Joo, C., Yun, A.: Homomorphic authenticated encryption secure against chosen-ciphertext attack. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8874, pp. 173–192. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45608-8_10
22. Lai, J., Deng, R.H., Pang, H., Weng, J.: Verifiable computation on outsourced encrypted data. In: Kutyłowski, M., Vaidya, J. (eds.) ESORICS 2014. LNCS, vol. 8712, pp. 273–291. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11203-9_16
23. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_16
24. Papamanthou, C., Shi, E., Tamassia, R.: Signatures of correct computation. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 222–242. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36594-2_13
25. Parno, B., Raykova, M., Vaikuntanathan, V.: How to delegate and verify in public: verifiable computation from attribute-based encryption. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 422–439. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28914-9_24
26. Tran, N.H., Pang, H., Deng, R.H.: Efficient verifiable computation of linear and quadratic functions over encrypted data. In: Wang, X., Huang, X. (eds.) ASIACCS 2016, pp. 605–616. ACM (2016)



Multi-party (Leveled) Homomorphic Encryption on Identity-Based and Attribute-Based Settings

Veronika Kuchta^(✉), Gaurav Sharma, Rajeev Anand Sahu,
and Olivier Markowitch

Université libre de Bruxelles, Brussels, Belgium
veronika.kuchta@ulb.ac.be

Abstract. We present constructions of CPA-secure (leveled) homomorphic encryption from learning with errors (LWE) problem. We use the construction introduced by Gentry, Sahai and Waters ‘GSW’ (CRYPTO’13) as building blocks of our schemes. We apply their *approximate eigenvector* method to our scheme. In contrast to the GSW scheme we provide extensions of the (leveled) homomorphic identity-based encryption (IBE) and (leveled) homomorphic attribute-based encryption (ABE) on the multi-identity and multi-attribute settings respectively. We realize the (leveled) homomorphic property for the multi-party setting by applying tensor product and natural logarithm. Tensor product and natural logarithm allow to evaluate different ciphertexts computed under different public keys. Similar to the GSW scheme, our constructions do not need any evaluation key, which enables evaluation even without the knowledge of user’s public key.

1 Introduction

Since the proposal of public key cryptography (PKC), construction of an efficient encryption has always been interesting and challenging problem. The first efficient constructions were Boneh-Franklin identity-based encryption (IBE) [8] and Cocks’s IBE [20]. The former uses pairing over elliptic curve and the later was based on quadratic residuosity. After years when lattices were found useful to design post-quantum constructions, Gentry et al. [22] proposed new possibility to design IBE from lattices. The topic of IBE has been widely studied in cryptography and various possibilities on it have been explored. Attribute-based encryption (ABE) is a special form of IBE, where identities are fine grained and replaced by particular attributes of the users. Homomorphic encryption [21] is another special encryption which has been studied parallel to ABE and serves various useful application in cryptography. In a wide review of PKC of last decade these topics namely IBE, ABE, homomorphic encryption, lattice-based encryption have gained much attentions as they cover a major section of recent research in PKC. Since the last couple of years researchers have focused to achieve mixed functionality by combining two or more properties in a single protocol.

In this paper we achieve compact encryption schemes by combining functionalities of above crucial notions. Below we discuss each individual topic with their state of art.

Identity-Based Encryption. An identity-based encryption was introduced by Shamir [38] and it allows users to send encrypted messages knowing only the recipient's identity. Practical implementations were proposed only many years later. The first IBE was given by Boneh and Franklin [8] and since then it got a lot of attention from the cryptographic community. The first construction using lattices was given by Gentry et al. [22]. Other IBE construction were presented in [1, 2, 15, 34] and proved secure in the standard model using LWE assumption. Gentry et al.'s construction [23] allows to construct a fully homomorphic identity-based encryption which is also secure under the LWE hardness problem. The shortcoming of an IBE scheme is that it cannot have a unique identifier for each person. Usually users are identified by their attributes. This leads to the next cryptographic construction, called attribute-based encryption. In a nutshell, an attribute-based encryption represents a generalization of an IBE scheme, since in an IBE scheme ciphertexts are encrypted under one attribute, the identity. In contrast, an attribute-based encryption provides a scheme where ciphertexts are associated with many attributes. In the next paragraph we give an overview of this scheme.

Attribute-Based Encryption. An attribute-based encryption (ABE) scheme that allows fine-grained access control on encrypted data, was introduced by Sahai and Waters [37]. The idea of an ABE is to associate ciphertexts and private keys with sets of descriptive attributes such that the decryption is only possible if the overlap of these two sets is sufficient. There are two flavors of an attribute-based encryption, a key-policy ABE (KP-ABE) and a ciphertext-policy ABE (CP-ABE). A key-policy ABE handles with ciphertexts which are annotated with attributes while private keys which are associated with certain access structures. The reason for these access structure is to specify which ciphertexts can be chosen to be decrypted by user. The other ABE flavor, a ciphertext-policy model was introduced by Bethencourt et al. [6] and by Cheung and Newport [17]. A work that analyzes the first expressive construction was presented by Goyal et al. [25] in the standard model. Other standard model CP-ABE constructions were provided by Waters [41] and Lewko et al. [28]. In CP-ABE scheme attribute sets are assigned to private keys, where the sender specifies an access policy such that receiver's attribute set can comply with it. Attrapadung et al. [5] introduced an ABE scheme with constant-size ciphertexts. Goyal et al. [26] generalized those techniques from [37] and introduced a new technique where user's key is associated with a tree-access structure and the leaves are associated with attributes. User is able to decrypt a ciphertext if the attributes associated with a ciphertext satisfy key's access structure. This technique differs from secret-sharing schemes by the fact that any communication between different parties is forbidden. An ABE scheme which allows a group of authorities to monitor only a certain subset of attributes was developed by Chase [16]. This multi-authority ABE construction allows to corrupt any number of attribute authorities but

guarantees security of encryption as long as not all required attributes can be obtained from those corrupt authorities. The first ABE construction based on lattices was introduced by Boyen [10]. Since these both discussed encryption flavours provide attractive features for security issues of cloud computing, we recall in the following paragraph the motivation of cryptographic applications in cloud computing.

Homomorphic Encryption. Our paper handles with leveled homomorphic encryption which represents a meaningful field of fully homomorphic encryption. The latter had an enormous development in recent years and became an attractive cryptographic tool due to its functionality which allows to evaluate certain computations on encrypted data sets. Gentry [21] introduced the first fully homomorphic encryption scheme based on cryptographic assumptions. His construction is based on the hardness of problems defined on *ideal* lattices, which are not deeply and well-studied yet. The benefit of using these ideal lattices is that they support addition and multiplication of homomorphic encryption. Other fully homomorphic encryption schemes which are not based on lattices but relied on ideals in rings were presented in [14, 39, 40]. Brakerski and Vaikuntanathan [13] presented a fully homomorphic scheme based on a well-studied assumption - called the learning with errors assumption (LWE). A comparatively simple fully homomorphic encryption scheme also based on LWE problem has been presented by Gentry et al. [23]. They presented a new technique which they called *approximate eigenvector* method where homomorphic addition and multiplication are provided by simple matrix addition and multiplication. In contrast to previous fully homomorphic schemes, Gentry et al.'s construction does not require any evaluation key and evaluation can even be calculated without knowing user's public key. This feature allowed the authors to construct the first fully homomorphic identity-based encryption.

Lattice-Based Encryption. Lattice-based cryptography developed rapidly and became a significant part of cryptographic primitives in the last few years. Cryptosystems based on the hardness of lattice problems became so powerful because of their provable security guarantees, simplicity, potential efficiency and their security against quantum attacks. This new kind of cryptography which represents a part of post-quantum cryptography, was invented by the breakthrough results of Ajtai [4] in 1996. There are so far several constructions of lattice-based primitives, such like one-way functions [31], collision resistant hash functions [4], signatures [9], public-key encryption [35, 36], encryption for threshold functions [3], identity based encryption [15, 22], lossy trapdoor functions [22]. Agrawal and Boyen [2] presented an IBE construction based on hard problems in lattices in the standard model. The construction is anonymous, which means that it is usable for searching on encrypted data because the ciphertext does not reveals the identity of the recipient. The most of these cryptographic applications [2, 3, 22, 36] are based on the presumed hardness of LWE (Learning With Errors) problem. One of the connections between lattices and LWE is given by a polynomial-time quantum algorithm that solves standard lattice problems, given access to an oracle that solves the LWE problem. There are other algorithms

which run in exponential time, e.g. the Blum et al. [7], Micciancio and Voulgaris [32] algorithms are the best known algorithms for solving LWE problem which run in time $2^{\mathcal{O}(n)}$. Most of the cryptosystems based on lattices [2, 3, 22] rely on the *Learning With Errors* (LWE) problem which was introduced by Regev [36]. Before we can present our contribution we recall shortly the basics of identity-based encryption and attribute-based encryption as it takes an important part of our underlying work.

Cloud computing. Cloud computing allows users to use big data storage and computation capabilities at a very low price. Since its invention, cloud computing became an important application for the recent cryptographic protocols. Storing data on a cloud system enables users to reduce purchase and maintaining cost of computing and storage tools which attracted a lot of attention from computer users. When personal and confidential data is outsourced to a cloud server there is a need to guarantee the customers that their data will not be watched by anybody. Therefore cryptographic encryption became a crucial tool in cloud security. Distributing the role and responsibility of a single party involved in a cloud application, allowed to improvements for the cloud security. The idea of distributing the power of a single party under multiple parties in a multi-party protocol has been developed by Kamara et al. [27]. López-Alt et al. [30] presented a multi-key fully homomorphic encryption from the NTRU encryption scheme that allows computation of ciphertexts under different unrelated keys. In the following paragraph we present our main contribution which encompasses the aforementioned cryptographic constructions and we suggest how to apply our construction to cloud computing.

Contribution. In contrast to the scheme in [23] which introduced a single-authority leveled homomorphic attribute-based encryption (FHABE) and single identity-based encryption (FHIBE), we present in our work two constructions employing multiple authorities in case of attribute-based encryption or multiple identities in case of identity-based encryption where a ciphertext is encrypted under different public keys. The construction in [23] is advantageous in comparison to previous fully homomorphic encryption [12] which required existence of evaluation keys to evaluate several ciphertexts. This is also an advantage of our scheme, because our evaluator can execute homomorphic operations without using any evaluation key. Our scheme presents an alternative construction of a leveled homomorphic IBE and leveled homomorphic ABE schemes employing multiple identities. In addition to the technique from [23] we recall the well-known tensor product in order to allow the homomorphic encryption which supports multiplication operations of different ciphertexts using multiple identities in case of IBE scheme and multiple authorities in case of ABE scheme. In comparison to the López-Alt et al. work [30] which provided a multi-party construction, we introduce a new technique for the addition of ciphertexts without use of evaluation keys, which makes our construction more advantageous than the scheme in [30].

Related Work. The first multi-key fully homomorphic encryption introduced by López-Alt et al. [30] relies on a non-standard assumption, called the Decisional Small Polynomial Ratio assumption and employs evaluation keys during the evaluation process. Clear and McGoldrick [18] introduced the first multi-identity and multi-key leveled FHE and multi-identity fully homomorphic IBE (FHIBE) scheme secure under the hardness of learning with errors assumption. They presented a new compiler, which converts a single-identity FHIBE scheme into a multi-identity FHIBE scheme. Their technique involves a masking system which makes the computations more difficult than in case of a single-identity FHIBE. In their later work, Clear and McGoldrick [19] presented a pure fully homomorphic attribute-based encryption scheme which is the first achievement without using indistinguishability obfuscation. However they couldn't achieve a pure fully homomorphic multi-attribute based encryption scheme. We note that we do not claim achievement of pure fully homomorphic property, but we claim achievement of homomorphism according to the multiplication of ciphertexts using the new mathematical constructs such like tensor product and natural logarithm. The latter guarantees that the compactness property of the evaluated ciphertext keeps preserved. In contrast to the construction in [18], our work provides a simple and alternative construction of multi-identity homomorphic IBE scheme and a new construction of multi-authority leveled homomorphic ABE using the natural logarithm as an auxiliary for the homomorphic evaluation. Brakerski et al. [11] showed that a cross-evaluation of attributes is possible, such that the size of the ciphertext remains independent of attributes. Mukherjee and Wichs [33] showed how to homomorphically evaluate data which as encrypted under different public keys.

2 Preliminaries

In this section we recall learning with errors problem and the flattening technique from [23]. Other preliminaries are provided in the appendix.

Definition 1 (LWE Problem). *For an integer q and error distribution χ , the goal of $LWE_{q,\chi}$ in n dimensions problem is to find $\mathbf{s} \in \mathbb{Z}_q^n$ with overwhelming probability, given access to any arbitrary $\text{poly}(n)$ number of samples from $A_{\mathbf{s},\chi}$ for some random \mathbf{s} .*

In matrix form this problem looks as follows: collecting the vectors $\mathbf{a}_i \in \mathbb{Z}_q^n$ into a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and the error terms $e_i \in \mathbb{Z}$ and values $t_i \in \mathbb{Z}_q$ as the entries of the m -dimensional vector $\mathbf{t} \in \mathbb{Z}_q^m$ we obtain the input \mathbf{A} , $\mathbf{t} = \mathbf{A}^t \mathbf{s} + \mathbf{e} \pmod q$.

2.1 Flattening Ciphertexts

In this paragraph we recall the technique from [23] which keeps ciphertexts strongly bounded. It was used to realize the first leveled homomorphic identity-based and leveled homomorphic attribute-based encryption as showed in [23].

Using transformations from [13], vectors can be modified without affecting dot products.

We assume two vectors $\mathbf{a}, \mathbf{b} \in \mathbb{Z}_q^k$ and set $l = \lfloor \log_2 q \rfloor + 1$ and $N = k \cdot l$. Let $\text{BitDecomp}(\mathbf{a})$ be the N -dimensional vector $(a_{1,0}, \dots, a_{1,l-1}, \dots, a_{k,0}, \dots, a_{k,l-1})$, where $a_{i,j}$ is the j -th bit in a_i 's binary representation. For some vector $\mathbf{a}' = (a_{1,0}, \dots, a_{1,l-1}, \dots, a_{k,0}, \dots, a_{k,l-1})$, let

$$\text{BitDecomp}^{-1}(\mathbf{a}') = \left(\sum_{j=0}^{l-1} 2^j \cdot a_{1,j}, \dots, \sum_{j=0}^{l-1} 2^j \cdot a_{k,j} \right)$$

be the inverse of BitDecomp , which is well defined. It means that even if the input is not a bit-vector, the inverse is well-defined.

Let $\text{Flatten}(\mathbf{a}') = \text{BitDecomp}(\text{BitDecomp}^{-1}(\mathbf{a}'))$ be a N -dimensional bit vector. For a matrix A , let $\text{BitDecomp}(A)$, $\text{BitDecomp}^{-1}(A)$, $\text{Flatten}(A)$ being applied to each row of A . Let $\text{Powerof2}(\mathbf{b}) = (b_1, 2b_1, \dots, 2^{l-1}b_1, \dots, b_k, 2b_k, \dots, 2^{l-1}b_k)$. Observe following properties for any N -dimensional \mathbf{a}' :

$$\begin{aligned} \langle \text{BitDecomp}(\mathbf{a}, \text{Powerof2}(\mathbf{b})) \rangle &= \langle \mathbf{a}, \mathbf{b} \rangle \\ \langle \mathbf{a}', \text{Powerof2}(\mathbf{b}) \rangle &= \langle \text{BitDecomp}^{-1}(\mathbf{a}', \mathbf{b}) \rangle = \langle \text{Flatten}(\mathbf{a}'), \text{Powerof2}(\mathbf{b}) \rangle. \end{aligned}$$

The leveled homomorphic encryption (LHE) scheme from [23] works as follows. For suitable parameters $q, n, m = \mathcal{O}(n \log q)$ the LWE instance consists of a $m \times (n+1)$ matrix A , s.t. there is a vector $s \in \mathbb{Z}_q^{n+1}$, where the first entry is 1 and $e = A \cdot s$ is a small error vector. We assume that A is public and s is secret. A ciphertext C encrypts μ if $C \cdot \mathbf{v} = \mu \mathbf{v} + e$, where \mathbf{v} is a N -dimensional secret key. To decrypt message μ , the i -th row C_{id_i} is extracted from C and $x \leftarrow \langle C_{id_i}, \mathbf{v} \rangle = \mu v_i + e_i$ computed. The vector \mathbf{v} is called approximate eigenvector. Let $\mathbf{v} = \text{Powerof2}(\mathbf{s})$, which is a vector of dimension $N = (n+1) \cdot l$ for $l = \lfloor \log_2 q \rfloor + 1$. It holds: $\text{Flatten}(C) \cdot \mathbf{v} = C \cdot \mathbf{v}$.

To encrypt a message $\mu \in \mathbb{Z}_q$, a random matrix $R \in \{0, 1\}^{N \times m}$ is generated and $C = \text{Flatten}(\mu \cdot I_N + \text{BitDecomp}(R \cdot A))$ computed. Note that Flatten operation does not affect the product with \mathbf{v} , i.e.

$$C \cdot \mathbf{v} = \mu \cdot \mathbf{v} + \text{BitDecomp}(R \cdot A) \cdot \mathbf{v} = \mu \cdot \mathbf{v} + R \cdot A \cdot s = \mu \cdot \mathbf{v} + \text{small}.$$

3 Leveled Homomorphic Multi-identity-Based Encryption

Intuition. As mentioned before, the first leveled homomorphic multi-identity-based encryption scheme was introduced by Clear and McGoldric [18]. The idea is to extend the single-identity setting to the multi-identity setting, such that each ciphertext encrypts a different message under a different identity. To enable the evaluation of different ciphertexts, we propose a new technique using the already presented mathematical tool, “tensor product” for multiplication of those ciphertexts. In this section we present the compilation of our leveled

homomorphic multi-identity-based encryption (LHMIBE) from LWE-based IBE scheme, where the ciphertexts are computed on different identities and evaluation procedure calculates a function on input of these ciphertexts. We provide the syntax of our LHMIBE scheme in the following definition.

Definition 2 (LHMIBE). *A leveled homomorphic multi-identity-based encryption scheme consists of the following five algorithms:*

Setup(1^λ): *On input the security parameter 1^λ , $\lambda \in \mathbb{N}$ output the master key pair (msk, mpk) .*

Extract(mpk, msk, id_i): *On input a master secret key msk and an identity id_i , output (id_i, sk_{id_i}) .*

Encrypt(mpk, id_i, μ_i): *On input mpk , an identity id_i and a message μ_i , output a ciphertext C .*

Eval($F, C_{id_1}, \dots, C_{id_n}$): *On input a function F , n different ciphertexts $C_{id_i}, i \in [n]$, output \hat{C} .*

Decrypt($\hat{C}, sk_{id_1}, \dots, sk_{id_n}$): *On input n secret keys $\{sk_{id_i}\}_{i \in [n]}$ and evaluated ciphertext \hat{C} , output $\hat{\mu}(= F(\mu_1, \dots, \mu_n))$.*

Further, we propose a transformation from an LWE-based IBE scheme into a leveled homomorphic multi-identity-based encryption (LHMIBE) scheme, that supports homomorphic operations on ciphertexts produced for different identities. We rely on the following properties of LWE-based IBE schemes [1, 15, 22]:

- (1) The decryption key for identity id_i and the corresponding ciphertext for id_i , are $sk_{id_i}, C_{id_i} \in \mathbb{Z}_q^{n'}$. We extend the decryption key by adding 1 as the first component.
- (2) If C_{id_i} encrypts 0, then $\langle C_{id_i}, sk_{id_i} \rangle$ is small.
- (3) Encryptions of 0 are indistinguishable from uniform vectors over \mathbb{Z}_q (under LWE assumption).

We stress that the technique from [23] cannot be applied to our setting where ciphertexts can possibly be encryptions under different identities. Our construction offers an alternative evaluation technique based on tensor product and natural logarithm. The evaluation function F is a homomorphic function, allowing to compute a product of ciphertexts by summing the evaluated individual ciphertexts. To provide this functionality, we use the natural logarithm. Since the ciphertext C_{id_i} is represented as a $N \times N$ matrix in the following paragraphs, and the secret keys are N -dimensional vectors, i.e. $C_{id_i} \in \mathbb{Z}_q^{N \times N}$, $sk_{id_i} \in \mathbb{Z}_q^N$, the evaluation function F has the following form:

$$\begin{aligned} F(C_{id_1}, \dots, C_{id_n}) &= \log \left(\bigotimes_{i=1}^n C_{id_i} \right) = \log [(C_{id_1} \otimes I_N) \cdots (I_{N^{i-1}} \otimes C_{id_n})] \\ &= (C_{id_1} \otimes I_N) \prod_{i=1}^{n-1} (I_{N^i} \otimes C_{id_{i+1}}) = \log (C_{id_1} \otimes I_N) + \log (I_N \otimes C_{id_2}) + \end{aligned}$$

$$\dots + \log(I_{N^{n-1}} \otimes C_{id_n}) = \log(C_{id_1} \otimes I_N) + \sum_{i=1}^{n-1} \log(I_{N^i} \otimes C_{id_{i+1}}).$$

The n different decryption keys $\{\mathbf{v}_{id_i}\}_{i \in [n]}$ operate on the resulting ciphertext as follows:

$$\begin{aligned} F(C_{id_1}, \dots, C_{id_n}) + \log\left(\bigotimes_{i=1}^n \mathbf{sk}_{id_i}\right) &= \log\left(\bigotimes_{i=1}^n C_{id_i}\right) + \log\left(\bigotimes_{i=1}^n \mathbf{sk}_{id_i}\right) \\ &= \log\left[\left(\bigotimes_{i=1}^n C_{id_i}\right) \left(\bigotimes_{i=1}^n \mathbf{sk}_{id_i}\right)\right] = \log\left[\bigotimes_{i=1}^n C_{id_i} \mathbf{sk}_{id_i}\right]. \end{aligned}$$

3.1 The Scheme

Let Σ be a LWE-based IBE scheme with the above properties. Our transformation of Σ into an LHMIBE scheme proceeds as follows:

Setup(1^λ): Run the **Setup** algorithm of Σ to generate (mpk, msk) .

Extract(mpk, msk, id_i): Run the extraction algorithm of Σ scheme to compute $sk_{id_i}^{ibe} \in \mathbb{Z}_q^m$, which is the decryption key of IBE scheme. Then set $\mathbf{s}_{id_i} := \mathbf{sk}_{id_i}' = (1, sk_{id_i}^{ibe}) \in \mathbb{Z}_q^{m+1}$. Compute the decryption key of LHMIBE scheme as **Powerof2**(\mathbf{s}_{id_i}) = $\mathbf{v}_{id_i} \in \mathbb{Z}_q^{l \cdot (m+1)}$, where $\mathbf{v}_{id_i} = (v_{id_i,1}, \dots, v_{id_i,N})$ for $i \in \{1, \dots, n\}$, $N = l(m+1)$. Output (i, \mathbf{v}_{id_i}) .

Encrypt(mpk, id_i, μ_i): To encrypt the message $\mu_i \in \{0, 1\}$ for $i \in [n]$, invoke **Encrypt** of Σ in order to compute $N = l \cdot (m+1)$ encryptions of 0. The resulted ciphertext is denoted by C'_{id_i} . Taking C'_{id_i} , compute the ciphertext of LHMIBE as follows: $C_{id_i} = \text{Flatten}(\mu \cdot I_N + \text{BitDecomp}(C'_{id_i}))$.

Eval($mpk, C_{id_1}, \dots, C_{id_n}, F$): Take as input ciphertexts, $C_{id_1}, \dots, C_{id_n}$ and an evaluation function F . Output $F(C_{id_1}, \dots, C_{id_n}) = \log(\bigotimes_{i=1}^n C_{id_i}) = \hat{C}$.

Decrypt($mpk, \hat{C}, \mathbf{v}_{id_1}, \dots, \mathbf{v}_{id_n}$): On input master public key mpk , evaluated ciphertext \hat{C} and the n secret keys $\mathbf{v}_{id_1}, \dots, \mathbf{v}_{id_n}$, compute $\log[(\bigotimes_{i=1}^n C_{id_i}) (\bigotimes_{i=1}^n \mathbf{v}_{id_i})] = \log[\bigotimes_{i=1}^n C_{id_i} \mathbf{v}_{id_i}]$.

Correctness. To show the validity of decryption procedure, we observe the following computation:

$$\begin{aligned} &\log(\mathbf{v}_{id_1}^{-1} \otimes \dots \otimes \mathbf{v}_{id_n}^{-1}) + \log(C_{id_1} \otimes \dots \otimes C_{id_n}) + \log(\mathbf{v}_{id_1} \otimes \dots \otimes \mathbf{v}_{id_n}) \\ &= \log\left[\bigotimes_{i=1}^n \mathbf{v}_{id_i}^{-1} \bigotimes_{i=1}^n C_{id_i} \mathbf{v}_{id_i}\right] = \log\left[\bigotimes_{i=1}^n (\mu_i \mathbf{v}_{id_i} \mathbf{v}_{id_i}^{-1} + e_i \mathbf{v}_{id_i})\right] \\ &\stackrel{\text{exp}(\cdot)}{\implies} \exp\left(\log\left[\prod_{i=1}^n \mu_i + \text{"small"}\right]\right) = \prod_{i=1}^n \mu_i + \text{"small"} \approx \mu_1 \cdot \dots \cdot \mu_n. \end{aligned}$$

Note: $\mathbf{v}_{id_i}^{-1} := (\mathbf{v}_{id_i,1}^{-1}, \dots, \mathbf{v}_{id_i,N}^{-1})$ is defined as inverse of the components of $\mathbf{v}_{id_i} := (\mathbf{v}_{id_i,1}, \dots, \mathbf{v}_{id_i,N})$. Furthermore holds $C_{id_i} \mathbf{v}_{id_i} = (\mu_i \mathbf{v}_{id_i} + e_i)$.

3.2 Security Analysis

We prove in this section that the resulting LHMIBE construction is IND-ID-CPA secure according to the following Definition below. We note that an adversary obtains at most $n - 1$ secret keys. Since the security of our construction is given in the CPA model we assume an adversary having access to the extract oracle which on input an identity outputs the corresponding secret key corresponding to that identity. We provide the limits of an adversary by not allowing her to query the extraction oracle on the same identity which was used during the encryption process. The security definition is given below:

Definition 3 (LHMIBE Indistinguishability). *Let \mathcal{A}_{ind} be a probabilistic polynomial time adversary against the IND-ID-CPA security of the LHMIBE scheme, F an evaluation function and $b \in \{0, 1\}$ a bit which is associated with the following experiment $\mathbf{Exp}_{LHMIBE, \mathcal{A}_{ind}}^{IND-ID-CPA-b}(1^\lambda)$:*

1. $(mpk, msk) \xleftarrow{r} \mathbf{Setup}(1^\lambda)$.
2. $F, st, (id_1^*, \mu_{1,0}, \mu_{1,1}), \dots, (id_n^*, \mu_{n,0}, \mu_{n,1}) \leftarrow \mathcal{A}_{ind}^{\mathcal{O}\mathbf{Extract}(\cdot)}(mpk, find)$.
3. Compute $\{\mathbf{v}_{id_i}\}_{i \in [n-1]} \leftarrow \mathbf{Extract}(mpk, msk, id_i)$ and set $S = \{(id_i, \mathbf{v}_{id_i})\}_i$ with $i \in [n]$. At the beginning of the experiment the set S is empty.
4. If $(id_i, \cdot) \notin S$, run $\mathbf{v}_{id_i} \leftarrow \mathbf{Extract}(mpk, msk, id_i)$ and add $(id_i, \mathbf{v}_{id_i})$ to S .
5. Compute $C_{i,b}^* = \mathbf{Encrypt}(mpk, id_i^*, \mu_{i,b})$, with different identities $i \in [n]$.
6. $\hat{C}_b = \mathbf{Eval}(mpk, C_{1,b}^*, \dots, C_{n,b}^*, F)$.
7. $b' \leftarrow \mathcal{A}_{ind}^{\mathcal{O}\mathbf{Extract}(\cdot)}(\hat{C}_b, \{C_{i,b}^*\}_{i \in [n]}, st)$, where $b' \in \{0, 1\}$.

$\mathcal{O}\mathbf{Extract}(id_i)$: On input id_i the oracle checks if (id_i, \cdot) is in the list S . If so, returns \mathbf{v}_{id_i} to the adversary. Otherwise the oracle runs $\mathbf{v}_{id_i} \xleftarrow{r} \mathbf{Extract}(mpk, msk, id_i)$ and gives \mathbf{v}_{id_i} to \mathcal{A} . If $|S| > n - 1$, the oracle returns \perp .

\mathcal{A}_{ind} wins if $b' = b$ meaning that \mathcal{A}_{ind} can distinguish whether \hat{C}_b was produced from $C_{1,0}, \dots, C_{n,0}$ or $C_{1,1}, \dots, C_{n,1}$ and \mathcal{A}_{ind} did not issue secret key extraction query on id_i^* . The advantage of \mathcal{A}_{ind} is $\mathbf{Adv}_{LHMIBE, \mathcal{A}_{ind}}^{IND-ID-CPA} =:$

$$|Pr[\mathbf{Exp}_{LHMIBE, \mathcal{A}_{ind}}^{IND-ID-CPA-0}(1^\lambda) = 1] - Pr[\mathbf{Exp}_{LHMIBE, \mathcal{A}_{ind}}^{IND-ID-CPA-1}(1^\lambda) = 1]|.$$

The LHMIBE scheme is IND-ID-CPA secure if $\mathbf{Adv}_{LHMIBE, \mathcal{A}_{ind}}^{IND-ID-CPA}$ is negligible.

Remark 1. Furthermore, our LHMIBE scheme has to fulfill the *compactness* property which is formulated as following: There exists a polynomial $p(\lambda, L, \cdot)$, such that $|\hat{C}| \leq p(\lambda, L, \cdot)$, where L is the depth of the ciphertext. We note that this property is satisfied by our construction since \hat{C} is the result of natural logarithm on input of individual ciphertexts. W.l.o.g. for sufficiently large arguments of the natural logarithm, it is obvious that $\log(\cdot) \leq p(\cdot)$.

Theorem 1. *Our LHMIBE scheme is IND-ID-CPA secure given that (\mathbb{Z}_q, n, χ) -LWE is hard.*

Proof. Let \mathcal{A}_{ind} be an adversary against IND-ID-CPA security of LHMIBE scheme. We use \mathcal{A}_{ind} to construct an algorithm \mathcal{B} against the IND-ID-CPA security of the underlying Σ scheme which was proven secure in [23]. Thereafter we use \mathcal{B} to construct an adversary \mathcal{C} against LWE problem. The challenger \mathcal{B} sets the public parameters of Σ equal to (mpk, msk) of LHMIBE scheme.

Key Extract Queries: When \mathcal{A}_{ind} issues queries on id'_i where $(id'_i, \cdot) \notin S$ and $id'_i \neq id_i^*$, the algorithm \mathcal{B} , which controls the set S , is invoked on that input and forwards the query to its own oracle $\mathcal{O}_{\text{Extract}}_{\Sigma}$ of the underlying IBE scheme Σ which returns sk_{id_i} to \mathcal{B} . Algorithm \mathcal{B} sets $s_{id'_i} = (1, sk_{id'_i}) \in \mathbb{Z}_q^{m+1}$ and $v_{id'_i} = \text{Powerof2}(s_{id'_i}) \in \mathbb{Z}_q^{l(m+1)}$ and sends $v_{id'_i}$ to \mathcal{A}_{ind} . At some point \mathcal{A}_{ind} outputs $(id_1^*, \mu_1), \dots, (id_n^*, \mu_n)$ on which it wants to be challenged. If \mathcal{B} didn't guess the identities and messages correctly then it aborts the simulation. The probability that \mathcal{B} does not abort is $1/|\mathcal{M}|^2$, where \mathcal{M} is the message space.

Challenge Ciphertext: Algorithm \mathcal{B} computes $C_i^* \leftarrow \text{Encrypt}(mpk, id_i^*, m_i)$ by running the encryption algorithm of Σ scheme and taking as input the randomly guessed id_i^* . \mathcal{A}_{ind} computes challenge ciphertext $C_i^* \leftarrow \text{Encrypt}(pp, id_i^*, m_i)$. \mathcal{A}_{ind} does the same for the remained $n - 2$ identities. \mathcal{B} simulates F by randomly choosing $F \xleftarrow{r} \mathbb{Z}_q$ and sends it to \mathcal{A}_{ind} .

Guess: Simulator \mathcal{B} issues up to q_E queries on id_i and outputs a guess b' . After making additional queries \mathcal{A}_{ind} outputs a guess b . The probability that $b' = b$ is $\frac{1}{q_E}$. Thus the advantage that \mathcal{A}_{ind} wins the game is given by $\text{Adv}_{\mathcal{B}} \geq \frac{1}{q_E |\mathcal{M}|^2} \text{Adv}_{\mathcal{A}_{ind}}$.

Reduction to LWE problem: Now we assume an adversary \mathcal{C} against LWE problem which simulates the outputs for adversary \mathcal{B} against Σ scheme. The instance of LWE problem is given as a sampling oracle \mathcal{O} . This oracle can be either purely random \mathcal{O}_r or pseudo-random \mathcal{O}_s for some secret $s \in \mathbb{Z}_q^N$, where $N = l(m+1)$. \mathcal{C} queries from his sampling oracle \mathcal{O} and receives for each request i a fresh pair $(a_i, t_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$. In the next step \mathcal{B} chooses target identity it wants to attack id^* . The challenger \mathcal{C} simulates for \mathcal{B} the public parameters (mpk, msk) using LWE samples and sends them to \mathcal{B} . When \mathcal{B} issues private key extraction queries on id_i , \mathcal{C} simulates them using the samples which it received from its oracle \mathcal{O} that statistically close to uniform values. \mathcal{C} sends the simulated values to \mathcal{B} .

The simulation of the challenge ciphertext proceeds in a similar manner using as input entries from the LWE instance. Finally simulator \mathcal{C} sends the ciphertext to \mathcal{B} . For the simulation of the ciphertext we differ between two oracles. When the LWE oracle is given by \mathcal{O}_s (i.e. it is pseudo-random), the ciphertext is randomly distributed including some random noise vector which is distributed corresponding to the distribution Φ_{α}^m , which describes a certain noise distribution over \mathbb{Z}_q , as showed in [36]. When \mathcal{O} is given by \mathcal{O}_r then the ciphertext is uniform and independent over \mathbb{Z}_q^N , for some n' . Eventually the simulated ciphertext is always uniform in $\mathbb{Z}_q \times \mathbb{Z}_q^N$. After issuing additional queries, \mathcal{B} guesses a bit b' . The LWE adversary \mathcal{C} outputs its guess as the result of the LWE challenge.

Finally we follow that \mathcal{C}' 's advantage in solving LWE is at least the same as \mathcal{B}' 's advantage in distinguishing the ciphertext from a random value, i.e.: $\text{Adv}_{\mathcal{C}} \geq \frac{1}{q_E} \text{Adv}_{\mathcal{B}}$. \square

4 Leveled Homomorphic Attribute-Based Encryption in Single and Multi-authority Setting

In this section we extend the definition of a single setting attribute-based encryption introduced in [23] and present a leveled homomorphic attribute-based encryption (LHABE). We first define a LHABE scheme assuming existence of a single attribute authority, which is responsible for the generation of the secret keys corresponding to a certain string. This string can either describe an attribute set in case of a ciphertext-policy ABE scheme or the string can be related to an access policy in case of key-policy ABE scheme. We do not specify the definition for one of the mentioned flavours of ABE scheme. Instead, we provide a general definition where attributes and policy are represented by certain strings. To do so, we assume that a leveled homomorphic ABE scheme is associated to some computable relation $R(x, y)$ for $x \in \{0, 1\}^l, y \in \{0, 1\}^{l'}$ as it was showed in [23].

Gentry et al. [23] mentioned the possibility of extension of their scheme so that the evaluation algorithm operates under multiple indices x_1, \dots, x_n . The decryption process can rely on different possibilities. The result can be decrypted using either the same secret key sk_y such that $R(x_i, y) = 1$ for all $i \in [1, k]$ or using different secret keys $sk_{y_1}, \dots, sk_{y_k}$ such that $R(x_i, y_j) = 1$ for $i, j \in [1, k]$. Our evaluation techniques based on tensor product and natural logarithm allow us to realize these extensions. We note that in first case where we have only one decryption key sk_y and different strings x_i , we can provide a single authority ABE scheme whose ciphertexts are encrypted under different indices, while in second case with different secret keys sk_{y_j} we can construct the first leveled homomorphic ABE scheme employing multiple authorities, such that each secret key can be generated by a different authority. In this section we present the two extensions of [23], a leveled homomorphic single authority ABE and a leveled homomorphic multi-authority ABE (LHMABE) schemes.

4.1 Leveled Homomorphic ABE Scheme (LHABE)

In this section we introduce a leveled homomorphic ABE scheme that operates on different indices x_i . Since the construction in [23] didn't provide a concrete scheme over different indices, we resolve this drawback and instantiate in our work a construction of a LHABE scheme where distinct messages μ_i are encrypted using another public string x_i . The decryption process is possible if the decryption key which was generated on a fixed chosen string y is valid and the following relation holds: $R(x_i, y) = 1$ for all $x_i \in \{0, 1\}^l$.

Syntax. A leveled homomorphic ABE scheme consists of the following algorithms:

Setup(1^λ): On input a security parameter 1^λ , output (mpk, msk) .

KeyGen(mpk, msk, y): On input (mpk, msk) , a string y generate sk_y .

Encrypt(mpk, m_i, x_i): On input a master public key mpk , a message m_i and a string x_i , output a ciphertext C_i for $i \in \{1, \dots, k\}$.

Eval($mpk, F, \{x_i\}_{i \in [k]}, C_1, \dots, C_k$): On input mpk , an evaluation function F , set of strings $\{x_i\}_{i \in [k]}$ and a set of k ciphertexts C_1, \dots, C_n homomorphically evaluate F and output \hat{C} .

Decrypt(mpk, \hat{C}, sk_y): On input master public key mpk , an evaluated ciphertext \hat{C} and the secret key sk_y , decrypt the function $\hat{C} = F(C_1, \dots, C_k)$ if $R(x, y) = 1$.

In the following definition we present a leveled homomorphic attribute-based encryption (LHABE) which is compiled from a secure LWE based attribute-based encryption scheme.

The Scheme. Let Σ' denote an LWE-based attribute-based encryption scheme. A leveled homomorphic ABE scheme consists of the following algorithms:

Setup(1^λ): On input security parameter, run **Setup** algorithm of Σ' and generate authority's public key and authority's secret key (apk, ask) .

Extract(apk, ask, y): Run the **KeyGen** algorithm of Σ' scheme to compute $sk_y \in \mathbb{Z}_q^m$, which is the decryption key of that scheme embedding a string $y \in \{0, 1\}^l$ into the key. Set $s := sk'_y = (1, sk_y) \in \mathbb{Z}_q^{m+1}$. It computes the decryption key of LHABE scheme as **Powerof2**(s) = $\mathbf{v}_y \in \mathbb{Z}_q^{l \cdot (m+1)}$.

Encrypt(apk, x_i, μ_i): On input authority's public key apk , an attribute string $x_i, i \in [n]$ with $R(x_i, y) = 1$ and a message $\mu_i, i \in [n]$ run **Encrypt** of ABE scheme Σ' in order to compute $N = l \cdot (m + 1)$ encryptions of 0. The result is denoted by C'_i . Taking C'_i compute: $C_i = \text{Flatten}(\mu \cdot I_N + \text{BitDecomp}(C'_i))$.

Eval($apk, \{x_i\}_{i \in [n]}, C_1, \dots, C_n, F$): Take as input apk , the ciphertexts, C_1, \dots, C_n on messages μ_1, \dots, μ_n and an evaluation function F . Output: $F(C_1, \dots, C_n) = \log(\bigotimes_{i=1}^n C_i) = \hat{C}$.

Decrypt($mpk, \hat{C}, \mathbf{v}_y$): On input the authorities' secret keys \mathbf{v}_y , an evaluated ciphertext F , compute \mathbf{v}_y^{-1} (where \mathbf{v}_y^{-1} is the vector consisting of inverse components of vector \mathbf{v}_y). Using this inverse \mathbf{v}_y^{-1} , compute:

$$\begin{aligned} & \log(\mathbf{v}_y^{-1} \otimes \dots \otimes \mathbf{v}_y^{-1}) + \log(C_1 \otimes \dots \otimes C_n) + \log(\mathbf{v}_y \otimes \dots \otimes \mathbf{v}_y) \\ & = \log[(\mathbf{v}_y^{-1} \otimes \dots \otimes \mathbf{v}_y^{-1})(C_1 \otimes \dots \otimes C_n)(\mathbf{v}_y \otimes \dots \otimes \mathbf{v}_y)]. \end{aligned}$$

It outputs a product of messages $\hat{\mu} = \prod_{i=1}^n \log(\mu_i) + \text{"small"}$.

Correctness. Since there is only one secret key v , the decryption process is given by multiplication with the secret key v and then by division of this product by v^{-1} . Correctness of decryption can be verified in the following computations assuming that the different ciphertexts can be decrypted using the same secret key. In the end we apply the exponential function to get the decrypted plaintext:

$$\begin{aligned} \log(\mathbf{v}_y^{-1} \otimes \dots \otimes \mathbf{v}_y^{-1}) + \log(C_1 \otimes \dots \otimes C_n) + \log(\mathbf{v}_y \otimes \dots \otimes \mathbf{v}_y) &= \log\left(\bigotimes_{i=1}^n \mathbf{v}_y^{-1} C_i \mathbf{v}_y\right) \\ &= \log\left(\prod_{i=1}^n (\mu_i + e_i)\right) \xrightarrow{\exp(\cdot)} \exp\left(\log\left(\prod_{i=1}^n (\mu_i + e_i)\right)\right) = \prod_{i=1}^n \mu_i + \text{“small”}. \end{aligned}$$

4.2 Security Analysis of LHABE

In this paragraph we define the security of our leveled homomorphic ABE scheme and provide the proof of security. We assume an adaptive adversary who specifies the set of strings x_i , $i \in [k]$ after receiving the public key. He is allowed to issue queries to the private key extraction oracle to string y of his choice, as long as $R(x_i, y) = 0$, where $x_i, i \in [k]$ are strings required for the encryption process, which have to be announced before the adversary obtains the public and secret keys of the authority.

Definition 4 (LHABE Indistinguishability). Let \mathcal{A}_{ind} be a probabilistic polynomial time adversary against the IND-CPA security of the leveled homomorphic ABE scheme, F an evaluation function and $b \in \{0, 1\}$ is a bit associated with the following experiment: $\mathbf{Exp}_{LHABE, \mathcal{A}_{ind}}^{IND-CPA-b}(1^\lambda)$:

1. $(apk, ask) \leftarrow \mathbf{Setup}(1^\lambda)$,
2. $F, st, (x_1^*, \mu_{1,0}, \mu_{1,1}), \dots, (x_n^*, \mu_{n,0}, \mu_{n,1}) \leftarrow \mathcal{A}_{ind}^{\mathcal{O}\mathbf{Extract}(\cdot)}(find, apk)$.
3. Compute $\{\mathbf{v}_y\}_{i \in [n-1]} \leftarrow \mathbf{Extract}(apk, ask, y)$.
4. Compute $C_{i,b}^* = \mathbf{Encrypt}(apk, x_i^*, \mu_{i,b})$, where $i \in [n]$ are different attributes and messages. We assume that each message is encrypted under another attribute.
5. $\hat{C}_b = \mathbf{Eval}(mpk, C_{1,b}^*, \dots, C_{n,b}^*, F)$.
6. $b' \leftarrow \mathcal{A}_{ind}^{\mathcal{O}\mathbf{Extract}(\cdot)}(\hat{C}_b, \{C_{i,b}^*\}_{i \in [n]}, st)$, where $b' \in \{0, 1\}$.

$\mathcal{O}\mathbf{Extract}(y)$: On input a string y , the oracle checks if $R(x_i^*, y) = 1$. If so, it returns \perp , otherwise runs $\mathbf{v}_y \xleftarrow{r} \mathbf{Extract}(apk, ask, y)$ and gives \mathbf{v}_y to \mathcal{A}_{ind} .

\mathcal{A}_{ind} wins if $b' = b$, meaning that \mathcal{A}_{ind} can distinguish whether \hat{C}_b was produced from $C_{1,0}, \dots, C_{n,0}$ or from $C_{1,1}, \dots, C_{n,1}$. The advantage of \mathcal{A}_{ind} is defined as:

$$\mathbf{Adv}_{LHABE, \mathcal{A}_{ind}}^{IND-CPA} = |Pr[\mathbf{Exp}_{LHABE, \mathcal{A}_{ind}}^{IND-CPA-0}(1^\lambda) = 1] - Pr[\mathbf{Exp}_{LHABE, \mathcal{A}_{ind}}^{IND-CPA-1}(1^\lambda) = 1]|.$$

The leveled homomorphic ABE (LHABE) scheme is IND-CPA secure if the above defined advantage $\mathbf{Adv}_{LHABE, \mathcal{A}_{ind}}^{IND-CPA}$ is negligible.

Remark 2. Furthermore, our LHABE scheme fulfills the compactness property which is justified analogously to the compactness property of LHMIBE scheme.

Theorem 2. Our leveled homomorphic ABE scheme is IND-CPA secure provided that (\mathbb{Z}_q, n, χ) -LWE holds.

4.3 Leveled Homomorphic Multi-authority ABE Scheme (LHMABE)

In this section we present the compilation of our leveled homomorphic multi-authority ABE scheme (LHMABE) from an LWE-based ABE scheme. We begin with the description of its syntax. Our scheme is associated to some efficient computable relation $R(x_i, y_j), x \in \{0, 1\}^l, y \in \{0, 1\}^{l'}$.

Definition 5 (LHMABE Scheme). *A leveled homomorphic multi-authority ABE scheme consists of the following five algorithms:*

Setup($1^\lambda, 1^n$): *On input a security parameter 1^λ output attribute public key apk_i and attribute secret key ask_i for each authority $j \in \{1, \dots, k\}$.*

KeyGen(apk_i, ask_i, y_i): *On input master public and master secret key pair, a string $y_i \in \{0, 1\}^{l'}$, the attribute authority i , generate a secret key sk_{y_i} which embeds the corresponding policy.*

Encrypt($\{apk_i\}_{i \in [k]}, \mu_\xi, x_i$): *On input a set of attributes represented by string x_i , a set of trusted authorities and their public keys, output a ciphertext C_i .*

Eval($apk, F, \{x_i\}_{i \in [k]}, C_1, \dots, C_n$): *On input a function F , set of strings $\{x_i\}_{i \in [k]}$ and a set of n ciphertexts C_1, \dots, C_n , homomorphically evaluate F and output \hat{C} .*

Decrypt($C, \{sk_{y_i}\}_{i \in [k]}$): *On input a ciphertext C , a set of secret keys $\{sk_{y_i}\}_{i \in [k]}$ decrypt the message if for every index i there is some index j s.t. $R(x_i, y_j) = 1$.*

The main idea of our leveled homomorphic MABE (LHMABE) scheme is a compilation from an already existing LWE-based ABE scheme and an extension to the multi-authority setting. There exist only few of such systems which have been realized and proved secure under the LWE assumption. Boyen [10] introduced a key policy attribute-based functional encryption which relies on the LWE problem. Gorbunov et al. [24] constructed an ABE scheme for circuits based on LWE. Gentry et al. [23] presented the first leveled homomorphic ABE scheme using compilation from ABE schemes [26, 37].

With introduction of multiple authorities the role of the key extraction algorithm in [10, 23, 24] is distributed among a multiple number of authorities where each of them computes a secret key for the user corresponding to a string $y_i \in \{0, 1\}^{l'}$. Our construction allows to encrypt different messages μ_i using different strings $x_i \in \{0, 1\}^l, i \in [1, n]$ and to evaluate them to a certain ciphertext. The user is able to decrypt the evaluated value only if for each x_i there exists some j such that $R(x_i, y_j) = 1$.

The Scheme. Let Σ' be a LWE-based attribute-based encryption scheme. We note that the encryption of different messages can be computed using different strings $x_i \in \{0, 1\}^l$, but it also includes the possibility to encrypt at least two different messages μ_i, μ_j under the same string x_i . A leveled homomorphic MABE scheme consists of the following algorithms:

Setup(1^λ): On input security parameter, generate authority's public key and authority's secret key apk_i, ask_i for each authority $i \in [k]$.

Extract(apk_i, ask_i, y_i): Run the **KeyGen** algorithm of Σ' scheme to compute $sk_i \in \mathbb{Z}_q^m$, which is the decryption key of that scheme embedding an access policy given by y_i into the key. Set $\mathbf{s}_i := sk_i = (1, sk_{i, \Sigma'}) \in \mathbb{Z}_q^{m+1}$. We note that $sk_{i, \Sigma'}$ is the decryption key of scheme Σ' . Compute the decryption key of ABE scheme as $\text{Powerof2}(\mathbf{s}_i) = \mathbf{v}_{y_i} \in \mathbb{Z}_q^{l \cdot (m+1)}$, for $i \in \{1, \dots, n\}$.

Encrypt($\{apk_i\}_{i \in [n]}, x_i, \mu_i$): On input authority's public key $\{apk_i\}$ for all $i \in [n]$ authorities, an attribute string y_i , for $i \in [n]$ and a message $\mu_i, i \in [n]$, run **Encrypt** of ABE scheme Σ' in order to compute $N = l \cdot (m + 1)$ encryptions of 0. The resulted ciphertext is denoted by C'_i . Taking C'_i compute: $C_i = \text{Flatten}(\mu_i \cdot I_N + \text{BitDecomp}(C'_i))$.

Eval($\{apk_i\}_{i \in [n]}, \{x_i\}_{i \in [n]}, C_1, \dots, C_n, F$): On input the ciphertexts, C_1, \dots, C_n on messages μ_1, \dots, μ_n and an evaluation function $F(C_1, \dots, C_n)$, compute: $F(C_1, \dots, C_n) = \log(\prod(C_1 \otimes \dots \otimes C_n))$.

Decrypt($mpk, \hat{C}, \{\mathbf{v}_{y_j}\}_{j \in [n]}$): On input master public key mpk , evaluated ciphertext \hat{C} and n secret keys $\mathbf{v}_{y_1}, \dots, \mathbf{v}_{y_n}$, compute

$$\log[\mathbf{v}_{y_1}^{-1} \otimes \dots \otimes \mathbf{v}_{y_n}^{-1}] + \log\left[\bigotimes_{i=1}^n C_i\right] + \log\left[\left(\bigotimes_{i=1}^n \mathbf{v}_{y_i}\right)\right]$$

and output $\hat{\mu}$, where $\hat{\mu} = \log\left[\prod_{i=1}^n \mu_i + \text{"small"}\right]$.

Correctness. Correctness of decryption can be verified in the following computations, assuming that the different ciphertexts can be decrypted using different secret key. In the end we apply the exponential function to get the decrypted plaintext:

$$\begin{aligned} & \log(\mathbf{v}_{y_1}^{-1} \otimes \dots \otimes \mathbf{v}_{y_n}^{-1}) + \log(C_1 \otimes \dots \otimes C_n) + \log(\mathbf{v}_{y_1} \otimes \dots \otimes \mathbf{v}_{y_n}) \\ &= \log\left(\bigotimes_{i=1}^n \mathbf{v}_{y_i}^{-1} C_i \mathbf{v}_{y_i}\right) = \log\left(\prod_{i=1}^n (\mu_i + e_i)\right) \xrightarrow{\exp(\cdot)} \exp\left(\log\left(\prod_{i=1}^n (\mu_i + e_i)\right)\right). \end{aligned}$$

4.4 Security Analysis of LHMABE

In the following definition we define the indistinguishability property of our scheme. We assume an adaptive adversary \mathcal{A}_{ind} who outputs a set of target strings y_i^* for $i \in [n]$ after receiving the public key. Further we give \mathcal{A}_{ind} access to a key extraction oracle on input a string x_i with the restriction that there is no $y_j^*, j \in [n]$, such that $R(y_j^*, x_i) = 1$. A successful decryption is only possible if the user has all of the n decryption keys corresponding to the strings $\{x_i\}_{i \in [n]}$ which were used during the encryption of n different messages. Without loss of generality, there can be messages which were encrypted under the same string y_i^* .

Definition 6 (LHMABE Indistinguishability). Let \mathcal{A}_{ind} be a probabilistic polynomial time adversary against the IND-CPA security of the leveled homomorphic MABE scheme, F an evaluation function and $b \in \{0, 1\}$ a bit associated with the following experiment: $\mathbf{Exp}_{LHMABE, \mathcal{A}_{ind}}^{IND-CPA-b}(1^\lambda)$:

1. $(apk_i, ask_i) \leftarrow \text{Setup}(1^\lambda)$,
2. $F, st, (y_1^*, \mu_{1,0}, \mu_{1,1}), \dots, (y_n^*, \mu_{n,0}, \mu_{n,1}) \leftarrow \mathcal{A}_{ind}^{\text{Extract}(\cdot)}(find, apk_i)$.
3. Compute $\{v_{y_i}\}_{i \in [n-1]} \leftarrow \text{Extract}(apk_i, ask_i, y_i)$ and set $S = \{(y_i, v_{y_i})\}_{i \in [n]}$. At the beginning of the experiment the set S is empty.
4. If $(y_i^*, \cdot) \notin S$, run $v_{y_i^*} \leftarrow \text{Extract}(apk_i, ask_i, y_i^*)$, s.t. $R(y_i^*, x_i) = 1$, add $(y_i^*, v_{y_i^*})$ to S .
5. Compute $C_{i,b}^* = \text{Encrypt}(apk_i, x_i^*, \mu_{i,b})$, where $i \in [n]$ is the index of different identities.
6. $\hat{C}_b = \text{Eval}(\{apk_i\}_{i \in [n]}, C_{1,b}^*, \dots, C_{n,b}^*, F)$.
7. $b' \leftarrow \mathcal{A}_{ind}^{\text{Extract}(\cdot)}(\hat{C}_b, \{C_{i,b}^*\}_{i \in [n]}, st)$, where $b' \in \{0, 1\}$.

$\mathcal{O}\text{Extract}(ask_i, y_i)$: On input (ask_i, y_i) , the oracle checks if there is an y_i^* in the announced set of strings, such that $R(y_j^*, x_i) = 1$. If so, returns \perp . Otherwise it runs $v_{y_i} \xleftarrow{r} \text{Extract}(apk_i, ask_i, y_i)$ and gives v_{y_i} to \mathcal{A}_{ind} . If $|L| > n - 1$, the oracle returns \perp .

\mathcal{A}_{ind} wins if $b' = b$, meaning that \mathcal{A}_{ind} can distinguish whether \hat{C}_b was produced from $C_{1,0}, \dots, C_{n,0}$ or from $C_{1,1}, \dots, C_{n,1}$ and \mathcal{A}_{ind} did not issue queries on x_i with $R(y_j^*, x_i) = 1$ for some $y_j^* \in \{0, 1\}^l$. \mathcal{A}_{ind} 's advantage is:

$$\mathbf{Adv}_{LHMABE, \mathcal{A}_{ind}}^{IND-CPA} = |\Pr[\mathbf{Exp}_{LHMABE, \mathcal{A}_{ind}}^{IND-CPA-0}(1^\lambda) = 1] - \Pr[\mathbf{Exp}_{LHMABE, \mathcal{A}_{ind}}^{IND-CPA-1}(1^\lambda) = 1]|.$$

The LHMABE scheme is IND-CPA secure if $\mathbf{Adv}_{LHMABE, \mathcal{A}_{ind}}$ is negligible.

Remark 3. Furthermore, our LHMABE scheme fulfills the compactness property which is justified analogously to the compactness property of LHMIBE scheme.

Theorem 3. Our leveled homomorphic MABE scheme is IND-CPA secure provided that (\mathbb{Z}_q, n, χ) -LWE holds.

Proof. Let \mathcal{A}_{ind} be an adversary against IND-CPA security of our leveled homomorphic multi-authority ABE scheme. We use \mathcal{A}_{ind} to construct an algorithm \mathcal{B} against the LWE problem.

Setup: The instance of LWE problem is given as a sampling oracle \mathcal{O} . This oracle can be either purely random \mathcal{O}_r or pseudo-random \mathcal{O}_s for some secret $s \in \mathbb{Z}_q^N$, where $N = l(m+1)$ as in the scheme. In order to simulate \mathcal{A}'_{ind} 's public parameters, \mathcal{B} issues N queries on samples to his sampling oracle \mathcal{O} and receives upon each request i a fresh pair $(\mathbf{a}_i, t_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$. The simulator \mathcal{B} computes for \mathcal{A}_{ind} the public parameters (apk_i, ask_i) for each attribute authority using LWE samples and sends them to \mathcal{A}_{ind} .

Key Extract Queries: When \mathcal{A}_{ind} issues private key extraction queries to its key extract oracle on input (ask_i, y_i) , $i \in [n]$, simulator \mathcal{B} computes the required

secret keys sk_i using the samplings obtained from its oracle and the simulated public key. We assume that \mathcal{B} has control over the set of strings $\{y_i^*\}_{i \in [n]}$. If there is an $y_j \in \{y_i^*\}_{i \in [n]}$ such that $R(y_j, x_i) = 1$, \mathcal{B} aborts the simulation. Otherwise it returns the simulated secret keys to \mathcal{A}_{ind} . The outputs are statistically close to uniform values. \mathcal{B} sends the simulated values to \mathcal{A}_{ind} . We assume that \mathcal{A}_{ind} issued in total q_E private key extraction queries.

Furthermore \mathcal{A}_{ind} outputs a set of target strings $\{y_i^*\}_{i \in [n]}$ with the corresponding messages μ_1, \dots, μ_n it wants to be challenged on.

Challenge ciphertext: The simulation of the challenge ciphertext also works using as input the entries from the LWE instance, choosing N random vector pairs $\mathbf{b}_i, \mathbf{e}_i \xleftarrow{r} \mathbb{Z}_q^N$ for $i \in [n]$, taking the message bits μ_i and calculating $C \leftarrow \mathbf{b}_i \cdot \mu_i + \mathbf{e}_i$. Finally the ciphertext is sent to \mathcal{B} . When the LWE oracle is given by \mathcal{O}_s (i.e. it is pseudo-random), the ciphertext is randomly distributed including some random noise vector according to the noisy distribution χ . When \mathcal{O} is given by \mathcal{O}_r then the ciphertext is uniform and independent over \mathbb{Z}_q^N . Eventually the simulated ciphertext is always uniform in $\mathbb{Z}_q \times \mathbb{Z}_q^N$.

Guess: After issuing additional queries, \mathcal{A}_{ind} guesses a bit $b' \in \{0, 1\}$. The LWE adversary \mathcal{B} outputs its guess as result of the LWE challenge. Finally we follow that \mathcal{B} 's advantage in solving LWE is at least the same as \mathcal{A}'_{ind} 's advantage in distinguishing the ciphertext from a random value, i.e.: $\mathbf{Adv}_{\mathcal{B}} \geq \frac{1}{q_E} \mathbf{Adv}_{\mathcal{A}_{ind}}$. \square

4.5 Application to Cloud Computing

Leveled homomorphic attribute-based encryption has significant relevance for cloud systems and their security. Attribute-based encryption allows an additional option for many applications of functional encryption in cloud computing. It enables a data owner, who outsourced her encrypted data to a cloud, to control the access to the uploaded data. A useful application to personal health records in cloud computing based on multi-authorities and multi-users attribute-based encryption presented by Li et al. [29] can profit by enabling users of the scheme to evaluate different ciphertexts on different messages without even revealing those messages. The shortcoming of Li et al.'s [29] construction is the impossibility to perform complex mathematical computations on encrypted data. Our ABE construction in multi-authority setting in Sect. 4.3 provides this attractive property. The data owner which uploads the data to a cloud server has the possibility to encrypt further several data files and compute a functional value of the resulted ciphertexts. The data user, which has the valid access formula is able to decrypt the evaluated ciphertexts and obtain a functional value of the plaintexts. Our construction allows the cloud users to take advantage of analytical cloud services. Our scheme from Sect. 3.1, where distinct ciphertexts are encrypted using distinct identities can also be applied to the cloud storage setting. In this case our construction allows a data owner to encrypt different data for a certain group of users with distinct identities, such that each user is able to decrypt an evaluated value of different plaintexts.

5 Conclusion

In this paper we presented a new construct called tensor product in combination with natural logarithm in order to enable leveled homomorphic encryption under different public keys. Using these mathematical constructions we first introduced a leveled homomorphic encryption under multiple identities. We defined the security of that scheme and provided the corresponding proof. Furthermore we presented a leveled homomorphic attribute-based encryption in two different settings. In the first setting we assumed that the evaluation function operates on ciphertext under common index x , while in the second setting the evaluation function was performed under distinct indices. We defined the security notions for both ABE schemes and proved them secure. Our constructions enable a multi-key leveled homomorphic encryption on lattices using simple mathematical computations in contrast to so far existing multi-identity FHE by [18] and provide efficient applications to the cloud storage setting.

A Lattices

Let $B = \{b_1, \dots, b_n\} \subset \mathbb{R}^n$ be a basis of a lattice Λ which consists of n linearly independent vectors. The n -dimensional lattice Λ is then defined as $\Lambda = \sum_{i=1}^n \mathbb{Z}b_i$.

The i -th minimum of a lattice Λ , denoted by $\lambda_i(\Lambda)$ is the smallest radius r such that Λ contains i linearly independent vectors of norms $\leq r$. (The norm

of vector b_i is defined as $\|b_i\| = \sqrt{\sum_{j=1}^n c_{i,j}^2}$, where $c_{i,j}, j \in \{1, \dots, n\}$ are the

coefficients of vector b_i . We denote by $\lambda_1^\infty(\Lambda)$ the minimum distance measured in the infinity norm, which is defined as $\|b_i\|_\infty := \max(|c_{i,1}|, \dots, |c_{i,n}|)$. Additionally we recall $\|B\| = \max \|b_i\|$ and its fundamental parallelepiped is given

by $P(B) = \left\{ \sum_{i=1}^n a_i b_i \mid \mathbf{a} \in [0, 1)^n \right\}$. The integer n is called the rank of the basis.

Note that a lattice basis is not unique, since for any unimodular matrix $U \in \mathbb{Z}^{n \times n}$ the product $B \cdot U$ is also a basis of Λ .

Integer Lattices. The following specific lattices contain $q\mathbb{Z}^m$ as a sub-lattice for a prime q . For $A \in \mathbb{Z}^{n \times m}$ and $s \in \mathbb{Z}_q^n$, define:

$$\begin{aligned} \Lambda_q(A) &:= \{e \in \mathbb{Z}^m \mid \exists s \in \mathbb{Z}_q^n, \text{ where } A^T s = e \pmod{q}\}, \\ \Lambda_q^\perp(A) &:= \{e \in \mathbb{Z}^m \mid Ae = 0 \pmod{q}\}, \end{aligned}$$

Many lattice-based works rely on Gaussian-like distributions called Discrete Gaussians. In the following paragraph we recall the main notations of this distribution.

Discrete Gaussians. Let L be a subset of \mathbb{Z}^m . For a vector $c \in \mathbb{R}^m$ and a positive $\sigma \in \mathbb{R}$, define

$$\rho_{\sigma,c}(x) = \exp\left(-\pi \frac{\|x - c\|^2}{\sigma^2}\right) \quad \text{and} \quad \rho_{\sigma,c}(L) = \sum_{x \in L} \rho_{\sigma,c}(x).$$

The discrete Gaussian distribution over L with center c and parameter σ is given by $\mathcal{D}_{L,\sigma,c}(y) = \frac{\rho_{\sigma,c}(y)}{\rho_{\sigma,c}(L)}$, $\forall y \in L$. The distribution $\mathcal{D}_{L,\sigma,c}$ is usually defined over the lattice $L = \Lambda_q^\perp(A)$ for $A \in \mathbb{Z}_q^{n \times m}$.

B Learning With Errors (LWE)

The LWE problem, first introduced by Regev [36], relies on the Gaussian error distribution χ , which is given as $\chi = D_{\mathbb{Z},s}$ over the integers. The LWE problem assumes of access to a challenge oracle \mathcal{O} , which is either a purely random sampler \mathcal{O}_r or a noisy pseudo-random sampler \mathcal{O}_s , with some random secret key $s \in \mathbb{Z}_q^n$. For positive integers n and $q \geq 2$, a vector $\mathbf{s} \in \mathbb{Z}_q^n$ and error term $e \leftarrow \chi$, the LWE distribution $A_{s,\chi}$ is sampled over $\mathbb{Z}_q^n \times \mathbb{Z}_q$. Chosen a vector $\mathbf{a} \in \mathbb{Z}_q^n$ uniformly at random it outputs the pair $(\mathbf{a}, t = \langle \mathbf{a}, \mathbf{s} \rangle + e \pmod{q}) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$. A more detailed description of χ can be found in [36]. The sampling oracles work in the following way:

\mathcal{O}_s : outputs samples of the form $(\mathbf{a}, t) = (\mathbf{a}, \mathbf{a}\mathbf{s} + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, where $\mathbf{s} \in \mathbb{Z}_q^n$ is uniformly distributed value across all invocations and $e \in \mathbb{Z}_q$ is a fresh sample from χ .

\mathcal{O}_r : outputs truly random samples from $\mathbb{Z}_q^n \times \mathbb{Z}_q$.

C Proof of Theorem 2

Proof. Since the security of this construction relies on the hardness of LWE problem we show how to build an algorithm which can simulate the outputs for the LHABE adversary. Let \mathcal{A}_{ind} be an adversary against IND-CPA security of our leveled homomorphic ABE scheme. We use \mathcal{A}_{ind} to construct an algorithm \mathcal{B} against the LWE problem. As known from the Definition of LWE, the decision algorithm has access to a sampling oracle \mathcal{O} , which can be either a pseudorandom sampler \mathcal{O}_s or a truly random sampler \mathcal{O}_r . We assume a simulator \mathcal{B} which simulates the environment for LHABE adversary \mathcal{A}_{ind} in order to decide which oracle is given. \mathcal{B} queries from its oracle \mathcal{O} the LWE samples and obtains n pairs $(\mathbf{a}_i, t_i) \in \mathbb{Z}_q^N \times \mathbb{Z}_q$, for $N = l(m+1)$. \mathcal{A}_{ind} announces a set of strings $\{x_i\}_{i \in k}$ it wants to be challenged on. The simulator \mathcal{B} constructs the public key using the obtained LWE instance of l pairs (\mathbf{a}_i, t_i) for $i \in [l(m+1)]$, where the public key is represented by a $n \times m$ matrix and a m -dimensional vector. When \mathcal{A} issues key generation queries on input apk , the LWE adversary simulates the queries using previously sampled public key apk and setting $\mathbf{s} = (1, s_1) \in \mathbb{Z}_q^{l(m+1)}$, where

$apk \cdot \mathbf{s} = \mathbf{e}$ that is small and $s_1 \in \mathbb{Z}_q^{lm}$ is also assumed to be small according to distribution χ . In order to encrypt 0, \mathcal{B} samples N times the vectors $\mathbf{b}, \mathbf{e}' \xleftarrow{r} \mathbb{Z}_q^N$ according to χ and outputs a ciphertext $C \leftarrow \mathbf{b} \cdot apk + \mathbf{e}'$. This ciphertext is indistinguishable from random by applying a standard hybrid argument. The decryption is possible by computing a product of $\langle C, \mathbf{s} \rangle$ and outputting $\mu = 0$ if the result is small or $\mu = 1$ otherwise. \square

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_28
2. Agrawal, S., Boyen, X.: Identity-based encryption from lattices in the standard model. <http://www.cs.stanford.edu/~xb/ab09/>
3. Agrawal, S., Boyen, X., Vaikuntanathan, V., Voulgaris, P., Wee, H.: Functional encryption for threshold functions (or fuzzy IBE) from lattices. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 280–297. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30057-8_17
4. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: Proceedings of 28th Annual ACM Symposium on the Theory of Computing, pp. 99–108. ACM (1996)
5. Attrapadung, N., Herranz, J., Laguillaumie, F., Libert, B., de Panafieu, E., Ràfols, C.: Attribute-based encryption schemes with constant-size ciphertexts. *Theoret. Comput. Sci.* **422**, 15–38 (2012)
6. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: 2007 IEEE Symposium on Security and Privacy (S&P 2007), pp. 321–334. IEEE Computer Society (2007)
7. Blum, A., Kalai, A., Wasserman, H.: Noise-tolerant learning, the parity problem, and the statistical query model. In: Proceedings of 32nd Annual ACM Symposium on Theory of Computing, pp. 435–440 (2000)
8. 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). https://doi.org/10.1007/3-540-44647-8_13
9. Boyen, X.: Lattice mixing and vanishing trapdoors: a framework for fully secure short signatures and more. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 499–517. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13013-7_29
10. Boyen, X.: Attribute-based functional encryption on lattices. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 122–142. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36594-2_8
11. Brakerski, Z., Cash, D., Tsabary, R., Wee, H.: Targeted homomorphic attribute-based encryption. In: Hirt, M., Smith, A. (eds.) TCC 2016. LNCS, vol. 9986, pp. 330–360. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53644-5_13
12. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: Fully homomorphic encryption without bootstrapping. In: Electronic Colloquium on Computational Complexity (ECCC), vol. 18, p. 111 (2011)

13. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS, 2011, pp. 97–106. IEEE Computer Society (2011)
14. Brakerski, Z., Vaikuntanathan, V.: Fully homomorphic encryption from ring-LWE and security for key dependent messages. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 505–524. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_29
15. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_27
16. Chase, M.: Multi-authority attribute based encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70936-7_28
17. Cheung, L., Newport, C.C.: Provably secure ciphertext policy ABE. In: Proceedings of 2007 ACM Conference on Computer and Communications Security, CCS 2007, pp. 456–465. ACM (2007)
18. Clear, M., McGoldrick, C.: Multi-identity and multi-key leveled FHE from learning with errors. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9216, pp. 630–656. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_31
19. Clear, M., McGoldrick, C.: Attribute-based fully homomorphic encryption with a bounded number of inputs. In: Pointcheval, D., Nitaj, A., Rachidi, T. (eds.) AFRICACRYPT 2016. LNCS, vol. 9646, pp. 307–324. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-31517-1_16
20. Cocks, C.: An identity based encryption scheme based on quadratic residues. In: Honary, B. (ed.) Cryptography and Coding 2001. LNCS, vol. 2260, pp. 360–363. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45325-3_32
21. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of 41st Annual ACM Symposium on Theory of Computing, STOC 2009, pp. 169–178. ACM (2009)
22. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of 40th Annual ACM Symposium on Theory of Computing, STOC 2008, pp. 197–206. ACM (2008)
23. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 75–92. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_5
24. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: Symposium on Theory of Computing Conference, STOC 2013, pp. 545–554. ACM (2013)
25. Goyal, V., Jain, A., Pandey, O., Sahai, A.: Bounded ciphertext policy attribute based encryption. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfssdóttir, A., Walukiewicz, I. (eds.) ICALP 2008. LNCS, vol. 5126, pp. 579–591. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-70583-3_47
26. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data, pp. 89–98 (2006)
27. Kamara, S., Mohassel, P., Raykova, M.: Outsourcing multi-party computation. IACR Cryptology ePrint Archive, 2011:272 (2011)

28. 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). https://doi.org/10.1007/978-3-642-13190-5_4
29. Li, M., Yu, S., Ren, K., Lou, W.: Securing personal health records in cloud computing: patient-centric and fine-grained data access control in multi-owner settings. In: Jajodia, S., Zhou, J. (eds.) SecureComm 2010. LNICST, vol. 50, pp. 89–106. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16161-2_6
30. López-Alt, A., Tromer, E., Vaikuntanathan, V.: On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Proceedings of 44th Symposium on Theory of Computing Conference, STOC 2012, pp. 1219–1234 (2012)
31. Micciancio, D.: Generalized compact knapsacks, cyclic lattices, and efficient one-way functions from worst case complexity assumptions. In: FOCS 2002, pp. 356–365 (2002)
32. Micciancio, D., Voulgaris, P.: A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. *SIAM J. Comput.* **42**(3), 1364–1391 (2013)
33. Mukherjee, P., Wichs, D.: Two round multiparty computation via multi-key FHE. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9666, pp. 735–763. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_26
34. Peikert, C.: Bonsai trees (or, arboriculture in lattice-based cryptography). *IACR Cryptology ePrint Archive*, 2009:359 (2009)
35. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem. In: Proceedings of 41st Annual ACM Symposium on Theory of Computing, STOC 2009, pp. 333–342 (2009)
36. Regev, O.: On lattices, learning with errors, random linear codes and cryptography. In: Proceedings of 37th Annual ACM Symposium on Theory of Computing, STOC 2005, pp. 84–93 (2005)
37. Sahai, A., Waters, B.: Fuzzy identity based encryption. *IACR Cryptology ePrint Archive*, 2004:86 (2004)
38. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985). https://doi.org/10.1007/3-540-39568-7_5
39. Smart, N.P., Vercauteren, F.: Fully homomorphic encryption with relatively small key and ciphertext sizes. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 420–443. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13013-7_25
40. 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). https://doi.org/10.1007/978-3-642-13190-5_2
41. 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). https://doi.org/10.1007/978-3-642-19379-8_4



Improved Key Generation Algorithm for Gentry's Fully Homomorphic Encryption Scheme

Yang Zhang^{1,2} , Renzhang Liu¹ , and Dongdai Lin¹

¹ State Key Laboratory of Information Security,
Institute of Information Engineering, Chinese Academy of Sciences,
Beijing 100093, China

liurenzhang@iie.ac.cn

² School of Cyber Security, University of Chinese Academy of Sciences,
Beijing 100049, China

Abstract. At EUROCRYPT 2011, Gentry and Halevi implemented a variant of Gentry's fully homomorphic encryption scheme. The core part in their key generation is to generate an odd-determinant ideal lattice having a particular type of Hermite Normal Form. However, they did not give a rigorous proof for the correctness. We present a better key generation algorithm, improving their algorithm from two aspects.

- We show how to deterministically generate ideal lattices with odd determinant, thus increasing the success probability close to 1.
- We give a rigorous proof for the correctness. To be more specific, we present a simpler condition for checking whether the ideal lattice has the desired Hermite Normal Form. Furthermore, our condition can be checked more efficiently.

As a result, our key generation is about 1.5 times faster. We also give experimental results supporting our claims. Our optimizations are based on the properties of ideal lattices, which might be of independent interests.

Keywords: Fully homomorphic encryption · Key generation
Hermite Normal Form · Ideal lattice

1 Introduction

Fully homomorphic encryptions (FHE) support arbitrary operations on encrypted data which have a wide range of applications such as private information retrieval [19], electronic watermark [11]. Ever since the introduction of the concept of FHE by Rivest et al. [13], there have appeared many homomorphic encryption schemes which support limited operations [1, 14]. Nevertheless, all these schemes failed to be FHE.

It was not until 2009 that Gentry [6] proposed the first plausible FHE scheme using ideal lattice. Since then, researchers proposed many FHE schemes [2, 3, 9].

Nowadays, all existing FHE schemes follow Gentry’s blueprint. Specifically, one first constructs a “somewhat homomorphic” scheme that supports some limited operations on the encrypted data. Then he “squashes” the decryption procedure so that it can be expressed as operations supported by the scheme. Finally, he converts the “somewhat homomorphic” scheme into a fully homomorphic scheme. The low efficiency of FHE is one of the main obstacles that prevents it from being practical.

The first attempt to implement Gentry’s FHE scheme was made by Smart and Vercauteren [18] in 2010. They gave an optimized version of Gentry’s scheme, which decreased the key size by a linear factor. However, they needed to generate ideal lattices with prime determinant during the key generation. In practice, one may need to try as many as $n^{1.5}$ candidates before finding one with prime determinant when working with lattices in dimension n . As a result, they failed to implement Gentry’s fully homomorphic encryption scheme.

In 2011, Gentry and Halevi [7] first completely implemented Gentry’s fully homomorphic encryption scheme continuing in the same direction of the implementation of [18]. The strong requirement that the lattice has a prime determinant was removed. Instead, they only required that the determinant is odd and the lattice has a simple-HNF (see Definition 2). They found in practice that the success probability was roughly 0.5. Thus one needs to try two candidates on average to get a valid key. Moreover, they proposed a method to check whether a lattice has a simple-HNF. However, they did not provide a rigorous proof for the correctness of their algorithm and some details on their implementation were not very clear.

Gentry-Halevi’s implementation depends heavily on the underlying algebraic structure of the scheme, that is, the 2-power cyclotomic fields. Scholl and Smart [16] showed how to generalize the fast key generation techniques to arbitrary cyclotomic fields. They also obtained a key generation algorithm which is roughly twice faster.

Our Contribution. By studying the properties of ideal lattices, we present further improvements on the key generation algorithm in [7]. Our algorithm has a rigorous proof for correctness and is about 1.5 times faster in practice. Our focus is on Gentry-Halevi’s original implementation, which is different from [16].

As stated before, the core idea in the key generation algorithm of [7] is to generate an odd-determinant ideal lattice, which has a simple-HNF.

In order to generate ideal lattice with odd determinant, the authors of [7] chose a random generator and computed the determinant of the generated ideal lattice. Since the ideal lattice under consideration is highly structured and we only care about the parity of the determinant, it might be possible to determine whether the determinant is odd without computing it. In fact, we do find that the parity of the determinant is connected to the generator in a very simple way. Thus we can generate odd-determinant ideal lattice deterministically by imposing a simple constraint on the generator.

To check whether the ideal lattice has the desired HNF, they gave a simple checking condition, which avoided computing the HNF explicitly out of the consideration of efficiency. By studying and exploiting the properties of the HNF of ideal lattice, we are able to present another checking condition. By verifying that our condition holds, we can rigorously prove that our algorithm outputs an ideal lattice with the desired HNF. Furthermore, our condition can be checked more efficiently.

We need to point out that our improvements are of more theoretical than practical significance due to the following reasons. First, Gentry and Halevi’s implementation is the very first one, and there are much better implementations of different schemes [5, 8, 15, 17]. Besides, key generation is not a bottleneck of FHE schemes and the algorithm in [7] is very fast even for very large parameters, which makes our slight speedup less significant.

Nevertheless, it is important to make things more clear. Besides, our improvements are based on some new, special properties of the HNF of ideal lattices, which we believe are of independent interests.

Roadmap. The paper is organized as follows. In Sect. 2, some preliminaries are presented. In Sect. 3, we revisit the key generation algorithms in previous implementations. In Sect. 4, we propose our optimized key generation for Gentry’s FHE scheme, together with some theoretical analysis and experimental results. A brief conclusion will be given in the final section.

2 Preliminaries

Notations. We use bold capital and lowercase letters to denote matrices and vectors respectively. For a matrix $\mathbf{M} \in \mathbb{R}^{n \times m}$, the i -th row is denoted as \mathbf{M}_i and the entry in (i, j) -th entry is denoted as $m_{i,j}$. For a vector $\mathbf{x} \in \mathbb{R}^n$, the i -th entry is denoted as x_i . These notations are used throughout the paper unless specified otherwise.

2.1 Hermite Normal Form

For integer matrices, there is a very important standard form known as the Hermite Normal Form.

Definition 1 (Hermite Normal Form). *A nonsingular matrix $\mathbf{H} \in \mathbb{Z}^{n \times n}$ is said to be in HNF, if*

- $h_{i,i} > 0$ for $1 \leq i \leq n$.
- $h_{j,i} = 0$ for $1 \leq j < i \leq n$.
- $0 \leq h_{j,i} < h_{i,i}$ for $1 \leq i < j \leq n$.

For “randomly” generated matrices of high dimensions, the diagonals of its HNF are highly unbalanced: most of them are small (in fact most of them are 1), and the first diagonal is usually the only large one. We can define a very special HNF.

Definition 2 (Simple-HNF). A nonsingular matrix $\mathbf{H} \in \mathbb{Z}^{n \times n}$ is said to be in simple-HNF if it is in HNF and $h_{i,i} = 1$ for $2 \leq i \leq n$.

It has been shown that asymptotically the density of “simple-HNF” for randomly generated $n \times n$ matrices (in the sense that each entry is chosen uniformly at random from $\{-M, -M + 1, \dots, M\}$ for large enough M) is $\frac{1}{\prod_{j=2}^n \zeta(j)} \approx 44\%$ [10, 12], where $\zeta(j)$ is the Riemann zeta function.

2.2 Lattice

A lattice is a discrete subgroup of \mathbb{R}^m . Formally,

Definition 3 (Lattice). Given n linearly independent vectors $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n\}$, where $\mathbf{b}_i \in \mathbb{R}^m$, the lattice $\mathcal{L}(\mathbf{B})$ generated by \mathbf{B} is defined as following

$$\mathcal{L}(\mathbf{B}) = \left\{ \sum_{i=1}^n x_i \mathbf{b}_i : x_i \in \mathbb{Z} \right\} = \{ \mathbf{x} \mathbf{B} : \mathbf{x} \in \mathbb{Z}^n \}.$$

We call \mathbf{B} a basis of $\mathcal{L}(\mathbf{B})$, m and n the dimension and the rank of $\mathcal{L}(\mathbf{B})$ respectively. When $m = n$, we say $\mathcal{L}(\mathbf{B})$ is full-rank.

Definition 4 (Determinant). For lattice $\mathcal{L}(\mathbf{B})$, the determinant is defined as

$$\det(\mathcal{L}) = \sqrt{\det(\mathbf{B} \mathbf{B}^T)}.$$

When $\mathcal{L}(\mathbf{B})$ is full-rank, \mathbf{B} is nonsingular and $\det(\mathcal{L}) = |\det(\mathbf{B})|$.

Definition 5 (Primitive Lattice Vector). A lattice vector $\mathbf{v} \in \mathcal{L}$ is called a primitive lattice vector if for any integer $k > 1$, $\mathbf{v}/k \notin \mathcal{L}$.

There is an easy criterion for determining whether a lattice vector is primitive.

Proposition 1. Given a lattice \mathcal{L} with basis \mathbf{B} , $\mathbf{v} = \mathbf{x} \mathbf{B} \in \mathcal{L}$ is primitive if and only if $\gcd(x_1, \dots, x_n) = 1$.

2.3 Ideal Lattice

In what follows, we focus on $R = \mathbb{Z}[x]/\langle f(x) \rangle$, where $f(x) \in \mathbb{Z}[x]$ is a monic polynomial of degree n and $\langle f(x) \rangle$ is the ideal generated by $f(x)$ in $\mathbb{Z}[x]$.

Consider the coefficient embedding

$$\begin{aligned} \sigma : R &\rightarrow \mathbb{Z}^n, \\ \sum_{i=0}^{n-1} a_i x^i &\mapsto (a_0, a_1, \dots, a_{n-1}). \end{aligned}$$

For any polynomial $v(x) \in R$, we use \mathbf{v} to denote the image of $v(x)$ under σ , and consider them equivalent.

Let $g(x)$ be a polynomial of degree $< n$ and $v(x) \in \langle g(x) \rangle$, then there exists a polynomial $w(x) \in \mathbb{Z}[x]$ of degree $< n$ such that $v(x) = w(x)g(x) \pmod{f(x)}$. Write $w(x) = w_{n-1}x^{n-1} + \dots + w_1x + w_0$, then $v(x) = \sum_{i=0}^{n-1} w_i x^i g(x) \pmod{f(x)}$. So each element in $\langle g(x) \rangle$ is a linear combination of $g(x) \pmod{f(x)}$, $xg(x) \pmod{f(x)}$, \dots , $x^{n-1}g(x) \pmod{f(x)}$. It follows that $\langle g(x) \rangle$ is a lattice under the coefficient embedding. Thus, we can define

Definition 6 (Ideal Lattice). For $g(x) \in R = \mathbb{Z}[x]/\langle f(x) \rangle$, where $f(x) \in \mathbb{Z}[x]$ is a monic polynomial of degree n , the ideal generated by $g(x)$ forms a lattice \mathcal{L} under the coefficient embedding. We call \mathcal{L} the ideal lattice generated by $g(x)$.

Moreover, if $g(x)$ and $f(x)$ are coprime over \mathbb{Q} (hence over \mathbb{Z} since $f(x)$ is monic), $g(x) \pmod{f(x)}$, $xg(x) \pmod{f(x)}$, \dots , $x^{n-1}g(x) \pmod{f(x)}$ are linearly independent. Otherwise there exist integers y_0, y_1, \dots, y_{n-1} not all zero, such that $\sum_{i=0}^{n-1} y_i x^i g(x) \pmod{f(x)} = 0$, that is, $(\sum_{i=0}^{n-1} y_i x^i)g(x) = 0 \pmod{f(x)}$, which indicates that $g(x)$ and $f(x)$ are not coprime. Therefore, the ideal lattice generated by $g(x)$ is full-rank.

2.4 Resultant

The resultant of two polynomials is defined as

Definition 7 (Resultant). Let $a(x) = a_m x^m + \dots + a_1 x + a_0$, $b(x) = b_n x^n + \dots + b_1 x + b_0 \in \mathbb{R}[x]$. Define the Sylvester matrix of $a(x)$ and $b(x)$ as

$$Sylv(a(x), b(x)) = \begin{bmatrix} a(x) \\ \vdots \\ x^{n-1}a(x) \\ b(x) \\ \vdots \\ x^{m-1}b(x) \end{bmatrix} = \begin{bmatrix} a_0 & a_1 & \dots & a_m & & & \\ & a_0 & a_1 & \dots & a_m & & \\ & & \ddots & & & & \\ & & & a_0 & a_1 & \dots & a_m \\ b_0 & \dots & & & b_n & & \\ & \ddots & & & & \ddots & \\ & & b_0 & \dots & & & b_n \end{bmatrix}_{(n+m) \times (n+m)}$$

Then the resultant of $a(x)$ and $b(x)$, denoted as $Res(a, b)$, is the determinant of $Sylv(a(x), b(x))$.

3 Key Generation Algorithms of Gentry’s FHE Scheme

In this section we revisit the key generation algorithms implementing Gentry’s fully homomorphic encryption scheme in [7, 18].

3.1 The Smart-Vercauteren’s Key Generation

Smart and Vercauteren [18] first attempted to implement Gentry’s FHE schemes. For the key generation, they fixed a monic irreducible polynomial $f(x) \in \mathbb{Z}[x]$ of

degree n . Then they repeatedly chose another polynomial $g(x)$ randomly from some set until the resultant of $f(x)$ and $g(x)$ was prime. They showed that the HNF of such ideal lattice had the following simple-HNF.

$$\begin{bmatrix} d & 0 & 0 & 0 \\ [-r]_d & 1 & 0 & 0 \\ [-r^2]_d & 0 & 1 & 0 \\ & & & \ddots \\ [-r^{n-1}]_d & 0 & 0 & 1 \end{bmatrix}. \tag{1}$$

where $[x]_d = x \bmod d$. Such lattices can be represented implicitly by two integers d and r , thus the key size is reduced by a factor of n .

However, they failed to implement the FHE scheme since they were unable to generate keys for lattice of dimension $n > 2048$. One of the main obstacles lies in the inefficiency of generating ideal lattices with prime determinant by trial and error.

3.2 The Gentry-Halevi’s Key Generation

In [7], Gentry and Halevi found that it was not necessary for the determinant to be prime. Instead, they showed that the scheme can go through as long as the ideal lattice has an odd determinant and a simple-HNF. The key generation consists of the following steps.

- (i) Fix $f(x) = x^n + 1$ with n a power of 2. Then they choose a vector $\mathbf{v} = (v_0, \dots, v_{n-1})$, where each entry v_i is chosen at random as a t -bit integer. Consider the ideal lattice generated by $v(x) = v_{n-1}x^{n-1} + \dots + v_1x + v_0$ in $\mathbb{Z}[x]/\langle f(x) \rangle$. Then the lattice is generated by the following basis

$$\mathbf{V} = \begin{bmatrix} v_0 & v_1 & v_2 & v_{n-1} \\ -v_{n-1} & v_0 & v_1 & v_{n-2} \\ -v_{n-2} & -v_{n-1} & v_0 & v_{n-3} \\ & & & \ddots \\ -v_1 & -v_2 & -v_3 & v_0 \end{bmatrix}.$$

- (ii) Given $v(x)$ and $f(x)$, use their algorithm to compute the resultant d of $v(x)$ and $f(x)$ and the constant term w_0 of $w(x)$, where $w(x) = w_{n-1}x^{n-1} + \dots + w_1x + w_0$ is the unique polynomial such that $v(x)w(x) = d \bmod f(x)$. If d is odd, compute w_1 by applying their algorithm to $(xv(x) \bmod f(x))$ and $f(x)$.
- (iii) If $\gcd(w_1, d) = 1$, compute $r = \frac{w_0}{w_1} \bmod d$, check whether $r^n = -1 \bmod d$. If so, they consider that the ideal lattice has the desired HNF, then they compute an odd coefficient w_i via $w_i = rw_{i+1} \bmod d$ and w_0, w_1 (The subscripts are modulo n .) and use (d, r) as the public key and w_i the secret key.

For the sake of clarity and comparison, the key generation algorithm is summarized in Algorithm 1. We need to emphasize that there are some details missing in their description (Lines 3–5 and Lines 7–9 in Algorithm 1) and Algorithm 1 is adopted from their source code.

Algorithm 1. Key Generation in [7]

Input: dimension n , bit length t .

Output: $\text{pk}=(d, r)$, $\text{sk}=w_i$.

- 1 Choose a random n -dimensional vector \mathbf{v} , where each v_i is a t -bit integer.
 - 2 Compute the resultant d of $v(x)$ and $f(x)$, and the constant term w_0 of $w(x)$, where $w(x)v(x) = d \pmod{f(x)}$.
 - 3 **if** d is even **then**
 - 4 | Go to 1.
 - 5 **end**
 - 6 Compute the coefficient w_1 of $w(x)$.
 - 7 **if** w_1 has no inverse modulo d **then**
 - 8 | Go to 1.
 - 9 **end**
 - 10 Compute $r = \frac{w_0}{w_1} \pmod{d}$.
 - 11 **if** $r^n \neq -1 \pmod{d}$ **then**
 - 12 | Go to 1.
 - 13 **else**
 - 14 | Compute an odd w_i via $w_i = rw_{i+1} \pmod{d}$ and w_0, w_1 . (The subscripts are modulo n .)
 - 15 | Output: $\text{pk}=(d, r)$, $\text{sk}=w_i$.
 - 16 **end**
-

3.3 Remarks on Gentry-Halevi’s Key Generation

The authors of [7] solved the main issues that prevented the key generation in [18] from being practical. They also gave many optimizations focusing on practical performance. However, we note that there are several natural questions regarding to their implementation.

1. In [7], the authors mentioned that “it was observed by Nigel Smart that the HNF has the correct form whenever the determinant is odd and square-free. Indeed, in our tests this condition was met with probability roughly 0.5...” Since the determinants in the experiments are numbers with millions of bits, it is difficult to determine whether they are square-free or not. We believe that “this condition” means “the HNF has the correct form” rather than “the determinant is odd and square-free”. We also rerun the experiment to confirm our assertion.
2. In step (iii), they did not mention what to do if $\gcd(w_1, d) \neq 1$. Generally, there are two options. We can start over by choosing another lattice, or we can compute another random coefficient of $w(x)$ until we find one that is

coprime to d . (Judging from their code, they did the former.) In fact, we will show if $\gcd(w_1, d) \neq 1$, then all the coefficients of $w(x)$ are not coprime to d . That means the latter is not optional.

3. Also in step (iii), when $\gcd(w_1, d) = 1$, we compute r , and check if $r^n = -1 \pmod d$ holds. Even if this equality held, they did not show that the HNF has the desired form. In other words, they did not provide a proof for the correctness of their algorithm. They did mention to check a serial of conditions that could guarantee the HNF is simple, however, in their implementation they actually tested only the last condition. It is vital for the FHE scheme to work correctly. We will give a simpler condition and by checking our condition holds, we can guarantee the HNF is indeed simple.

4 Improved Key Generation for Gentry's FHE Scheme

In this section, we present our improved key generation algorithm for Gentry's FHE scheme. Our improvements consist of two aspects:

- we can generate ideal lattices with odd determinant deterministically.
- we present a rigorous proof for the correctness of our algorithm by checking a simpler sufficient and necessary condition for the ideal lattice having a simple-HNF.

Before presenting our key generation algorithm, we first show some new properties on ideal lattices. All the proofs can be found in the appendix.

4.1 Some Properties of Ideal Lattices

The following lemma was already stated without proof in [7].

Lemma 1. *Let \mathcal{L} be the ideal lattice generated by $g(x) \in R = \mathbb{Z}[x]/\langle f(x) \rangle$, where $f(x)$ is a monic polynomial of degree n and $g(x)$ is of degree m . Assume $g(x)$ is relatively prime to $f(x)$, then $\det(\mathcal{L}) = |\text{Res}(g, f)| = |\text{Res}(f, g)|$.*

The next lemma shows that the HNF of an ideal lattice has a very special structure.

Lemma 2. *Let \mathcal{L} be the ideal lattice generated by $g(x) \in R = \mathbb{Z}[x]/\langle f(x) \rangle$, where $f(x)$ is a monic polynomial of degree n and is relatively prime to $g(x)$. Then the Hermite Normal Form of \mathcal{L}*

$$\mathbf{H} = \begin{bmatrix} h_{1,1} & & & & \\ h_{2,1} & h_{2,2} & & & \\ \vdots & \vdots & \ddots & & \\ h_{n,1} & h_{n,2} & \cdots & h_{n,n} & \end{bmatrix}$$

satisfies $h_{i,i} | h_{j,l}$, for $1 \leq l \leq j \leq i \leq n$.

Remark 1. A part of the result has already been proven in [4] for identifying ideal lattices. In fact, they proved that the diagonal entries of an HNF form a division chain, that is, $h_{n,n} | \cdots | h_{2,2} | h_{1,1}$. Our results show that off-diagonal entries are also divisible by the corresponding diagonal entry in the same row, which means the primitive part¹ of the polynomial corresponding to each row is monic.

Remark 2. For the HNF of a full-rank lattice, the first row is the *primitive lattice vector* of the form $(d, 0, \dots, 0)$, where $d > 0$ is a factor of the determinant. In the case of ideal lattice, the first row corresponds to the smallest positive constant polynomial in the ideal. This characterization will be useful for proving Proposition 3.

Lemma 3. *Let the notations be the same as in Lemma 2 and let $H_i(x)$ denote the corresponding polynomial of i -th row in the HNF, $i \leq n$. Let β be the root of $h_{2,1} + h_{2,2}x$. Then $H_i(\beta) = 0 \pmod{\frac{h_{1,1}h_{i,i}}{h_{2,2}}}$, $\forall i \geq 2$ and $f(\beta) = 0 \pmod{\frac{h_{1,1}}{h_{2,2}}}$.*

4.2 Generating Ideal Lattice with Odd Determinant Deterministically

The key generation algorithm in [7] needs to generate an ideal lattice with odd determinant. The original paper achieved this by trial and error. We first show that the parity of the determinant is related to the generator in a very simple way, and then give a deterministic way to generate an ideal lattice with odd determinant.

In the next two subsections, we fix $f(x) = x^n + 1$ with n a power of 2.

Proposition 2. *Let \mathcal{L} be the ideal lattice generated by $v(x) = v_{n-1}x^{n-1} + \cdots + v_1x + v_0$ in $\mathbb{Z}[x]/\langle f(x) \rangle$. Then $\det(\mathcal{L}) \equiv v_0 + v_1 + \cdots + v_{n-1} \pmod{2}$.*

From the above proposition, we know how to generate ideal lattices with odd determinant deterministically. Specifically, we choose a random n -dimensional vector \mathbf{v} from the set

$$\{v(x) \in \mathbb{Z}[x]/\langle f(x) \rangle : v_i \text{ is a random } t\text{-bit integer and } \sum_{i=0}^{n-1} v_i \equiv 1 \pmod{2}\}.$$

This can be done by choosing v_i randomly, and use $v(x) + (v(1) + 1 \pmod{2})$ as the generator or simply use $2u(x) + 1$, where $u(x)$ is a random vector. Then the generated ideal lattice has an odd determinant.

¹ The primitive part of an integer polynomial $s(x)$ is $s(x)/r$, where r is the g.c.d of the coefficients of $s(x)$.

4.3 A Simpler Condition for Checking the HNF

In this part, we give a simpler condition, which yields a rigorous proof for the correctness of our algorithm. Our condition is based on the following proposition.

Proposition 3. *Let \mathcal{L} be the ideal lattice generated by $v(x)$ in $\mathbb{Z}[x]/\langle f(x) \rangle$. Suppose $w(x) \in \mathbb{Z}[x]/\langle f(x) \rangle$ is a polynomial such that $v(x)w(x) = d \pmod{f(x)}$ where d is the determinant of \mathcal{L} . Then the following conditions are equivalent.*

- (1) \mathcal{L} has a simple-HNF.
- (2) \mathcal{L} contains a vector of the form $\mathbf{r} = (-r, 1, 0, \dots, 0)$.
- (3) There exists an index i , $0 \leq i \leq n-1$, $\gcd(w_i, d) = 1$.
- (4) For arbitrary $0 \leq i \leq n-1$, $\gcd(w_i, d) = 1$.

From the above proposition, we can conclude that in order to check whether the ideal lattice has a simple-HNF, we only need to compute a w_i and $\gcd(w_i, d)$, which is simpler and more efficient to check. If $\gcd(w_i, d) = 1$, then from the proposition, the lattice has a simple-HNF. Otherwise, the lattice does not have a simple-HNF.

In what follows, we show that once w_1 has an inverse modulo d , it must hold that $r^n = -1 \pmod{d}$. Thus the checking can be safely left out.

Proposition 4. *Let \mathcal{L} be the ideal lattice generated by $v(x)$ in $\mathbb{Z}[x]/\langle f(x) \rangle$. Suppose $w(x) \in \mathbb{Z}[x]/\langle f(x) \rangle$ is a polynomial such that $v(x)w(x) = d \pmod{f(x)}$ where d is the determinant of \mathcal{L} . Assume that $\gcd(w_1, d) = 1$ and $r = \frac{w_0}{w_1} \pmod{d}$, then $r^n = -1 \pmod{d}$.*

4.4 Our Key Generation Algorithm

We formally present our improved key generation algorithm (Algorithm 2) and give some theoretical analysis and experimental results supporting our claims in this section.

Theoretical Analysis. We first give time estimations for both algorithms. Here we show a few notations to represent the time for different routines.

- T_{res} : time to compute the resultant and w_i using the algorithm in [7]. Note that in both algorithms, we sometimes only need to compute the resultant or a single coefficient w_i , this will also take time T_{res} since the exactly same routine is used to achieve these goals.
- T_{xgcd} : time to apply the extended Euclidean algorithm.
- T_{pmod} : time to compute $r^n \pmod{d}$.
- T_{mul} : time to do one multiplication modulo d .
- T_{oddcoc} : time to compute an odd coefficient of $w(x)$ (line 14 in Algorithm 1 and line 7 in our algorithm).

Algorithm 2. Our Key Generation Algorithm

Input: dimension n , bit length t .
Output: $\text{pk}=(d, r)$, $\text{sk}=w_i$.

- 1 Choose a random vector v from $\{v(x) = \sum_{i=0}^{n-1} v_i x^i : v_i \text{ is a random } t\text{-bit integer and } \sum_{i=0}^{n-1} v_i \equiv 1 \pmod{2}\}$.
- 2 Compute the resultant d of $v(x)$ and $f(x)$, and coefficient w_1 of the linear term of $w(x)$, where $w(x)v(x) = d \pmod{f(x)}$.
- 3 **if** $\gcd(w_1, d) \neq 1$ **then**
- 4 | Go to 1.
- 5 **else**
- 6 | Compute w_0 and $r = \frac{w_0}{w_1} \pmod{d}$.
- 7 | Compute an odd w_i via $w_i = r w_{i+1} \pmod{d}$ and w_0, w_1 . (The subscripts are modulo n .)
- 8 | Output: $\text{pk}=(d, r)$, $\text{sk}=w_i$.
- 9 **end**

For Algorithm 1, since the probability of choosing an odd-determinant ideal lattice is 0.5, we need to try twice on average to get an odd-determinant ideal lattice, which costs time $2T_{res}$ in order to compute the determinants. At the same time, we have also computed a coefficient w_0 . Afterwards, we need to compute another coefficient w_1 , which costs time T_{res} . From the experiment, we found that the probability of an odd-determinant ideal lattice having simple-HNF is very high, we simply omit the failure cases. Then computing the inverse of w_1 and r takes time T_{xgcd} and T_{mul} respectively, and checking if $r^n = -1 \pmod{d}$ takes time T_{pmod} . Finally we need to choose an odd coefficient of $w(x)$, which costs time T_{oddcoc} . So the expected running time of Algorithm 1 is $3T_{res} + T_{xgcd} + T_{pmod} + T_{mul} + T_{oddcoc}$.

In our algorithm, the ideal lattice always has an odd determinant. As before, we omit the failure cases where an odd-determinant ideal lattice doesn't have a simple-HNF, then we only need to compute two consecutive coefficients of $w(x)$, which costs $2T_{res}$. Then it costs time $T_{xgcd} + T_{mul}$ to perform an extended Euclidean algorithm and multiplication modulo d to compute r . At last the time for choosing an odd coefficient of $w(x)$ is T_{oddcoc} . Thus the total time used in our key generation algorithm is $2T_{res} + T_{xgcd} + T_{mul} + T_{oddcoc}$.

In our experiments, we found that T_{res} takes most proportion of the whole time. So on average, our algorithm is about 1.5 times faster.

Experimental Results. We present our experimental results in the following. Our experiments were performed on a PC (Intel(R) Core(TM) i7, 3.4 GHz, 2G RAM) and based on Shoup's NTL library [17] (version 9.10.0). NTL is installed with MPC (version 1.0.2), GMP (version 5.1.3), and MPFR (version 3.1.5).

1. We rerun the experiments of [7] to confirm our assertion in Sect. 3.3. We generated 100 random ideal lattices for each parameter set, and count the number of

ideal lattices in different categories. In the table, the first row are the parameters (n, t) , like $(512, 380)$, where n is the dimension of the ideal lattice and t is the bit length of each coefficient in the generator. “even (or odd) d ” indicates the generated lattice has even (or odd) determinant, and “(non-)SHNF” indicates the lattice has (doesn’t have) a simple-HNF.

	(512, 380)		(2048, 380)		(8192, 380)		(32768, 380)	
	Algorithm 1	Algorithm 2	Algorithm 1	Algorithm 2	Algorithm 1	Algorithm 2	Algorithm 1	Algorithm 2
Even d , SHNF	25	0	32	0	25	0	24	0
Even d , non-SHNF	27	0	25	0	18	0	25	0
Odd d , SHNF	48	98	42	98	57	100	47	90
Odd d , non-SHNF	0	2	1	2	0	0	4	10

From the experiment we can see that the probability of a randomly generated ideal lattice in Algorithm 1 having the correct form (odd-determinant and simple-HNF) is roughly 0.5 and that the failure cases is usually due to the determinant being even.

Also, there is something interesting here. When the ideal lattices have an even determinant, about half of them have a simple-HNF. It is not surprising if the probability of a random ideal lattice having simple-HNF is somewhat higher than 44% due the divisibility of the diagonal entries of its HNF. However, when the ideal lattices have an odd determinant, most of them also have a simple-HNF. As a result, the probability of simple-HNF is about 75%, which is much higher than 44%. We conjecture this might has something to do with the underlying 2-power cyclotomic fields.

2. We present some experimental results on the running time of both algorithms. In the experiments, we generated 20 valid keys and count the averaged time for the different parts in theoretical analysis. In the next table, the “#(trials)” column denotes the number of trials we did to get 20 valid keys. t_{res} , t_{xgcd} , t_{pmod} , t_{mul} and t_{oddcoe} are averaged time for doing the corresponding operations, so $t_{res} \approx 3T_{res}$ in Algorithm 1. t_{total} is the averaged time to generate a valid key. All the timings are measured in seconds.

We can see that the ratio between t_{res} in Algorithm 1 and t_{res} in Algorithm 2 is about 1.5, which matches the theoretical prediction $(3T_{res}/2T_{res})$. In two algorithms, t_{xgcd} and t_{mul} are almost the same and t_{oddcoe} is very short. We note that this is because in most cases one of w_0 and w_1 is odd. From the experiments, we can see that our key generation is about 1.5 (more close to 1.6) times faster.

dim n	bit-size t	Gentry and Halevi's algorithm						
		# (trials)	t_{res}	t_{xgcd}	t_{pmod}	t_{mul}	t_{oddcoc}	t_{total}
512	380	37	0.171	0.048	0.054	0.007	0.003	0.284
2048	380	46	1.585	0.354	0.347	0.037	0.009	2.336
8192	380	36	12.068	2.120	2.040	0.173	0.086	16.500
32768	380	46	123.152	14.128	14.494	0.962	0.799	153.591
dim n	bit-size t	Our algorithm						
		# (trials)	t_{res}	t_{xgcd}	t_{mul}	t_{oddcoc}	t_{total}	Speedup
512	380	20	0.118	0.047	0.007	0.002	0.174	1.632x
2048	380	20	0.966	0.351	0.038	0.036	1.393	1.677x
8192	380	20	7.568	2.108	0.173	0.069	9.924	1.663x
32768	380	22	78.276	13.882	0.964	0.432	93.592	1.641x

5 Conclusion

In this paper, we made improvements on the key generation of [7]. Using the properties of ideal lattice, we present an improved key generation algorithm with a rigorous proof for correctness. As a result, our algorithm is about 1.5 times faster.

Acknowledgements. The authors would like to thank all anonymous referees of ISC'2017 and ICISC'2017 for their valuable comments that greatly improve the manuscript. This work is supported by the National Natural Science Foundation of China (No. Y31005A102, No. Y610092302).

A Appendix

A.1 Proof of Lemma 1

Proof. Note that

$$\text{Sylv}(g(x), f(x)) = \begin{bmatrix} g(x) \\ \vdots \\ x^{n-1}g(x) \\ f(x) \\ \vdots \\ x^{m-1}f(x) \end{bmatrix} = \begin{bmatrix} g_0 & g_1 & \cdots & g_m & & & \\ & g_0 & g_1 & \cdots & g_m & & \\ & & \ddots & & \ddots & & \\ & & & g_0 & g_1 & \cdots & g_m \\ f_0 & & & & & f_n & \\ \vdots & \ddots & & & & & \ddots \\ & f_0 & & & & & f_n \end{bmatrix} \cdot$$

Since $f_n = 1$, the Sylvester matrix can always be transformed unimodularly into the following (block-triangular) form by adding proper multiples of rows in the lower half to each row on the top half,

$$\begin{bmatrix} \mathbf{B}_{n \times n} & \mathbf{0} \\ f_0 & \cdots & f_n \\ \ddots & & \ddots \\ & f_0 & \cdots & f_n \end{bmatrix} = \begin{bmatrix} g(x) \pmod{f(x)} \\ \vdots \\ \frac{x^{n-1}g(x) \pmod{f(x)}}{f(x)} \\ \vdots \\ x^{m-1}f(x) \end{bmatrix}.$$

It follows that $\text{Res}(g, f) = f_n^m \det(\mathbf{B}) = \det(\mathbf{B})$. Since $g(x)$ is relatively prime to $f(x)$, \mathcal{L} is a full-rank lattice with basis \mathbf{B} . Therefore, $\det(\mathcal{L}) = |\det(\mathbf{B})| = |\text{Res}(g, f)| = |\text{Res}(f, g)|$. \square

A.2 Proof of Lemma 2

Proof. We prove the lemma by induction on i .

For $i = 1$, it is trivial.

For $i = 2$, consider the first row, which corresponds to the constant polynomial $h_{1,1}$. Since \mathcal{L} is an ideal lattice, the vector $(0, h_{1,1}, 0, \dots, 0)$, which corresponds to the polynomial $h_{1,1}x$, is also in \mathcal{L} .

It's obvious that $(0, h_{1,1}, 0, \dots, 0) = x_1\mathbf{H}_1 + x_2\mathbf{H}_2$ for some $x_1, x_2 \in \mathbb{Z}$. Then $h_{1,1} = x_2h_{2,2}$, $x_1h_{1,1} + x_2h_{2,1} = 0$. Hence $h_{2,2}|h_{1,1}$, $h_{2,2}|h_{2,1}$, which completes the proof for $i = 2$.

Assume the result holds for $i \leq k \leq n - 1$, $h_{i,i}|h_{j,l}$, where $1 \leq l \leq j \leq i \leq k$. We show that for $i = k + 1$, $h_{k+1,k+1}|h_{j,l}$.

Consider the k -th row. The corresponding polynomial of k -th row is

$$h_{k,k}x^{k-1} + h_{k,k-1}x^{k-2} + \cdots + h_{k,2}x + h_{k,1}.$$

After multiplying x , we get a vector $(0, h_{k,1}, \dots, h_{k,k}, 0, \dots, 0)$, which is a linear combination of $\mathbf{H}_1, \dots, \mathbf{H}_{k+1}$ with integer coefficients, i.e.

$$(0, h_{k,1}, \dots, h_{k,k}, 0, \dots, 0) = \sum_{i=1}^{k+1} y_i \mathbf{H}_i,$$

where $y_i \in \mathbb{Z}$, for $i = 1, \dots, k + 1$.

So

$$\begin{cases} h_{k,k} & = y_{k+1}h_{k+1,k+1} \\ h_{k,k-1} & = y_k h_{k,k} + y_{k+1}h_{k+1,k} \\ & \vdots \\ h_{k,1} & = \sum_{i=2}^{k+1} y_i h_{i,2} \\ 0 & = \sum_{i=1}^{k+1} y_i h_{i,1} \end{cases} \tag{2}$$

From the first equation, we get $y_{k+1} = \frac{h_{k,k}}{h_{k+1,k+1}}$ and

$$\begin{cases} h_{k+1,k} &= \frac{h_{k,k-1} - y_k h_{k,k}}{h_{k,k}} h_{k+1,k+1} \\ h_{k+1,k-1} &= \frac{h_{k,k-2} - y_{k-1} h_{k-1,k-1} - y_k h_{k,k-1}}{h_{k,k}} h_{k+1,k+1} \\ &\vdots \\ h_{k+1,2} &= \frac{h_{k,1} - \sum_{i=2}^k y_i h_{i,2}}{h_{k,k}} h_{k+1,k+1} \\ h_{k+1,1} &= \frac{-\sum_{i=1}^k y_i h_{i,1}}{h_{k,k}} h_{k+1,k+1} \end{cases}.$$

From the induction hypothesis, we have $h_{k,k} | h_{j,l}$ for $1 \leq l \leq j \leq k \leq n$. So the coefficient of $h_{k+1,k+1}$ in each equation is in fact an integer, therefore $h_{k+1,k+1} | h_{k+1,l}$, $1 \leq l \leq k+1$. Since $h_{k+1,k+1} | h_{k,k}$, we know $h_{k+1,k+1} | h_{j,l}$, where $1 \leq l \leq j \leq k+1 \leq n$. Thus, the result holds for $i = k+1$.

By the principle of induction, the lemma follows. \square

A.3 Proof of Lemma 3

Proof. We prove the first equality by induction on i .

For $i = 2$, by the definition of β , $H_2(\beta) = 0 \pmod{h_{1,1}}$.

Assume $H_i(\beta) = 0 \pmod{\frac{h_{1,1} h_{i,i}}{h_{2,2}}}$, for $i \leq k$, where $2 \leq k \leq n-1$. From Eq. 2, we have

$$\begin{cases} h_{k,k} x^k &= y_{k+1} h_{k+1,k+1} x^k \\ h_{k,k-1} x^{k-1} &= y_k h_{k,k} x^{k-1} + y_{k+1} h_{k+1,k} x^{k-1} \\ &\vdots \\ h_{k,1} x &= \sum_{i=2}^{k+1} y_i h_{i,2} x \\ 0 &= \sum_{i=1}^{k+1} y_i h_{i,1} \end{cases}.$$

Sum the equations up,

$$y_{k+1} H_{k+1}(x) + y_k H_k(x) + y_{k-1} H_{k-1}(x) + \cdots + y_1 H_1(x) = x H_k(x),$$

Set $x = \beta$,

$$y_{k+1} H_{k+1}(\beta) + (y_k - \beta) H_k(\beta) + y_{k-1} H_{k-1}(\beta) + \cdots + y_1 H_1(\beta) = 0.$$

Note that $H_1(x) = h_{1,1}$, $H_1(\beta) = 0 \pmod{\frac{h_{1,1} h_{k,k}}{h_{2,2}}}$. By induction hypothesis, $H_i(\beta) = 0 \pmod{\frac{h_{1,1} h_{i,i}}{h_{2,2}}}$ and $\frac{h_{1,1} h_{i+1,i+1}}{h_{2,2}} | \frac{h_{1,1} h_{i,i}}{h_{2,2}}$, $y_{k+1} H_{k+1}(\beta) = 0 \pmod{\frac{h_{1,1} h_{k,k}}{h_{2,2}}}$.

Since $y_{k+1} = \frac{h_{k,k}}{h_{k+1,k+1}}$, we have

$$H_{k+1}(\beta) = 0 \pmod{\frac{h_{1,1} h_{k+1,k+1}}{h_{2,2}}}.$$

Therefore for $i = k+1$, the equality also holds. Thus $H_i(\beta) = 0 \pmod{\frac{h_{1,1} h_{i,i}}{h_{2,2}}}$, $\forall i \geq 2$.

For the second equality, note that $xH_n(x) \bmod f(x) = xH_n(x) - h_{n,n}f(x)$. Since the vector corresponding to $xH_n(x) - h_{n,n}f(x)$ is a lattice vector, there exist integers $z_1 \cdots z_n \in \mathbb{Z}$, $xH_n(x) - h_{n,n}f(x) = \sum_{i=1}^n z_i H_i(x)$.

Set $x = \beta$,

$$\beta H_n(\beta) - h_{n,n}f(\beta) = \sum_{i=1}^n z_i H_i(\beta).$$

Since $h_{n,n} | h_{i,i}$ for all i , $H_i(\beta) = 0 \pmod{\frac{h_{1,1}h_{n,n}}{h_{2,2}}}$, $\forall i \geq 2$. Also $H_1(\beta) = 0 \pmod{\frac{h_{1,1}h_{n,n}}{h_{2,2}}}$. Then $h_{n,n}f(\beta) = 0 \pmod{\frac{h_{1,1}h_{n,n}}{h_{2,2}}}$ and

$$f(\beta) = 0 \pmod{\frac{h_{1,1}}{h_{2,2}}}.$$

□

A.4 Proof of Proposition 2

Proof. Since we only concern the parity of the determinant, we work over \mathbb{F}_2 . Denote

$$\mathbf{P} = \begin{bmatrix} 0 & 1 & & 0 & 0 \\ 0 & 0 & & 0 & 0 \\ & & \ddots & & \\ 0 & 0 & & 0 & 1 \\ 1 & 0 & & 0 & 0 \end{bmatrix}.$$

Then

$$\mathbf{V} = \begin{bmatrix} v_0 & v_1 & v_2 & & v_{n-1} \\ -v_{n-1} & v_0 & v_1 & & v_{n-2} \\ -v_{n-2} & -v_{n-1} & v_0 & & v_{n-3} \\ & & & \ddots & \\ -v_1 & -v_2 & -v_3 & & v_0 \end{bmatrix} = v_0 \mathbf{I} + \cdots + v_{n-1} \mathbf{P}^{n-1} = v(\mathbf{P}) \text{ over } \mathbb{F}_2.$$

Now we compute the eigenvalues of \mathbf{P} .

Since \mathbf{P} is a cyclic shift matrix, $\mathbf{P}^n = \mathbf{I}$. Note that $x^n + 1 = (x + 1)^n$ over \mathbb{F}_2 (n is a power of 2), then all the eigenvalues of \mathbf{P} are 1. All the eigenvalues of \mathbf{V} are thus $v(1) = v_0 + v_1 + \cdots + v_{n-1}$. Hence,

$$\det(\mathcal{L}) \equiv (v_0 + v_1 + \cdots + v_{n-1})^n \equiv v_0 + v_1 + \cdots + v_{n-1} \pmod{2}.$$

□

A.5 Proof of Proposition 3

Proof.

(1) \Leftrightarrow (2) The equivalence between the two conditions was proved in [7].

(2) \Rightarrow (3) Assuming first that $\mathbf{r} = (-r, 1, 0, \dots, 0) \in \mathcal{L}$. Then there exists $y(x) \in \mathbb{Z}[x]/\langle f(x) \rangle$, such that $y(x)v(x) = r(x) \pmod{f(x)}$. Therefore,

$$y(x)v(x)w(x) = r(x)w(x) \pmod{f(x)}.$$

$$dy(x) = r(x)w(x) \pmod{f(x)}.$$

Note that $f(x) = x^n + 1$, we have

$$(-w_{n-1}, \dots, w_{n-3}, w_{n-2}) - r(w_0, \dots, w_{n-2}, w_{n-1}) = 0 \pmod{d}.$$

So $w_0r = -w_{n-1} \pmod{d}$ and $w_{i+1}r = w_i \pmod{d}$, for $0 \leq i \leq n-2$.

We prove by contradiction that $\exists 0 \leq i \leq n-1$, $\gcd(w_i, d) = 1$.

If for arbitrary $0 \leq i \leq n-1$, $\gcd(w_i, d) \neq 1$, let $\mu = \gcd(w_0, d) > 1$. From the relations among the w_i 's, we know μ divides all the w_i 's.

Therefore, $\mu \mid \gcd(w_0, \dots, w_{n-1}, d)$. Hence $\frac{d}{\mu} = \frac{w(x)}{\mu}v(x) \pmod{f(x)}$ is a lattice vector, which means the first diagonal of the HNF is a proper factor of d . So the second diagonal can't be 1, otherwise the determinant is $\frac{d}{\mu}$ rather than d . This is a contradiction. Therefore $\exists 0 \leq i \leq n-1$, $\gcd(w_i, d) = 1$.

(3) \Rightarrow (4) Assume $\exists 0 \leq i \leq n-1$, $\gcd(w_i, d) = 1$, fix i . We also prove by contradiction. Suppose there exists a $0 \leq j \leq n-1$ such that $\mu = \gcd(w_j, d) > 1$. Due to Lemma 2, we can assume the second row of HNF is $(-\alpha r, \alpha, 0, \dots, 0)$, for some $\alpha \in \mathbb{N}^+$. Then $\alpha^2 \mid \alpha h_{1,1} \mid d$ and $\gcd(\alpha, w_i) = 1$. Similar to previous proof,

$$\alpha(-w_{n-1}, \dots, w_{n-3}, w_{n-2}) - \alpha r(w_0, \dots, w_{n-2}, w_{n-1}) = 0 \pmod{d}.$$

Hence

$$(-w_{n-1}, \dots, w_{n-3}, w_{n-2}) - r(w_0, \dots, w_{n-2}, w_{n-1}) = 0 \pmod{\frac{d}{\alpha}}.$$

According to the steps above, $w_{i+1}r = w_i \pmod{\frac{d}{\alpha}}$, for $i \leq n-2$ and $w_0r = -w_{n-1} \pmod{\frac{d}{\alpha}}$. Since $\alpha^2 \mid d$, $\frac{d}{\alpha}$ and d share exactly the same prime factors, hence $\gcd(w_j, \frac{d}{\alpha}) = \mu' > 1$. Similar to the previous proof, we have μ' divides all the coefficients of $w(x)$. Specifically, $\mu' \mid w_i$. Hence $\mu' \mid \gcd(w_i, d)$, a contradiction. Thus $\gcd(w_i, d) = 1$ for any $0 \leq i \leq n-1$.

(4) \Rightarrow (2) Assume for any $0 \leq i \leq n-1$, $\gcd(w_i, d) = 1$. Since $d = w(x)v(x) \pmod{f(x)}$, $\gcd(w_0, \dots, w_{n-1}) \mid d$. Hence $\gcd(w_0, \dots, w_{n-1}) = 1$. Then $(d, 0, \dots, 0)$ is a primitive lattice vector in \mathcal{L} . According to Remark 2, it's the first row of the HNF of \mathcal{L} , which means all the other diagonals in the HNF are 1. Thus, \mathcal{L} contains a vector (the second row of its HNF) of the form $\mathbf{r} = (-r, 1, 0, \dots, 0)$.

□

A.6 Proof of Proposition 4

Proof. From the assumptions and the proof for Proposition 3, we know that $(-r, 1, 0, \dots, 0)$ is in \mathcal{L} . According to Lemma 3, r is a root of $f(x) = 0 \pmod{d}$. Therefore $r^n = -1 \pmod{d}$. \square

References

1. Ajtai, M., Dwork, C.: A public-key cryptosystem with worst-case/average-case equivalence. In: Symposium on Theory of Computing, pp. 284–293 (1997). <https://doi.org/10.1145/258533.258604>
2. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. In: ITCS, pp. 309–325 (2012). <https://doi.org/10.1145/2090236.2090262>
3. Clear, M., McGoldrick, C.: Multi-identity and multi-key leveled FHE from learning with errors. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9216, pp. 630–656. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_31
4. Ding, J., Lindner, R.: Identifying ideal lattice. IACR Cryptology ePrint Archive, 322 (2007)
5. Ducas, L., Micciancio, D.: FHEW: bootstrapping homomorphic encryption in less than a second. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 617–640. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46800-5_24
6. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Symposium on Theory of Computing, pp. 169–178 (2009). <https://doi.org/10.1145/1536414.1536440>
7. Gentry, C., Halevi, S.: Implementing Gentry’s fully-homomorphic encryption scheme. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 129–148. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20465-4_9. <http://researcher.watson.ibm.com/researcher/files/us-shaih/fhe-code.zip>
8. Gentry, C., Halevi, S., Smart, N.P.: Homomorphic evaluation of the AES circuit. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 850–867. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_49. Updated implementation version: <https://eprint.iacr.org/2012/099.pdf>
9. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 75–92. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_5
10. Hu, G., Pan, Y., Liu, R., Chen, Y.: On random nonsingular Hermite Normal Form. J. Number Theory **164**, 66–86 (2016). <https://doi.org/10.1016/j.jnt.2015.12.004>
11. Li, Z., Zhu, X., Lian, Y., et al.: Constructing secure content-dependent watermarking scheme using homomorphic encryption. In: IEEE International Conference on Multimedia and Expo, pp. 627–630 (2007). <https://doi.org/10.1109/icme.2007.4284728>
12. Maze, G.: Natural density distribution of Hermite Normal Forms of integer matrices. J. Number Theory **131**(12), 2398–2408 (2011). <https://doi.org/10.1016/j.jnt.2011.06.010>

13. Rivest, R., Adleman, L., Dertouzos, M.: On data banks and privacy homomorphisms. *Found. Secur. Comput.* **4**(11), 169–180 (1978)
14. Rivest, R., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **21**(2), 120–126 (1978). <https://doi.org/10.21236/ada606588>
15. Rohloff, K., Cousins, D.B.: A scalable implementation of fully homomorphic encryption built on NTRU. In: Böhme, R., Brenner, M., Moore, T., Smith, M. (eds.) *FC 2014*. LNCS, vol. 8438, pp. 221–234. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44774-1_18
16. Scholl, P., Smart, N.P.: Improved key generation for Gentry’s fully homomorphic encryption scheme. In: Chen, L. (ed.) *IMACC 2011*. LNCS, vol. 7089, pp. 10–22. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25516-8_2
17. Shoup, V.: NTL: a library for doing number theory. <http://www.shoup.net/ntl/>
18. Smart, N.P., Vercauteren, F.: Fully homomorphic encryption with relatively small key and ciphertext sizes. In: Nguyen, P.Q., Pointcheval, D. (eds.) *PKC 2010*. LNCS, vol. 6056, pp. 420–443. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13013-7_25
19. Yi, X., Kaosar, M.G., Paulet, R., Bertino, E.: Single-database private information retrieval from fully homomorphic encryption. *IEEE Trans. Knowl. Data Eng.* **25**(5), 1125–1134 (2013). <https://doi.org/10.1109/tkde.2012.90>



Subring Homomorphic Encryption

Seiko Arita^(✉) and Sari Handa

Graduate School of Information Security, Institute of Information Security,
Yokohama, Japan
{arita,dgs158101}@iisec.ac.jp

Abstract. In this paper, we construct *subring homomorphic encryption* scheme that is a homomorphic encryption scheme built on the decomposition ring, which is a subring of cyclotomic ring. In the scheme, each plaintext slot contains an integer in \mathbb{Z}_{p^l} , rather than an element of $\text{GF}(p^d)$ as in conventional homomorphic encryption schemes on cyclotomic rings. Our benchmark results indicate that the subring homomorphic encryption scheme is several times faster than HELib for *mod- p^l integer plaintexts*, due to its high parallelism of *mod- p^l integer slot structure*. We believe in that such plaintext structure composed of *mod- p^l integer slots* will be more natural, easy to handle, and significantly more efficient for many applications such as outsourced data mining, than conventional $\text{GF}(p^d)$ slots.

Keywords: Fully homomorphic encryption · Ring-LWE
Cyclotomic ring · Decomposition ring · Plaintext slots

1 Introduction

Background. Homomorphic encryption (HE) scheme enables us computation on encrypted data. One can add or multiply (or more generally “evaluate”) given ciphertexts and generate a new ciphertext that encrypts the sum or product (or “evaluation”) of underlying data of the input ciphertexts. Such computation (called *homomorphic addition or multiplication or evaluation*) can be done without using the secret key and one will never know anything about the processed or generated data.

Since the breakthrough construction given by Gentry [7], many efforts are dedicated to make such homomorphic encryption scheme more secure and more efficient. Especially, HE schemes based on the Ring-LWE problem [4, 6, 15, 16] have obtained theoretically-sound proof of security and well-established implementations such as HELib [10] and SEAL v2.0 [14]. Nowadays many researchers apply HE schemes to privacy-preserving tasks for mining of outsourced data such as genomic data, medical data, financial data and so on [5, 9, 11–13].

Our perspective: $\text{GF}(p^d)$ versus \mathbb{Z}_{p^l} slots. The HE schemes based on the Ring-LWE problem (*Ring-HE schemes* in short), depend on arithmetic of cyclotomic integers [15]. Cyclotomic integers a are algebraic integers generated by some

primitive m -th root of unity ζ and have the form like $a = a_0 + a_1\zeta + \dots + a_{n-1}\zeta^{n-1}$ where a_i are ordinary integers in \mathbb{Z} and $n = \phi(m)$.

Generally, plaintexts in the Ring-HE schemes are encoded by cyclotomic integers modulo some *small prime* p . (Here, taking modulo p of cyclotomic integers a means taking modulo p of each coefficient a_i .) Then, what type of algebraic structure will a cyclotomic integer $a \bmod p$ have? Its structure is known to be a tuple of elements of Galois field $\text{GF}(p^d)$ of some degree d . For small primes p , this degree $d (> \log_p(m))$ will be large. Thus, in the Ring-HE schemes, a plaintext is a tuple of *plaintext slots* and each plaintext slot represents an element of Galois field $\text{GF}(p^d)$ of large degree d [17]. Addition or multiplication of plaintexts actually means addition or multiplication of each plaintext slots as elements of Galois field $\text{GF}(p^d)$.

Such plaintext structure is good for applications that use data represented by elements of Galois field $\text{GF}(p^d)$, such as error correcting codes or AES ciphers. However, many applications will use integers modulo p^l (i.e., elements in \mathbb{Z}_{p^l}) for some positive integer l (and especially for $p = 2$), rather than elements of Galois field $\text{GF}(p^d)$. By using the Hensel lifting technique, Ring-HE schemes can have plaintext slots of integers modulo p^l (as some applications do in fact) but with expense of efficiency. If we want to encrypt mod- p^l integer plaintexts on slots using Ring-HE schemes, actually we can use only 1-dimensional constant polynomials in each d -dimensional slots for homomorphic evaluation. As stated earlier, the dimension d would not be small ¹.

In this paper, we propose a novel HE scheme in which plaintext structure is inherently a tuple of integers modulo p^l (for some positive integer l), that is, each plaintext slot contains an integer in \mathbb{Z}_{p^l} , rather than an element of $\text{GF}(p^d)$. We believe in that such plaintext structure will be more natural, easy to handle, and significantly efficient for many applications such as outsourced data mining.

Method. To realize plaintext structure composed of slots of mod- p^l integers, we use decomposition ring $R_{\mathbb{Z}}$ with respect to the prime p , instead of cyclotomic ring R .

Let ζ be a primitive m -th root of unity. The m -th cyclotomic ring $R = \{a_0 + a_1\zeta + \dots + a_{n-1}\zeta^{n-1} \mid a_i \in \mathbb{Z}\}$ is a ring of all cyclotomic integers generated by ζ , where $n = \phi(m)$ is the value of Euler function at m . Plaintext space of Ring-HE schemes will be the space of mod- p cyclotomic integers, i.e., $R_p = R/pR$ for some small prime p . It is known that in the cyclotomic ring R , the prime number p is not prime any more (in general) and it factors into a product of g prime ideals \mathfrak{P}_i (with some divisor g of n): $pR = \mathfrak{P}_0\mathfrak{P}_1 \dots \mathfrak{P}_{g-1}$. The residual

¹ For instance, Lu, Kawasaki and Sakuma [13] uses the HELib with parameters $n = m - 1 = 27892$ and $p \approx 2^{36}$ to perform homomorphic computation needed for their statistical analysis on encrypted data in 110-bit security, that results in the plaintext space composed of $l \approx 70$ tuples of the Galois field $\text{GF}(p^d)$ of the degree $d = n/l \approx 398$. They are enforced to use only constant polynomials in those Galois fields.

fields R/\mathfrak{P}_i of each factor \mathfrak{P}_i are nothing but the space of plaintext slots of Ring-HE schemes, which are isomorphic to $\text{GF}(p^d)$ with $d = n/g$. Thus, the plaintext space is

$$R_p \simeq R/\mathfrak{P}_0 \oplus \cdots \oplus R/\mathfrak{P}_{g-1} \simeq \text{GF}(p^d) \oplus \cdots \oplus \text{GF}(p^d).$$

As stated before, we can use only 1-dimensional subspace $\text{GF}(p) = \mathbb{Z}_p$ in each d -dimensional slot $\text{GF}(p^d)$ for homomorphic evaluation as mod- p integers.

The decomposition ring R_Z with respect to prime p is the minimum subring of R in which the prime p has the same form of prime ideal factorization as in R , that is,

$$pR_Z = \mathfrak{p}_0\mathfrak{p}_1 \cdots \mathfrak{p}_{g-1} \quad (1)$$

with the same number g of factors. By the minimality of R_Z , the residual fields R_Z/\mathfrak{p}_i of each factor \mathfrak{p}_i must be one-dimensional, that is, isomorphic to \mathbb{Z}_p . So the plaintext space on R_Z will be

$$(R_Z)_p \simeq R_Z/\mathfrak{p}_0 \oplus \cdots \oplus R_Z/\mathfrak{p}_{g-1} \simeq \mathbb{Z}_p \oplus \cdots \oplus \mathbb{Z}_p.$$

Applying the Hensel lifting l times, we get $(R_Z)_q \simeq \mathbb{Z}_q \oplus \cdots \oplus \mathbb{Z}_q$ for $q = p^l$. Thus, the decomposition ring R_Z realizes plaintext slots of integers modulo $q = p^l$, as desired. Note that now we can use *all of the dimensions* of R_Z as its plaintext slots for mod- p^l integer plaintexts. This high parallelism of slot structure will bring us significantly more efficient SIMD operations for mod- p^l integer plaintexts.

Two bases. The cyclotomic ring R has attractive features that enable efficient implementation of addition/multiplication of and noise handling on their elements. Can we do the similar thing even if we use the decomposition ring R_Z instead of cyclotomic ring R ?

The cyclotomic ring R 's nice properties are consolidated to the existence of two types of bases [16]:

- The power(ful) basis: Composed of short and nearly orthogonal vectors to each other. Used when rounding rational cyclotomic numbers to their nearest cyclotomic integers.
- The CRT basis: Related to the FFT transformation and multiplication. Vectors of coefficients of given two cyclotomic integers w.r.t. the CRT basis can be multiplied component-wise, resulting a new vector corresponding to the multiplied cyclotomic integer.

We investigate structure of the decomposition ring R_Z , following the way in cyclotomic cases given by Lyubashevsky et al. [16]. Then, we will give two types of bases of R_Z , called η -basis and ξ -basis in this paper, which can substitute for the power(ful) and CRT bases in cyclotomic cases, respectively. The trace map from R to R_Z enables us to observe structure of R_Z as an image of the cyclotomic ring R , along with some particular phenomenon emerging from flatness of the decomposition ring (the degree $d = 1$). We also study noise growth occurred by algebraic manipulations (especially, by multiplication) of elements in R_Z , following [16].

Construction. Based on the above investigation, we construct our *subring homomorphic encryption scheme* that is an HE scheme over the decomposition ring $R_{\mathcal{Z}}$, or a realization of the FV scheme [6] over $R_{\mathcal{Z}}$. The construction is described concretely using the η -basis and ξ -basis above. We show several bounds on the noise growth occurred among homomorphic computations on its ciphertexts and prove that our HE scheme is fully homomorphic using ciphertext modulus of the magnitude $q = O(\lambda^{\log \lambda})$ with security parameter λ , as the FV scheme is so.

For security we will need hardness of a variant of the decisional Ring-LWE problem over the decomposition ring. Recall the search version of Ring-LWE problem is already proved to have a quantum polynomial time reduction from the approximate shortest vector problem of ideal lattices in *any number field* by Lyubashevsky et al. [15]. They proved equivalence between the search and decisional versions of the Ring-LWE problems only for cyclotomic rings. However, it is not difficult to see that the equivalence holds also over the decomposition rings, since they are subrings of cyclotomic rings and inherit good properties from them.

Implementation and benchmark. We implemented our subring homomorphic encryption scheme using the C++ language and performed several experiments with different parameters. Our benchmark results show that the η -basis and ξ -basis can substitute well for the power(ful) and CRT bases of cyclotomic rings, and indicate that our subring homomorphic encryption scheme is several times faster than HELib for *mod- p^l integer plaintexts*, due to its high parallelism of *mod- p^l slot structure*.

Organization. In Sect. 2 we prepare notions and tools needed for our work, especially about cyclotomic rings. Section 3 investigates structure and properties of the decomposition ring, and gives its η -basis and ξ -basis as well as quasi-linear time conversion between them. In Sect. 4 we state a variant of the Ring-LWE problem over the decomposition ring and construct our subring homomorphic encryption scheme over the decomposition ring. Finally, Sect. 5 shows our benchmark results, comparing efficiency of our implementation of subring homomorphic encryption scheme and HELib. Proofs of lemmas or theorems are collected in the appendices.

2 Preliminaries

Notation. For a positive integer m , \mathbb{Z}_m denotes the ring of congruent integers mod m , and \mathbb{Z}_m^* denotes its multiplicative subgroup. For an integer a (that is prime to m), $\text{ord}_m^{\times}(a)$ denotes the order of $a \in \mathbb{Z}_m^*$. Basically vectors are supposed to represent column vectors. The symbol $\mathbf{1}$ denotes a column vector with all entries equal to 1. I_n denotes the $n \times n$ identity matrix. The symbol $\text{Diag}(\alpha_1, \dots, \alpha_n)$ means a diagonal matrix with diagonals $\alpha_1, \dots, \alpha_n$. For vectors \mathbf{x}, \mathbf{y} ($\in \mathbb{C}^n$), $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^n x_i \bar{y}_i$ denotes the inner product of \mathbf{x} and \mathbf{y} . $\|\mathbf{x}\|_2 = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$ denotes the l_2 -norm of vector \mathbf{x} and $\|\mathbf{x}\|_{\infty} = \max_{i=1}^n \{|x_i|\}$ denotes the

infinity norm of \mathbf{x} . For vectors \mathbf{a} and \mathbf{b} , $\mathbf{a} \odot \mathbf{b} = (a_i b_i)_i$ denotes the component-wise product of \mathbf{a} and \mathbf{b} . For a square matrix M over \mathbb{R} , $s_1(M)$ denotes the largest singular value of M . For a matrix A over \mathbb{C} , $A^* = \overline{A}^T$ denotes the transpose of complex conjugate of A .

2.1 Homomorphic Encryption Scheme

A homomorphic encryption scheme is a quadruple $\text{HE} = (\text{Gen}, \text{Encrypt}, \text{Decrypt}, \text{Evaluate})$ of probabilistic polynomial time algorithms. Gen generates a public key pk , a secret key sk and an evaluation key evk : $(\text{pk}, \text{sk}, \text{evk}) \leftarrow \text{Gen}(1^\lambda)$. Encrypt encrypts a plaintext $x \in \mathsf{X}$ to a ciphertext c under a public key pk : $c \leftarrow \text{Encrypt}(\text{pk}, x)$. Decrypt decrypts a ciphertext c to a plaintext x using the secret key sk : $x \leftarrow \text{Decrypt}(\text{sk}, c)$. Evaluate applies a function $f : \mathsf{X}^l \rightarrow \mathsf{X}$ (given as an arithmetic circuit) to ciphertexts c_1, \dots, c_l and outputs a new ciphertext c_f using the evaluation key evk : $c_f \leftarrow \text{Evaluate}(\text{evk}, f, c_1, \dots, c_l)$.

A homomorphic encryption scheme HE is L -homomorphic for $L = L(\lambda)$ if for any function $f : \mathsf{X}^l \rightarrow \mathsf{X}$ given as an arithmetic circuit of depth L and for any l plaintexts $x_1, \dots, x_l \in \mathsf{X}$, it holds that

$$\text{Decrypt}_{\text{sk}}(\text{Evaluate}_{\text{evk}}(f, c_1, \dots, c_l)) = f(x_1, \dots, x_l)$$

for $c_i \leftarrow \text{Encrypt}_{\text{pk}}(x_i)$ ($i = 1, \dots, l$) except with a negligible probability (i.e., $\text{Decrypt}_{\text{sk}}$ is ring homomorphic). A homomorphic encryption scheme is called *fully homomorphic* if it is L -homomorphic for any polynomial function $L = \text{poly}(\lambda)$.

2.2 Gaussian Distributions and Subgaussian Random Variables

For a positive real $s > 0$, the n -dimensional (spherical) Gaussian function $\rho_s : \mathbb{R}^n \rightarrow (0, 1]$ is defined as

$$\rho_s(x) = \exp(-\pi \|x\|_2 / s^2).$$

It defines the continuous Gaussian distribution D_s with density $s^{-n} \rho_s(x)$.

A random variable X over \mathbb{R} is called *subgaussian with parameter s* ($s > 0$) if $\mathbb{E}[\exp(2\pi t X)] \leq \exp(\pi s^2 t^2)$ ($\forall t \in \mathbb{R}$). A random variable X over \mathbb{R}^n is called subgaussian with parameter s if $\langle X, u \rangle$ is subgaussian with parameter s for any unit vector $u \in \mathbb{R}^n$. A random variable X according to Gaussian distribution D_s is subgaussian with parameter s . A bounded random variable X (as $|X| \leq B$) with $\mathbb{E}[X] = 0$ is subgaussian with parameter $B\sqrt{2\pi}$.

A subgaussian random variable with parameter s satisfies the tail inequality:

$$\Pr[|X| \geq t] \leq 2 \exp\left(-\pi \frac{t^2}{s^2}\right) \quad (\forall t \geq 0). \quad (2)$$

2.3 Lattices

For n linearly independent vectors $B = \{b_j\}_{j=1}^n \subset \mathbb{R}^n$, $\Lambda = \mathcal{L}(B) = \left\{ \sum_{j=1}^n z_j b_j \mid z_j \in \mathbb{Z} \ (\forall j) \right\}$ is called an n -dimensional *lattice*. For a lattice $\Lambda \subset \mathbb{R}^n$, its *dual lattice* is defined by $\Lambda^\vee = \left\{ y \in \mathbb{R}^n \mid \langle x, y \rangle \in \mathbb{Z} \ (\forall x \in \Lambda) \right\}$. The dual lattice is itself a lattice. The dual of dual lattice is the same as the original lattice: $(\Lambda^\vee)^\vee = \Lambda$. For a countable subset $A \subset \mathbb{R}^n$, the sum $D_s(A) \stackrel{\text{def}}{=} \sum_{x \in A} D_s(x)$ is well-defined. The discrete Gaussian distribution $D_{\Lambda+c, s}$ on a (coset of) lattice Λ is defined by restricting the continuous Gaussian distribution D_s on the (coset of) lattice Λ :

$$D_{\Lambda+c, s}(x) \stackrel{\text{def}}{=} \frac{D_s(x)}{D_s(\Lambda+c)} \quad (x \in \Lambda+c).$$

2.4 Number Fields

A complex number $\alpha \in \mathbb{C}$ is called an *algebraic number* if it satisfies $f(\alpha) = 0$ for some nonzero polynomial $f(X) \in \mathbb{Q}[X]$ over \mathbb{Q} . For an algebraic number α , the monic and irreducible polynomial $f(X)$ satisfying $f(\alpha) = 0$ is uniquely determined and called the *minimum polynomial* of α . An algebraic number α generates a *number field* $K = \mathbb{Q}(\alpha)$ over \mathbb{Q} , which is isomorphic to $\mathbb{Q}[X]/(f(X))$, via $g(\alpha) \mapsto g(X)$. The dimension of K as a \mathbb{Q} -vector space is called the *degree* of K and denoted as $[K : \mathbb{Q}]$. By the isomorphism, $[K : \mathbb{Q}] = \deg f$.

An algebraic number α is called an *algebraic integer* if its minimum polynomial belongs to $\mathbb{Z}[X]$. All algebraic integers belonging to a number field $K = \mathbb{Q}(\alpha)$ constitutes a ring R , called an *integer ring* of K .

A number field $K = \mathbb{Q}(\alpha)$ has $n (= [K : \mathbb{Q}])$ isomorphisms ρ_i ($i = 1, \dots, n$) into the complex number field \mathbb{C} . The trace map $\text{Tr}_{K|\mathbb{Q}} : K \rightarrow \mathbb{Q}$ is defined by $\text{Tr}_{K|\mathbb{Q}}(a) = \sum_{i=1}^n \rho_i(a)$ ($\in \mathbb{Q}$). If all of the isomorphisms ρ_i induce automorphisms of K (i.e., $\rho_i(K) = K$ for any i), the field K is called a *Galois extension* of \mathbb{Q} and the set of isomorphisms $\text{Gal}(K|\mathbb{Q}) \stackrel{\text{def}}{=} \{\rho_1, \dots, \rho_n\}$ constitutes a group, called the *Galois group* of K over \mathbb{Q} . By the Galois theory, all subfields L of K and all subgroups H of $G = \text{Gal}(K|\mathbb{Q})$ corresponds to each other one-to-one:

$$\begin{aligned} L &\mapsto H = G_L = \{\rho \in G \mid \rho(a) = a \text{ for any } a \in L\} \\ &\quad : \text{the stabilizer group of } L \\ H &\mapsto L = K^H = \{a \in K \mid \rho(a) = a \text{ for any } \rho \in H\} \\ &\quad : \text{the fixed field by } H. \end{aligned}$$

Here, K is also a Galois extension of L with Galois group $\text{Gal}(K|L) = H$ (since any isomorphism (of K into \mathbb{C}) that fixes L sends K to K). Especially, $[K : L] = |H|$. The trace map of K over L is defined by $\text{Tr}_{K|L}(a) = \sum_{\rho \in H} \rho(a)$ ($\in L$) for $a \in K$.

2.5 Cyclotomic Fields and Rings

Let m be a positive integer. A primitive m -th root of unity $\zeta = \exp(2\pi\sqrt{-1}/m)$ has the minimum polynomial $\Phi_m(X) \in \mathbb{Z}[X]$ of degree $n = \phi(m)$ that belongs to $\mathbb{Z}[X]$, called the *cyclotomic polynomial*. Especially, ζ is an algebraic integer. A number field $K = \mathbb{Q}(\zeta)$ generated by ζ is called the m -th *cyclotomic field* and its elements are called *cyclotomic numbers*. The integer ring R of the cyclotomic field $K = \mathbb{Q}(\zeta)$ is known to be $R = \mathbb{Z}[\zeta] = \mathbb{Z}[X]/\Phi_m(X)$. In particular, as a \mathbb{Z} -module, R has a basis (called *power basis*) $\{1, \zeta, \dots, \zeta^{n-1}\}$, i.e., $R = \mathbb{Z} \cdot 1 + \mathbb{Z} \cdot \zeta + \dots + \mathbb{Z} \cdot \zeta^{n-1}$. The integer ring R is called the m -th *cyclotomic ring* and its elements are called *cyclotomic integers*. For a positive integer q , $R_q = R/qR = \mathbb{Z}_q[X]/\Phi_m(X)$ is a ring of *cyclotomic integers mod q* .

The cyclotomic field $K = \mathbb{Q}(\zeta)$ is a Galois extension over \mathbb{Q} since it has $n = [K : \mathbb{Q}]$ automorphisms ρ_i defined by $\rho_i(\zeta) = \zeta^i$ for $i \in \mathbb{Z}_m^*$. Its Galois group $G = \overline{\text{Gal}(K|\mathbb{Q})}$ is isomorphic to \mathbb{Z}_m^* by corresponding ρ_i to i . Note that $\rho_i(\bar{b}) = \overline{\rho_i(b)}$, since $\bar{a} = \rho_{-1}(a)$.

The trace of ζ for the prime index m is simple:

Lemma 1. *If the index m is prime, we have*

$$\text{Tr}_{K|\mathbb{Q}}(\zeta^i) = \begin{cases} m - 1 & (i \equiv 0 \pmod{m}) \\ -1 & (i \not\equiv 0 \pmod{m}). \end{cases}$$

Structure of R_p . Let p be a prime that does not divide m . Although the cyclotomic polynomial $\Phi_m(X)$ is irreducible over \mathbb{Z} , by taking mod p , it will be factored into a product of several polynomials $F_i(X)$'s:

$$\Phi_m(X) \equiv F_0(X) \cdots F_{g-1}(X) \pmod{p}, \tag{3}$$

where all of $F_i(X)$ are irreducible mod p , and have the same degree $d = \text{ord}_m^{\times}(p)$ which is a divisor of n . The number of factors is equal to $g = n/d$.

It is known that there are g prime ideals $\mathfrak{P}_0, \dots, \mathfrak{P}_{g-1}$ of R lying over p : $\mathfrak{P}_i \cap \mathbb{Z} = p\mathbb{Z}$ ($i = 0, \dots, g - 1$) and p decomposes into a product of those prime ideals in R :

$$pR = \mathfrak{P}_0 \cdots \mathfrak{P}_{g-1}. \tag{4}$$

This decomposition of the prime p reflects the factorization of $\Phi_m(X) \pmod{p}$ (Eq. (3)). In fact, each prime factor \mathfrak{P}_i is generated by p and $F_i(\zeta)$ as ideals of R , $\mathfrak{P}_i = (p, F_i(\zeta))$ for $i = 0, \dots, g - 1$. The corresponding residual fields are given by

$$R/\mathfrak{P}_i \simeq \mathbb{Z}_p[X]/F_i(X) \simeq \text{GF}(p^d)$$

for $i = 0, \dots, g - 1$. Thus, we have

$$R_p \simeq R/\mathfrak{P}_0 \oplus \cdots \oplus R/\mathfrak{P}_{g-1} \simeq \text{GF}(p^d) \oplus \cdots \oplus \text{GF}(p^d).$$

In the Ring-HE schemes such as [3, 4, 6], plaintexts are encoded by cyclotomic integers $x \in R_p$ modulo some *small prime* $p (\nmid m)$. By the factorization of R_p

above, g plaintexts x_0, \dots, x_{g-1} belonging to $\text{GF}(p^d)$ are encoded into a single cyclotomic integer $x \in R_p$. The place of each plaintext $x_i \in \text{GF}(p^d)$ is called a *plaintext slot*. Thus, in the Ring-HE schemes, one can encrypt g plaintexts into a single ciphertext by setting them on corresponding plaintext slots and can evaluate or decrypt the g encrypted plaintexts at the same time using arithmetic of cyclotomic integers [17]. Gentry et al. [8] homomorphically evaluates AES circuit on HE-encrypted AES-ciphertexts in the SIMD manner, using such plaintext slot structure for $p = 2$, which fits to the underlying $\text{GF}(2^d)$ -arithmetic of the AES cipher.

Geometry of numbers. Using the n automorphisms ρ_i ($i \in \mathbb{Z}_m^*$), the cyclotomic field K is embedded into an n -dimensional complex vector space $\mathbb{C}^{\mathbb{Z}_m^*}$, called the *canonical embedding* $\sigma : K \rightarrow H \subset \mathbb{C}^{\mathbb{Z}_m^*}$: $\sigma(a) = (\rho_i(a))_{i \in \mathbb{Z}_m^*}$. Its image $\sigma(K)$ is contained in the space H defined as

$$H \stackrel{\text{def}}{=} \{x \in \mathbb{C}^{\mathbb{Z}_m^*} : x_i = \bar{x}_{m-i} \quad (\forall i \in \mathbb{Z}_m^*)\}.$$

Since $H = B\mathbb{R}^n$ with the unitary matrix $B = \frac{1}{\sqrt{2}} \begin{pmatrix} I & \sqrt{-1}J \\ J & -\sqrt{-1}I \end{pmatrix}$, the space H is isomorphic to \mathbb{R}^n as an inner product \mathbb{R} -space (where J is the reversal matrix of the identity matrix I).

By the canonical embedding σ , one can regard R (or its (fractional) ideals of R) as lattices in the n -dimensional real vector space H , called *ideal lattices*. Inner products or norms of elements $a \in K$ are defined through the embedding σ :

$$\langle a, b \rangle \stackrel{\text{def}}{=} \langle \sigma(a), \sigma(b) \rangle = \text{Tr}_{K|\mathbb{Q}}(a\bar{b}), \quad \|a\|_2 \stackrel{\text{def}}{=} \|\sigma(a)\|_2, \quad \|a\|_\infty \stackrel{\text{def}}{=} \|\sigma(a)\|_\infty.$$

3 Decomposition Rings and Their Properties

To realize plaintext structure composed of slots of $\text{mod-}p^l$ integers for some small prime p , we use decomposition rings R_Z w.r.t. p instead of cyclotomic rings R .

3.1 Decomposition Field

Let $G = \text{Gal}(K|\mathbb{Q})$ be the Galois group of the m -th cyclotomic field $K = \mathbb{Q}(\zeta)$ over \mathbb{Q} . Let p be a prime that does not divide m . Recall such p has the prime ideal decomposition of Eq. (4). The *decomposition group* G_Z of K w.r.t. p is the subgroup of G defined as

$$G_Z \stackrel{\text{def}}{=} \{\rho \in G \mid \mathfrak{P}_i^\rho = \mathfrak{P}_i \quad (i = 0, \dots, g-1)\}.$$

That is, G_Z is the subgroup of automorphisms ρ of K that stabilize each prime ideal \mathfrak{P}_i lying over p . Recall the Galois group $G = \text{Gal}(K|\mathbb{Q})$ is isomorphic to \mathbb{Z}_m^* via ρ^{-1} . Since p does not divide m , $p \in \mathbb{Z}_m^*$. It is known that the decomposition group G_Z is generated by the automorphism ρ_p corresponding to the prime p ,

called the Frobenius map w.r.t. p : $G_Z = \langle \rho_p \rangle \simeq \langle p \rangle \subseteq \mathbb{Z}_m^*$. The order of G_Z is equal to $d = \text{ord}_m^\times(p)$. The fixed field $Z = K^{G_Z}$ by G_Z is called the *decomposition field* of K (w.r.t. p). The decomposition field Z can be characterized as the smallest subfield Z of K such that $\mathfrak{P}_i \cap Z$ does not split in K , so that the prime p factorizes into prime ideals in Z in much the same way as in K . By the Galois theory, $G_Z = \text{Gal}(K|Z)$. For degrees, we have $[K : Z] = |G_Z| = d$, $[Z : \mathbb{Q}] = n/d = g$. The decomposition field Z is itself the Galois extension of \mathbb{Q} and its Galois group $\text{Gal}(Z|\mathbb{Q}) = G/G_Z$ is given by $\text{Gal}(Z|\mathbb{Q}) \simeq \mathbb{Z}_m^*/\langle p \rangle$.

3.2 Decomposition Ring

The integer ring $R_Z = R \cap Z$ of the decomposition field Z is called the *decomposition ring*. Primes ideals over p in the decomposition ring R_Z are given by $\mathfrak{p}_i = \mathfrak{P}_i \cap Z$ for $i = 0, \dots, g-1$, and the prime p factors into the product of those prime ideals in much the same way as in K :

$$pR_Z = \mathfrak{p}_0 \cdots \mathfrak{p}_{g-1}. \tag{5}$$

This leads to the decomposition of $(R_Z)_p$: $(R_Z)_p \simeq R_Z/\mathfrak{p}_0 \oplus \cdots \oplus R_Z/\mathfrak{p}_{g-1}$.

For each prime ideal \mathfrak{P}_i (of R) lying over \mathfrak{p}_i , the Frobenius map ρ_p acts as the p -th power automorphism $\text{pow}_p(x) = x^p$ on R/\mathfrak{P}_i :

$$\begin{array}{ccc} R & \longrightarrow & R/\mathfrak{P}_i \\ \rho_p \downarrow & & \text{pow}_p \downarrow \\ R & \longrightarrow & R/\mathfrak{P}_i \end{array}$$

Then, by definition of $R_Z = R^{(\rho_p)}$, any element in R_Z/\mathfrak{p}_i must be fixed by pow_p , which means:

$$R_Z/\mathfrak{p}_i = (R/\mathfrak{P}_i)^{\langle \text{pow}_p \rangle} = \mathbb{Z}_p.$$

Thus, we see that all slots of $(R_Z)_p$ must be one-dimensional: $(R_Z)_p \simeq \mathbb{Z}_p \oplus \cdots \oplus \mathbb{Z}_p$.

By applying the Hensel lifting (w.r.t. p) l times to the situation, we get

$$qR_Z = \mathfrak{q}_0 \cdots \mathfrak{q}_{g-1} \tag{6}$$

$$(R_Z)_q \simeq \mathbb{Z}_q \oplus \cdots \oplus \mathbb{Z}_q \tag{7}$$

for $q = p^l$ with any positive integer l . This structure of the decomposition ring $(R_Z)_q$ brings us the plaintext structure of our subring homomorphic encryption scheme, being composed of $g \bmod q$ integer slots.

3.3 Bases of the Decomposition Ring R_Z

To construct homomorphic encryption schemes using some ring R , we will need two types of bases of the ring R over \mathbb{Z} , one for decoding elements in $R \otimes$

\mathbb{R} into its nearest element in R , and another one that enables FFT-like fast computations among elements in R . In addition, we also need some quasi-linear time transformations among vector representations with respect to the two types of bases. Here, *assuming the index m of cyclotomic ring R is prime*, we construct such two types of bases for the decomposition ring R_Z , following the cyclotomic ring case given by Lyubashevsky et al. [16].

The η -basis. Let m be a prime and $K = \mathbb{Q}(\zeta)$ be the m -th cyclotomic field. For a prime $p (\neq m)$, let Z be the decomposition field of K with respect to p .

Fix any set of representatives $\{t_0, \dots, t_{g-1}\}$ of $\mathbb{Z}_m^*/\langle p \rangle \simeq \text{Gal}(Z|\mathbb{Q})$. For $i = 0, \dots, g - 1$, define

$$\eta_i \stackrel{\text{def}}{=} \text{Tr}_{K|Z}(\zeta^{t_i}) = \sum_{a \in \langle p \rangle} \zeta^{t_i a} \quad (\in R_Z).$$

Lemma 2. For $i = 0, \dots, g - 1$, we have $\text{Tr}_{Z|\mathbb{Q}}(\eta_i) = \sum_{i=0}^{g-1} \eta_i = -1$, $\text{Tr}_{Z|\mathbb{Q}}(\bar{\eta}_i) = \sum_{i=0}^{g-1} \bar{\eta}_i = -1$.

Lemma 3. For the prime index m , the set $\{\eta_0, \dots, \eta_{g-1}\}$ is a basis of the decomposition ring R_Z (w.r.t. $p (\neq m)$) over \mathbb{Z} , i.e., $R_Z = \mathbb{Z}\eta_0 + \dots + \mathbb{Z}\eta_{g-1}$.

Definition 1. We call the basis $\boldsymbol{\eta} := (\eta_0, \dots, \eta_{g-1})$ η -basis of R_Z . For any $a \in R_Z$, there exists unique $\mathbf{a} \in \mathbb{Z}^g$ satisfying $a = \boldsymbol{\eta}^T \mathbf{a}$. We call such $\mathbf{a} \in \mathbb{Z}^g$ η -vector of $a \in R_Z$.

The ξ -basis. By the choice of t_i 's, the Galois group $\text{Gal}(Z|\mathbb{Q})$ of Z is given by

$$\text{Gal}(Z|\mathbb{Q}) = \{\rho_{t_0}, \dots, \rho_{t_{g-1}}\}.$$

Elements $a \in Z$ in the decomposition field are regarded as g -dimensional \mathbb{R} -vectors through the canonical embedding $\sigma_Z : Z \rightarrow H_Z (\subset \mathbb{C}^{\mathbb{Z}_m^*/\langle p \rangle})$ defined as $\sigma_Z(a) = (\rho_i(a))_{i \in \mathbb{Z}_m^*/\langle p \rangle}$. The g -dimensional \mathbb{R} -subspace H_Z is as

$$H_Z \stackrel{\text{def}}{=} \{x \in \mathbb{C}^{\mathbb{Z}_m^*/\langle p \rangle} : x_i = \bar{x}_{m-i} \quad (\forall i \in \mathbb{Z}_m^*/\langle p \rangle)\}.$$

Define a $g \times g$ matrix Ω_Z over R_Z as

$$\Omega_Z = \left(\rho_{t_i}(\eta_j) \right)_{0 \leq i, j < g} \quad (\in R_Z^{g \times g}).$$

Note that each column of Ω_Z is the canonical embedding $\sigma_Z(\eta_j)$ of η_j . Since the index m is prime, the Galois group $\text{Gal}(Z|\mathbb{Q})$ is cyclic and we can take the representatives $\{t_0, \dots, t_{g-1}\}$ so that $t_j \equiv t^j \pmod{\langle p \rangle}$ with some $t \in \mathbb{Z}_m^*$ for $j = 0, \dots, g - 1$. Setting $\eta = \text{Tr}_{K|Z}(\zeta)$, for any i and j ,

$$\rho_{t_i}(\eta_j) = \rho_{t_i}(\rho_{t_j}(\eta)) = \rho_{t_i \cdot t_j}(\eta) = \rho_{t_{i+j}}(\eta) = \eta_{i+j}.$$

In particular, Ω_Z is symmetric. We can show that:

Lemma 4. $\Omega_Z^* \Omega_Z = (\text{Tr}_{Z|\mathbb{Q}}(\overline{\eta}_i \eta_j))_{0 \leq i, j < g} = mI_g - d\mathbf{1} \cdot \mathbf{1}^T \in \mathbb{Z}^{g \times g}$.

Corollary 1. *The set $\{m^{-1}(\eta_0 - d), \dots, m^{-1}(\eta_{g-1} - d)\}$ is the dual basis of conjugate η -basis $\{\overline{\eta}_0, \dots, \overline{\eta}_{g-1}\}$, i.e. for any $0 \leq i, j < g$,*

$$\text{Tr}_{Z|\mathbb{Q}}\left(\frac{\eta_i - d}{m} \cdot \overline{\eta}_j\right) = \delta_{ij}.$$

In particular, $R_Z^\vee = \mathbb{Z} \frac{\eta_0 - d}{m} + \dots + \mathbb{Z} \frac{\eta_{g-1} - d}{m}$.

Define a $g \times g$ matrix Γ_Z over Z as

$$\Gamma_Z \stackrel{\text{def}}{=} \left(\rho_{t_i}\left(\frac{\overline{\eta}_j - d}{m}\right)\right)_{0 \leq i, j < g} \in Z^{g \times g}.$$

Corollary 1 means that $\overline{\Gamma}_Z^T \overline{\Omega}_Z = I$. Since Ω_Z is symmetric,

$$\Gamma_Z \Omega_Z = \Omega_Z \Gamma_Z = I. \tag{8}$$

Lemma 5. *For any $\mathbf{b} = \Omega_Z \mathbf{a}$, we have*

$$\mathbf{a} = \Gamma_Z \mathbf{b} = \frac{1}{m} \left(\overline{\Omega}_Z \mathbf{b} - d \left(\sum_j b_j \right) \cdot \mathbf{1} \right).$$

Let r be a positive integer and $q = p^r$. Let $\mathfrak{q} = \mathfrak{q}_0$ be the first ideal that appears in the factorization of qR_Z (Eq. (6)). Recall that $R_Z/\mathfrak{q} \simeq \mathbb{Z}_q$.

Let

$$\Omega_Z^{(q)} \stackrel{\text{def}}{=} \Omega_Z \text{ mod } \mathfrak{q} \in (R_Z/\mathfrak{q})^{g \times g} \simeq \mathbb{Z}_q^{g \times g}$$

Since $p \nmid m$, $\Gamma_Z \text{ mod } \mathfrak{q}$ is well-defined and by Eq. (8), $\Omega_Z^{(q)}$ is invertible mod \mathfrak{q} .

Definition 2. *Define $\boldsymbol{\xi} = (\xi_0, \dots, \xi_{g-1}) \in (R_Z/\mathfrak{q})^g$ by $\boldsymbol{\eta}^T \equiv \boldsymbol{\xi}^T \Omega_Z^{(q)} \pmod{\mathfrak{q}}$. We call the basis $\boldsymbol{\xi}$ of (R_Z/\mathfrak{q}) over \mathbb{Z}_q $\boldsymbol{\xi}$ -basis of R_Z (with respect to \mathfrak{q}). For any $a \in (R_Z/\mathfrak{q})$, there exists unique $\mathbf{b} \in \mathbb{Z}_q^g$ satisfying that $a = \boldsymbol{\xi}^T \mathbf{b}$. We call such $\mathbf{b} \in \mathbb{Z}_q^g$ as $\boldsymbol{\xi}$ -vector of $a \in (R_Z/\mathfrak{q})$.*

Lemma 6. *For any $a \in R_Z$ it holds that*

$$\begin{aligned} a \equiv \boldsymbol{\eta}^T \cdot \mathbf{a} &\Leftrightarrow a \equiv \boldsymbol{\xi}^T \cdot (\Omega_Z^{(q)} \cdot \mathbf{a}) \pmod{\mathfrak{q}} \\ a &= \boldsymbol{\eta}^T \cdot \mathbf{a} \Leftrightarrow \sigma_Z(a) = \Omega_Z \mathbf{a} \\ a \equiv \boldsymbol{\xi}^T \cdot \mathbf{b} \pmod{\mathfrak{q}} &\Leftrightarrow \sigma_Z(a) \equiv \mathbf{b} \pmod{\mathfrak{q}} \end{aligned}$$

Lemma 7. *If $a_1 = \boldsymbol{\xi}^T \cdot \mathbf{b}_1$ and $a_2 = \boldsymbol{\xi}^T \cdot \mathbf{b}_2$, then $a_1 a_2 = \boldsymbol{\xi}^T \cdot (\mathbf{b}_1 \odot \mathbf{b}_2)$.*

3.4 Conversion Between η - and ξ -vectors

Resolution of unity in $R_Z \bmod \mathfrak{q}$. By Hensel-lifting the factorization of $\Phi_m(X) \bmod p$ (Eq. (3)) to modulus $q = p^r$, we get factorization of $\Phi_m(X) \bmod q$: $\Phi_m(X) \equiv \overline{F}_0(X) \cdots \overline{F}_{g-1}(X) \pmod{q}$. Here, note that the number g of irreducible factors and the degree d of each factors remain unchanged in the lifting. According to this factorization, the ideal qR of R is factored as $qR = \mathfrak{Q}_0 \cdots \mathfrak{Q}_{g-1}$ with ideals $\mathfrak{Q}_i = (q, \overline{F}_i(\zeta))$ of R .

For each $i = 0, \dots, g-1$, let $G_i(X) \stackrel{\text{def}}{=} \prod_{j \neq i} \overline{F}_j(X) \pmod{q}$ and $P_i(X) \stackrel{\text{def}}{=} (G_i(X)^{-1} \bmod (q, \overline{F}_i(X))) \cdot G_i(X) \pmod{q}$. It is verified that the set $\{\tau_i = P_i(\zeta)\}_{i=0}^{g-1}$ constitutes a *resolution of unity* in $R \bmod q$, i.e.

$$\tau_i \equiv \begin{cases} 1 & \pmod{\mathfrak{Q}_i} \quad (i = 0, \dots, g-1) \\ 0 & \pmod{\mathfrak{Q}_j} \quad (j \neq i) \end{cases}$$

and it satisfies that

$$\sum_{i=0}^{g-1} \tau_i \equiv 1, \quad \tau_i^2 \equiv \tau_i, \quad \tau_i \tau_j \equiv 0 \pmod{q} \quad (j \neq i).$$

By the Chinese remainder theorem, the resolution of unity $\{\tau_i\}_{i=0}^{g-1}$ is uniquely determined $\bmod qR$. In the following we take coefficients of each τ_i from $[-q/2, q/2)$ over the basis $B' = \{\zeta, \zeta^2, \dots, \zeta^{m-1}\}$ of R .

Lemma 8. *For any $0 \leq i < g$ it is that $\tau_i \in R_Z$, and $\{\tau_i\}_{i=0}^{g-1}$ is also a resolution of unity in $R_Z \bmod q$.*

Using the resolution of unity $\{\tau_i\}_{i=0}^{g-1}$ in R_Z , we can compute $a_i \in \mathbb{Z}_q$ satisfying $a \equiv a_i \pmod{\mathfrak{q}_i}$ given $a \in R_Z$, as follows:

$$a \bmod \mathfrak{q}_i = a \tau_i \bmod q = a_i \tau_i \bmod q \quad \xrightarrow{\text{dividing by } \tau_i} a_i.$$

Computation of $\Omega_Z^{(q)}$. Now we can compute the matrix $\Omega_Z^{(q)} = \left(\eta_{i+j} \bmod \mathfrak{q} \right)_{0 \leq i, j < g}$ ($\in \mathbb{Z}_q^{g \times g}$) by computing the entities η_{i+j} in Ω_Z as cyclotomic integers and reducing them modulo \mathfrak{q} ($= \mathfrak{q}_0$) using the resolution of unity $\{\tau_i\}_{i=0}^{g-1}$. Since the matrix $\Omega_Z^{(q)}$ has cyclic structure (the $(i+1)$ -th row is a left shift of the i -th row), it is sufficient to compute its first row. Here, we remark that once we have computed the matrix $\Omega_Z^{(q)}$, we can totally forget the original structure of cyclotomic ring R , and all we need is doing various computations among η - and ξ -vectors (of elements in R_Z) with necessary conversion between them using the matrix $\Omega_Z^{(q)}$.

Computation of $\mathbf{b} = \Omega_Z^{(q)} \cdot \mathbf{a}$. To convert η -vector \mathbf{a} of an element $a = \boldsymbol{\eta}^T \cdot \mathbf{a} \in R_Z$ to its corresponding ξ -vector \mathbf{b} (satisfying $a = \boldsymbol{\xi}^T \cdot \mathbf{b}$), by Lemma 6, we need to compute a matrix-vector product $\mathbf{b} = \Omega_Z^{(q)} \cdot \mathbf{a}$. By Lemma 5, the inverse

conversion from ξ -vector \mathbf{b} to its corresponding η -vector $\mathbf{a} = \Gamma_Z \cdot \mathbf{b}$ also can be computed using a similar matrix-vector product $\overline{\Omega}_Z^{(q)} \cdot \mathbf{b}$. Here, $\overline{\Omega}_Z^{(q)} \stackrel{\text{def}}{=} \overline{\Omega}_Z \bmod \mathfrak{q}$.

By definition of $\Omega_Z^{(g)}$, the j -th component b_j of the product $\mathbf{b} = \Omega_Z^{(g)} \cdot \mathbf{a}$ is $b_j = \sum_{i=0}^{g-1} a_i \eta_{i+j}$ (where indexes are mod g and we omit mod \mathfrak{q}). This means that \mathbf{b} is the convolution product of vector $\boldsymbol{\eta}$ and the reversal vector $(a_0, a_{g-1}, a_{g-2}, \dots, a_1)$ of \mathbf{a} , where $\boldsymbol{\eta} = (\eta_i)_{i=0}^{g-1}$ is the first row of $\Omega_Z^{(g)}$.

Define two polynomials $f(X) = \sum_{i=0}^{g-1} \eta_i X^i$ and $g(X) = a_0 + \sum_{i=1}^{g-1} a_{g-i} X^i$ over \mathbb{Z}_q . Since \mathbf{b} is the convolution product of $\boldsymbol{\eta}$ and the reversal vector of \mathbf{a} , it holds that $f(X)g(X) = \sum_{i=0}^{g-1} b_i X^i \pmod{X^g - 1}$. The polynomial product $f(X)g(X) \pmod{X^g - 1}$ can be computed in quasi-linear time $\tilde{O}(g)$ using the FFT multiplication. Thus, we know that conversions between η -vectors \mathbf{a} and ξ -vectors \mathbf{b} can be done in quasi-linear time $\tilde{O}(g)$.

4 Subring Homomorphic Encryption

Now we construct an HE scheme using the decomposition ring R_Z , *subring homomorphic encryption* scheme.

4.1 The Ring-LWE Problem on the Decomposition Ring

For security of our subring homomorphic encryption scheme, we will need hardness of a variant of the decisional Ring-LWE problem over the decomposition ring. Let m be a prime. Let R_Z be the decomposition ring of the m -th cyclotomic ring R with respect to some prime $p (\neq m)$. Let q be a positive integer. For an element $s \in R_Z$ and a distribution χ over R_Z , define a distribution $A_{s,\chi}$ on $(R_Z)_q \times (R_Z)_q$ as follows: First choose an element a uniformly from $(R_Z)_q$ and sample an element e according to the distribution χ . Then return the pair $(a, b = as + e \bmod q)$.

Definition 3 (The decisional Ring-LWE problem on the decomposition ring). *Let q, χ be as above. The R-DLWE $_{q,\chi}$ problem on the decomposition ring R_Z asks to distinguish samples from $A_{s,\chi}$ with $s \stackrel{\text{u}}{\leftarrow} \mathbb{Z}_q$ and (the same number of) samples uniformly chosen from $(R_Z)_q \times (R_Z)_q$.*

Recall the search version of Ring-LWE problem is already proved to have a quantum polynomial time reduction from the approximate shortest vector problem of ideal lattices in *any number field* by Lyubashevsky et al. [15]. They proved equivalence between the search and the decisional versions of the Ring-LWE problems only for cyclotomic rings. The key of their proof of equivalence is the existence of prime modulus q for Ring-LWE problem which totally decomposes into n prime ideal factors: $qR = \mathfrak{Q}_0 \cdots \mathfrak{Q}_{n-1}$. (Their residual fields R/\mathfrak{Q}_i have polynomial order q and we can guess the solution of the Ring-LWE problem modulo ideal \mathfrak{Q}_i , and then we can verify validity of the guess using the assumed oracle for the decisional Ring-LWE problem.) Since the decomposition ring R_Z is a subring of the cyclotomic ring R , such modulus q totally decomposes into

g prime ideals also in the decomposition ring R_Z : $qR_Z = \mathfrak{q}_0 \cdots \mathfrak{q}_{g-1}$. Using this decomposition, the proof of equivalence by [15] holds also over the decomposition rings R_Z , essentially as it is.

4.2 Parameters

Let m be a prime index of cyclotomic ring R . Choose a (small) prime p , distinct from m . Let $d = \text{ord}_m^\times(p)$ be the multiplicative order of $p \bmod m$, and $g = (m-1)/d$ be the degree of the decomposition ring R_Z of R with respect to p . Take two powers of p , $q = p^r$ and $t = p^l$ ($r > l$) as ciphertext and plaintext modulus, respectively. Set the quotient as $\Delta = q/t = p^{r-l}$. Choose two distributions χ_{key} and χ_{err} over \mathbb{Z}^g .

4.3 Encoding Methods and Basic Operations of Elements in R_Z

Basically, we use η -vectors $\mathbf{a} \in \mathbb{Z}^g$ to encode elements $a = \eta^T \cdot \mathbf{a}$ in R_Z . To multiply two elements encoded by η -vectors \mathbf{a} and \mathbf{b} modulo $q = p^r$, first we convert those η -vectors to corresponding ξ -vectors modulo q . We can multiply resulting ξ -vectors component-wise, and then re-convert the result into its corresponding η -vector modulo q . The functions `eta_to_xi` and `xi_to_eta` use the matrix $\Omega_Z^{(q)}$ computed in advance. $(\eta_i)_{i=0}^{g-1}$ denotes the first row of $\Omega_Z^{(q)}$.

<pre> mult_eta ($\mathbf{a}, \mathbf{b}, q$) : $\alpha = \text{eta_to_xi}(\mathbf{a}, q)$ $\beta = \text{eta_to_xi}(\mathbf{b}, q)$ $\gamma_i = \alpha_i \beta_i \bmod q$ ($i = 0, \dots, g-1$) return $\mathbf{c} = \text{xi_to_eta}(\boldsymbol{\gamma}, q)$ </pre>	<pre> eta_to_xi (\mathbf{a}, q) : $a(X) = a_0 + \sum_{i=1}^{g-1} a_{g-i} X^i$ $c(X) = \sum_{i=0}^{g-1} \eta_i X^i$ $b(X) = a(X)c(X) \bmod (q, X^g - 1)$ return $\mathbf{b} = (b_0, \dots, b_{g-1})$ </pre>
<pre> xi_to_eta (\mathbf{b}, q) : $b(X) = b_0 + \sum_{i=1}^{g-1} b_{g-i} X^i$ $c(X) = \sum_{i=0}^{g-1} \bar{\eta}_i X^i$ $a(X) = b(X)c(X) \bmod (q, X^g - 1)$ $t = b_0 + \dots + b_{g-1} \bmod q$ return $\mathbf{a} = (m^{-1}(a_i - dt) \bmod q)_{i=0}^{g-1}$ </pre>	

We regard plaintext vectors $\mathbf{m} \in \mathbb{Z}_t^g$ as ξ -vectors of corresponding elements $m_\xi = \boldsymbol{\xi}^T \mathbf{m} \in (R_Z)_t$. By Lemma 7 their products $m_\xi m'_\xi \in (R_Z)_t$ encodes the plaintext vector $\mathbf{m} \odot \mathbf{m}' \in \mathbb{Z}_t^g$. For a fixed integer base w , let $l_w = \lfloor \log_w(q) \rfloor + 1$. Any vector $\mathbf{a} \in \mathbb{Z}_q^g$ can be written as $\mathbf{a} = \sum_{k=0}^{l_w-1} w^k \mathbf{a}_k$ with vectors $\mathbf{a}_k \in \mathbb{Z}_w^g$ of small entries. Define $\text{WD}(\mathbf{a}) \stackrel{\text{def}}{=} (\mathbf{a}_k)_{k=0}^{l_w-1} (\in (\mathbb{Z}_w^g)^{l_w})$.

4.4 Scheme Description

Our subring homomorphic encryption scheme is a realization of the FV scheme by Fan and Vercauteren [6], using the decomposition ring R_Z . Here we describe

its symmetric key version. The public key version is easily derived as like in the FV and other HE schemes.

SecretKeyGen (λ) : $s \leftarrow \chi_{key}$ return $sk = s \in \mathbb{Z}^g$	Encrypt ($sk = s \in \mathbb{Z}^g, m \in \mathbb{Z}_t^g$) : $a \xleftarrow{u} \mathbb{Z}_q^g, e \leftarrow \chi_{err}, n = \text{xi_to_eta}(m, t)$ $b = \text{mult_eta}(a, s, q) + \Delta n + e \bmod q$ return $ct = (a, b)$
Decrypt ($sk = s \in \mathbb{Z}^g, ct = (a, b)$): $n = \left\lfloor \frac{1}{\Delta} (b - \text{mult_eta}(a, s, q) \bmod q) \right\rfloor$ $m = \text{eta_to_xi}(n, t)$ return m	Add ($ct_1 = (a_1, b_1), ct_2 = (a_2, b_2)$): $a = a_1 + a_2 \bmod q,$ $b = b_1 + b_2 \bmod q$ return $ct = (a, b)$
EvaluateKeyGen (s) : $\gamma = \text{mult_eta}(s, s, q)$ For $k = 0$ to $l_w - 1$: $\alpha_k \xleftarrow{u} \mathbb{Z}_q^g, x_k \leftarrow \chi_{err}, \beta_k = \text{mult_eta}(\alpha_k, s, q) + w^k \gamma + x_k \bmod q$ return $ev = ((\alpha_k, \beta_k))_{k=0}^{l_w-1}$	
Mult ($ct_1 = (a_1, b_1), ct_2 = (a_2, b_2), ev = ((\alpha_k, \beta_k))_k$) : $e = \left\lfloor \frac{1}{\Delta} \cdot \text{mult_eta}(b_1, b_2, q^2/t) \right\rfloor,$ $c = \left\lfloor \frac{1}{\Delta} \cdot (\text{mult_eta}(a_1, b_2, q^2/t) + \text{mult_eta}(a_2, b_1, q^2/t)) \right\rfloor,$ $d = \left\lfloor \frac{1}{\Delta} \cdot \text{mult_eta}(a_1, a_2, q^2/t) \right\rfloor, (d_0, \dots, d_{l_w-1}) = \text{WD}(d)$ $a = c + \sum_{k=0}^{l_w-1} \text{mult_eta}(d_k, \alpha_k, q) \bmod q,$ $b = e + \sum_{k=0}^{l_w-1} \text{mult_eta}(d_k, \beta_k, q) \bmod q$ return $ct = (a, b)$	

It is straightforward to see:

Theorem 1. *The subring homomorphic encryption scheme is indistinguishably secure under the chosen plaintext attack if the $\text{R-DLWE}_{q, \chi_{key}, \chi_{err}}$ problem on the decomposition ring R_Z is hard.*

For correctness we have the following theorem. (The proof is in Sect. A.3)

Theorem 2. *The subring homomorphic encryption scheme will be fully homomorphic under circular security assumption (i.e., an encryption of secret key s does not leak any information about s) by taking ciphertext modulus $q = O(\lambda^{\log \lambda})$.*

5 Benchmark Results

We implemented our subring homomorphic encryption scheme (SR-HE in short) using the C++ language and performed several experiments using different parameters, comparing efficiency of our implementation of SR-HE and homomorphic encryption library HELib by Halevi and Shoup [10], which is based on the BGV scheme [4]. For notation of parameters, see Sect. 4.2.

As common parameters, we choose four values of prime m so that the m -th cyclotomic ring R will have as many number of plaintext slots (i.e., large g and small d values) as possible. The plaintext modulus $t = 2^l$ is fixed as $l = 8$. The noise parameter $s_{err} = \sqrt{2\pi}\sigma_{err}$ is fixed as $\sigma_{err} = 3.2$. The ciphertext modulus $q = 2^r$ is chosen as small as possible so that it enables homomorphic evaluation of exponentiation by 2^8 (i.e., $\text{Enc}(\mathbf{s}, \mathbf{m})^{2^8}$) with respect to each implementation. Table 1 summarizes the chosen parameters.

Table 1. Chosen parameters.

	m	g	d	l	r (SR-HE)	r (HElib)
par-127	127	18	7	8	162	135
par-8191	8191	630	13	8	210	250
par-43691	43691	1285	34	8	234	256
par-131071	131071	7710	17	8	242	-

Assuming that there is no special attack utilizing the particular algebraic structure of involving rings, corresponding security parameters λ are estimated using the lwe-estimator-9302d4204b4f by [1, 2].

Table 2 shows timing results for HElib in milliseconds on Intel Celeron(R) CPU G1840 @ 2.80 GHz 2. (We could not perform the test for par-131071 due to shortage of memory.) The secret key is chosen uniformly random among binary vectors of Hamming weight 64 over the power basis (default of HElib) and we encrypt g number of mod- 2^l integer plaintexts into a single HElib ciphertext using plaintext slots. As seen in Sect. 2.5, HElib (based on the BGV scheme) basically realizes $GF(2^d)$ arithmetic in each of g slots. If we want to encrypt mod- 2^l integer plaintexts on slots and to homomorphically evaluate on them, we can use only 1-dimensional constant polynomials in each $d(= m/g)$ -dimensional slots. This should cause certain waste in time and space. In fact, for example, timings for par-43691 ($g = 1285$) is much larger than two times of those for par-8191 ($g = 630$). This indicates that the HElib scheme cannot handle many mod- 2^l integer slots with high parallelism. So, to encrypt large number of mod- 2^l integer plaintexts using HElib, we have no choice but to prepare many ciphertexts, each of which encrypts a divided set of small number of plaintexts on their slots.

On the other hand, Table 3 shows timing results (also in milliseconds on Intel Celeron(R) CPU G1840 @ 2.80 GHz 2) for our SR-HE scheme. The secret key is chosen uniformly random among binary vectors of Hamming weight 64 over η -basis and we encrypt g number of mod- 2^l integer plaintexts into a single SR-HE ciphertext. As seen, timings are approximately linear with respect to the numbers of slots g . This shows that our SR-HE scheme can handle many mod- 2^l slots with high parallelism, as expected. We can encrypt large number of mod- 2^l integer plaintexts into a single SR-HE ciphertext using mod- 2^l slots without waste, and can homomorphically compute on them with high parallelism.

Table 2. Timing results of HElib on mod- 2^l integer plaintexts.

	λ	Enc	Dec	Add	Mult	Exp-by- 2^8
par-127	26	0.23	0.18	0.00	0.66	4.78
par-8191	92	30.45	210.77	0.84	107.53	512.64
par-43691	237	268.00	5158.44	4.74	634.69	4187.81
par-131071	–	–	–	–	–	–

Table 3. Timing results of SR-HE on mod- 2^l integer plaintexts.

	λ	Enc	Dec	Add	Mult	Exp-by- 2^8
par-127	–	0.14	0.12	0.00	0.57	4.47
par-8191	29	7.39	7.37	0.03	39.43	318.65
par-43691	32	17.38	17.19	0.11	92.14	741.42
par-131071	91	104.33	103.93	0.97	574.44	4620.22

Then, which is faster to encrypt many number of mod- 2^l integer plaintexts between the following two cases?

- (1) A single SR-HE ciphertext with many plaintext slots.
- (2) Many HElib ciphertexts with small number of plaintext slots.

The result for par-131071 of Table 3 shows we can encrypt 7710 mod- 2^l integer slots in a single SR-HE ciphertext with security parameter $\lambda = 91$ with timing:

$$(104.33, 103.93, 0.97, 574.44, 4620.22)$$

On a while, the result for par-8191 of Table 2 shows we can encrypt the same number of 7710 mod- 2^l integer slots using $\lceil 7710/630 \rceil = 13$ ciphertexts with security parameter $\lambda = 92$. The 13 times of the line par-8191 of Table 2 is

$$(395.85, 2740.01, 10.92, 1397.89, 6664.32).$$

Thus, our benchmark results indicate that Case (1) (a single SR-HE ciphertext with many slots) is significantly faster than Case (2) (many HElib ciphertexts with small number of plaintext slots) under equivalent security parameters.

Acknowledgments. This work was supported by JST CREST Grant Number JPMJCR1503. This work is further supported by the JSPS KAKENHI Grant Number 17K05353.

A Appendices

A.1 Proofs of Lemma

Proof of Lemma 2. $\text{Tr}_{Z|\mathbb{Q}}(\eta_i) = \text{Tr}_{Z|\mathbb{Q}}(\text{Tr}_{K|Z}(\zeta^{t_i})) = \text{Tr}_{K|\mathbb{Q}}(\zeta^{t_i})$. So, by Lemma 1, $\text{Tr}_{Z|\mathbb{Q}}(\eta_i) = -1$ for any i . Similarly, $\text{Tr}_{Z|\mathbb{Q}}(\bar{\eta}_i) = \text{Tr}_{Z|\mathbb{Q}}(\text{Tr}_{K|Z}(\zeta^{-t_i})) = \text{Tr}_{K|\mathbb{Q}}(\zeta^{-t_i}) = -1$. \square

Proof of Lemma 3. Since the index m is prime, the cyclotomic ring R has a basis $B = \{1, \zeta, \dots, \zeta^{m-2}\}$ over \mathbb{Z} . Since ζ is a unit of R , $B' := \zeta B = \{\zeta, \zeta^2, \dots, \zeta^{m-1}\}$ is also a basis of R over \mathbb{Z} . The fixing group $G_Z = \langle \rho_p \rangle$ of Z acts on B' and decomposes it into g orbits $\zeta^{t_i \langle p \rangle} = \{\zeta^{t_i}, \zeta^{t_i p}, \dots, \zeta^{t_i p^{d-1}}\}$ ($i = 0, \dots, g-1$). An element $z = \sum_{i=1}^{m-1} z_i \zeta^i \in R_Z$ that is stable under the action of G_Z must have constant integer coefficients over the each orbits $\zeta^{t_i \langle p \rangle}$. Hence, z is a \mathbb{Z} -linear combination of $\{\eta_1, \dots, \eta_g\}$. \square

Proof of Lemma 4. For $0 \leq i, j < g$,

$$\begin{aligned} \bar{\eta}_i \eta_j &= \left(\sum_{a \in \langle p \rangle} \zeta^{-at_i} \right) \left(\sum_{b \in \langle p \rangle} \zeta^{bt_j} \right) = \sum_{a, b \in \langle p \rangle} \zeta^{-at_i + bt_j} = \sum_{a \in \langle p \rangle} \sum_{b \in \langle p \rangle} \rho_a(\zeta^{-t_i + ba^{-1}t_j}) \\ &= \sum_{a \in \langle p \rangle} \sum_{b \in \langle p \rangle} \rho_a(\zeta^{-t_i + bt_j}) = \sum_{b \in \langle p \rangle} \text{Tr}_{K|Z}(\zeta^{-t_i + bt_j}). \end{aligned}$$

Here, Suppose $i \neq j$. Then, $-t_i + bt_j \not\equiv 0 \pmod{m}$ for any $b \in \langle p \rangle$. Hence, by Lemma 1,

$$\text{Tr}_{Z|\mathbb{Q}}(\bar{\eta}_i \eta_j) = \sum_{b \in \langle p \rangle} \text{Tr}_{K|\mathbb{Q}}(\zeta^{-t_i + bt_j}) = |\langle p \rangle| \cdot (-1) = -d.$$

If $i = j$, since $\text{Tr}_{K|\mathbb{Q}}(\zeta^{-t_i + bt_i}) = m - 1$ only if $b = 1$ and -1 otherwise by Lemma 1,

$$\text{Tr}_{Z|\mathbb{Q}}(\bar{\eta}_i \eta_i) = \sum_{b \in \langle p \rangle} \text{Tr}_{K|\mathbb{Q}}(\zeta^{-t_i + bt_i}) = m - 1 + (d - 1) \cdot (-1) = m - d \quad \square$$

Proof of Corollary 1. For any i , by Lemmas 2 and 4 we have

$$\text{Tr}_{Z|\mathbb{Q}}\left(\frac{\eta_i - d}{m} \cdot \bar{\eta}_i\right) = \frac{1}{m}(m - d) - \frac{d}{m} \cdot (-1) = 1.$$

Similarly, for any $i \neq j$ we have

$$\text{Tr}_{Z|\mathbb{Q}}\left(\frac{\eta_i - d}{m} \cdot \bar{\eta}_j\right) = \frac{-d}{m} - \frac{d}{m} \cdot (-1) = 0 \quad \square$$

Proof of Lemma 5

$$\begin{aligned} \mathbf{a} &= \Gamma_Z \mathbf{b} = \left(\rho_{t_i} \left(\frac{\bar{\eta}_j - d}{m} \right) \right)_{ij} \mathbf{b} = \left(\frac{1}{m} \sum_j \rho_{t_i} (\bar{\eta}_j - d) b_j \right)_i \\ &= \frac{1}{m} \left(\sum_j \rho_{t_i} (\bar{\eta}_j) b_j - d \sum_j b_j \right)_i = \frac{1}{m} \left(\bar{\Omega}_Z \mathbf{b} - d \left(\sum_j b_j \right) \cdot \mathbf{1} \right) \quad \square \end{aligned}$$

Proof of Lemma 6. The first claim is the definition of $\boldsymbol{\xi}$.

Since $\Omega_Z = \left(\sigma_Z(\eta_j) \right)_{0 \leq j < g}$, $\mathbf{a} = \boldsymbol{\eta}^T \cdot \mathbf{a}$ if and only if $\sigma_Z(\mathbf{a}) = \Omega_Z \mathbf{a}$.

Next,

$$\begin{aligned} a = \boldsymbol{\xi}^T \cdot \mathbf{b} &\Leftrightarrow a \equiv \boldsymbol{\eta}^T (\Omega_Z^{(g)})^{-1} \cdot \mathbf{b} \pmod{\mathfrak{q}} \\ &\Leftrightarrow \sigma_Z(a) \equiv \Omega_Z (\Omega_Z^{(g)})^{-1} \cdot \mathbf{b} \equiv \mathbf{b} \pmod{\mathfrak{q}} \end{aligned} \quad \square$$

Proof of Lemma 7. $\sigma_Z(a_1 a_2) = \sigma_Z(a_1) \odot \sigma_Z(a_2) = \mathbf{b}_1 \odot \mathbf{b}_2$ □

Proof of Lemma 8. The ideal qR_Z factors in R_Z as

$$qR_Z = \mathfrak{q}_0 \mathfrak{q}_1 \cdots \mathfrak{q}_{g-1}$$

where $\mathfrak{q}_i = \mathfrak{Q}_i \cap R_Z$ for any i .

Let $\{\tau'_i\}_{i=0}^{g-1}$ be a resolution of unity in $R_Z \pmod{q}$. Here, we take the coefficients of each τ'_i from $[-q/2, q/2)$ over the η -basis $\{\eta_0, \dots, \eta_{g-1}\}$ of R_Z .

Then,

$$\tau'_i \equiv \begin{cases} 1 & \pmod{\mathfrak{q}_i} \quad (i = 0, \dots, g-1) \\ 0 & \pmod{\mathfrak{q}_j} \quad (j \neq i). \end{cases}$$

Since $\mathfrak{q}_i \subset \mathfrak{Q}_i$ for any i , $\{\tau'_i\}_{i=0}^{g-1}$ is also a resolution of unity in $R \pmod{q}$. Since the coefficients of each τ'_i over the η -basis are in $[-q/2, q/2)$, by definition of $\eta_i = \sum_{a \in \langle p \rangle} \zeta^{t_i a}$, their coefficients over the basis B' are trivially also in $[-q/2, q/2)$. Hence, by the uniqueness of resolution, it must be that $\tau'_i = \tau_i$ for all i . □

A.2 Norms on the Decomposition Ring

Norms of $a \in Z$ are defined by

$$\|a\|_2 \stackrel{\text{def}}{=} \|\sigma_Z(a)\|_2, \quad \|a\|_\infty \stackrel{\text{def}}{=} \|\sigma_Z(a)\|_\infty.$$

Lemma 9. *For any $a, b \in Z$, we have*

$$\|ab\|_\infty \leq \|a\|_\infty \cdot \|b\|_\infty.$$

Proof. $\|ab\|_\infty = \|\sigma_Z(ab)\|_\infty = \|\sigma_Z(a) \odot \sigma_Z(b)\|_\infty \leq \|\sigma_Z(a)\|_\infty \cdot \|\sigma_Z(b)\|_\infty = \|a\|_\infty \cdot \|b\|_\infty.$ □

In the following, \mathbf{a} means the η -vector of given $a = \boldsymbol{\eta}^T \cdot \mathbf{a} \in R_Z$.

Lemma 10. (1) *For any $a \in Z$, $\|a\|_2 \leq \sqrt{m} \|\mathbf{a}\|_2$.*

(2) *For any $\mathbf{a} \in \mathbb{R}^g$, $\|\mathbf{a}\|_2 \leq \|a\|_2$.*

(3) *If $\mathbf{a} \in \mathbb{R}^g$ is far from being proportional to vector $\mathbf{1}$ (far from constants in short), we have $\|\mathbf{a}\|_2 \approx \frac{1}{\sqrt{m}} \|a\|_2$.*

Proof. (1) By Lemma 6, $\sigma_Z(a) = \Omega_Z \mathbf{a}$ and by Lemma 4

$$\Omega_Z^* \Omega_Z = mI_g - d\mathbf{1} \cdot \mathbf{1}^T.$$

The right-hand side matrix has eigenvalues $g - 1$ times of m and 1 with corresponding eigenvectors $(1, -1, 0, \dots, 0)$, $(1, 0, -1, 0, \dots, 0)$, \dots , $(1, 0, \dots, 0, -1)$, $(1, 1, \dots, 1)$. So, the symmetric matrix $\Omega_Z^* \Omega_Z$ can be diagonalized to $\text{Diag}(m, \dots, m, 1)$ by an orthogonal transformation, and we have $s_1(\Omega_Z) = \sqrt{m}$. This means $\|\mathbf{a}\|_2 \leq \sqrt{m} \|\mathbf{a}\|_2$. (2), (3) Conversely, $\mathbf{a} = (\Omega_Z)^{-1} \sigma_Z(a) = \Gamma_Z \sigma_Z(a)$. Similarly as above, the matrix $\Gamma_Z^* \Gamma_Z$ can be diagonalized to $\text{Diag}(1/m, \dots, 1/m, 1)$ by the orthogonal transformation. Hence, $s_1(\Gamma_Z) = 1$ and $\|\mathbf{a}\|_2 \leq \|a\|_2$. Since almost all of the eigenvalues of $\Gamma_Z^* \Gamma_Z$ are $1/m$, except 1 for eigenvector $(1, 1, \dots, 1)$, if \mathbf{a} is far from being proportional to the eigenvector $(1, 1, \dots, 1)$, $\|\mathbf{a}\|_2 \approx \frac{1}{\sqrt{m}} \|a\|_2$. \square

Lemma 11. (1) For any $a \in Z$, $\|\mathbf{a}\|_\infty \leq \sqrt{mg} \|\mathbf{a}\|_\infty$.

(2) For any $\mathbf{a} \in \mathbb{R}^g$, $\|\mathbf{a}\|_\infty \leq \sqrt{g} \|\mathbf{a}\|_\infty$.

(3) If a is far from constants, we have $\|\mathbf{a}\|_\infty \lesssim \sqrt{g/m} \|\mathbf{a}\|_\infty$.

Proof. (1) By Lemma 10-(1), $\|\mathbf{a}\|_\infty \leq \|a\|_2 \leq \sqrt{m} \|\mathbf{a}\|_2 \leq \sqrt{mg} \|\mathbf{a}\|_\infty$.

(2) By Lemma 10-(2), $\|\mathbf{a}\|_\infty \leq \|\mathbf{a}\|_2 \leq \|a\|_2 \leq \sqrt{g} \|\mathbf{a}\|_\infty$.

(3) By Lemma 10-(3), $\|\mathbf{a}\|_\infty \leq \|\mathbf{a}\|_2 \approx \frac{1}{\sqrt{m}} \|a\|_2 \leq \sqrt{g/m} \|\mathbf{a}\|_\infty$. \square

Subgaussian elements. We call a random variable $a \in Z$ *subgaussian* with parameter s if corresponding random variable $\sigma_Z(a)$ on H_Z is subgaussian with parameter s .

Lemma 12 (Claim 2.1, Claim 2.4 [16]). Let a_i be independent subgaussian random variables over Z with parameter s_i ($i = 1, 2$). Then,

1. The sum $a_1 + a_2$ is subgaussian with parameter $\sqrt{s_1^2 + s_2^2}$.

2. For any a_2 fixed, the product $a_1 \cdot a_2$ is subgaussian with parameter $\|a_2\|_\infty s_1$.

Lemma 13. Let \mathbf{a} be a subgaussian random variable over \mathbb{R}^g of parameter s . Then, $a = \boldsymbol{\eta}^T \cdot \mathbf{a}$ is subgaussian over Z of parameter \sqrt{ms} .

Proof. By Lemma 6 $\sigma_Z(a) = \Omega_Z \mathbf{a}$. As seen in the proof of Lemma 10, $s_1(\Omega_Z) = \sqrt{m}$. Hence, $\sigma_Z(a)$ is subgaussian of parameter \sqrt{ms} . \square

A.3 Correctness of Our Subring Homomorphic Encryption Scheme

Let \mathcal{X}_{key} and \mathcal{X}_{err} be discrete Gaussian distributions over \mathbb{Z}^g of parameters s_{key} and s_{err} , respectively. In the following, vectors \mathbf{a} , \mathbf{b} , \dots mean corresponding η -vectors of elements $a = \boldsymbol{\eta}^T \cdot \mathbf{a}$, $b = \boldsymbol{\eta}^T \cdot \mathbf{b}$, \dots in the decomposition ring R_Z , respectively.

Definition 4. The inherent noise term e of ciphertext $ct = (\mathbf{a}, \mathbf{b})$ designed for $\mathbf{m} \in \mathbb{Z}_t^g$ is an element $e \in R_Z$ with the smallest norm $\|e\|_\infty$ satisfying that

$$b - as = \Delta m_\xi + e + q\alpha$$

for some $\alpha \in R_Z$, secret key $\mathbf{sk} = \mathbf{s}$, and $m_\xi = \boldsymbol{\xi}^T \cdot \mathbf{m} \in R_Z$.

By definition, a ciphertext $ct = (\mathbf{a}, \mathbf{b}) \leftarrow \text{Encrypt}(\mathbf{s}, \mathbf{m})$ has $e = \boldsymbol{\eta}^T \cdot \mathbf{e}$ as an inherent noise term designed for \mathbf{m} with $\mathbf{e} \leftarrow \chi_{err}$. By Lemma 13, e is subgaussian of parameter $\sqrt{m}s_{err}$ and by the tail inequality (Eq. 2), $\|e\|_\infty \leq \omega(\sqrt{\log \lambda})\sqrt{m}s_{err}$ with an overwhelming probability.

Define $B_{correct} \stackrel{\text{def}}{=} \frac{\sqrt{m}}{2\sqrt{g}}\Delta$.

Lemma 14 (Noise bound for correctness). *Let e be the inherent noise term of ciphertext $ct = (\mathbf{a}, \mathbf{b})$ designed for $\mathbf{m} \in \mathbb{Z}_t^g$. If $\|e\|_\infty < B_{correct}$ (i.e. if $\frac{\sqrt{g}}{\sqrt{m}}\|e\|_\infty < \frac{1}{2}\Delta$), then decryption works correctly, i.e. $\text{Decrypt}(\mathbf{s}, ct) = \mathbf{m}$.*

Proof. By definition of the inherent noise term, \mathbf{a} and \mathbf{b} satisfy that

$$\frac{1}{\Delta}(b - as - \alpha q) = m_\xi + \frac{e}{\Delta}. \quad (9)$$

By Lemma 11-(3),

$$\left\| \frac{e}{\Delta} \right\|_\infty < \sqrt{g/m} \cdot \left\| \frac{e}{\Delta} \right\|_\infty \leq \sqrt{g/m} \cdot \frac{\sqrt{m}}{2\sqrt{g}} = \frac{1}{2}.$$

Hence, the η -vector of the left-hand side of Eq. (9) rounds to \mathbf{n} satisfying that $\boldsymbol{\eta}^T \cdot \mathbf{n} = m_\xi = \boldsymbol{\xi}^T \cdot \mathbf{m}$. \square

Lemma 15 (Noise bound for Add). *Let e_1 and e_2 be inherent noise terms of ciphertexts $ct_1 = (\mathbf{a}_1, \mathbf{b}_1)$ and $ct_2 = (\mathbf{a}_2, \mathbf{b}_2)$ designed for \mathbf{m}_1 and $\mathbf{m}_2 \in \mathbb{Z}_t^g$, respectively. Let e be the inherent noise term of $ct = \text{Add}(ct_1, ct_2)$ designed for $\mathbf{m}_1 + \mathbf{m}_2 \in \mathbb{Z}_t^g$. Then,*

$$\|e\|_\infty \leq \|e_1\|_\infty + \|e_2\|_\infty.$$

Lemma 16 (Noise bound for linearization). *Let $ev = ((\boldsymbol{\alpha}_k, \boldsymbol{\beta}_k))_{k=0}^{l_w-1} \leftarrow \text{EvaluateKeyGen}(\mathbf{s})$ be an evaluation key for a secret key $\mathbf{sk} = \mathbf{s}$. Suppose that a triple of elements e, c, d in R_Z satisfies*

$$e - cs + ds^2 \equiv \Delta m_\xi + x \pmod{q}$$

with $m_\xi = \boldsymbol{\xi}^T \cdot \mathbf{m}$ and some $x \in R_Z$ bounded as $\|x\|_\infty \leq B$. Let $(\mathbf{d}_0, \dots, \mathbf{d}_{l_w-1}) = \text{WD}(\mathbf{d})$. Then, for $a = c + \sum_{k=0}^{l_w-1} d_k \alpha_k$ and $b = e + \sum_{k=0}^{l_w-1} d_k \beta_k$, the pair $ct = (\mathbf{a}, \mathbf{b})$ constitutes a ciphertext that has an inherent noise term y designed for \mathbf{m} bounded as

$$\|y\|_\infty \leq B + \omega(\sqrt{\log \lambda})\sqrt{l_w m g w s_{err}}.$$

Proof. By definition of EvaluateKeyGen , the k -th pair $(\boldsymbol{\alpha}_k, \boldsymbol{\beta}_k)$ of ev has an inherent noise term x_k designed for $w^k s^2$, which is subgaussian of parameter $\sqrt{m}s_{err}$. Then,

$$\begin{aligned}
 b - as &\equiv \left(e + \sum_{k=0}^{l_w-1} d_k \beta_k \right) - \left(c + \sum_{k=0}^{l_w-1} d_k \alpha_k \right) s \equiv e - cs + \sum_{k=0}^{l_w-1} d_k (\beta_k - \alpha_k s) \\
 &\equiv e - cs + \sum_{k=0}^{l_w-1} d_k (w^k s^2 + x_k) \equiv e - cs + ds^2 + \sum_{k=0}^{l_w-1} d_k x_k \\
 &\equiv \Delta m_\xi + x + \sum_{k=0}^{l_w-1} d_k x_k \pmod{q}.
 \end{aligned}$$

We estimate $\|y\|_\infty$ for $y := x + \sum_{k=0}^{l_w-1} d_k x_k$. First by Lemma 11 (1), $\|d_k\|_\infty \leq \sqrt{mg} \|\mathbf{d}_k\|_\infty \leq \sqrt{mg} w$. Then, by Lemma 12, $d_k x_k$ are independently subgaussian of parameter $\|d_k\|_\infty s_{err} \leq \sqrt{mg} w s_{err}$, and $\sum_{k=0}^{l_w-1} d_k x_k$ is subgaussian of parameter $\sqrt{l_w} \sqrt{mg} w s_{err}$. Hence,

$$\|y\|_\infty \leq \|x\|_\infty + \left\| \sum_{k=0}^{l_w-1} d_k x_k \right\| \leq B + \omega(\sqrt{\log \lambda}) \sqrt{l_w} \sqrt{mg} w s_{err}. \quad \square$$

Lemma 17 (Noise bound for Mult). *Let e_1 and e_2 be inherent noise terms of ciphertexts $ct_1 = (\mathbf{a}_1, \mathbf{b}_1)$ and $ct_2 = (\mathbf{a}_2, \mathbf{b}_2)$ designed for \mathbf{m}_1 and $\mathbf{m}_2 \in \mathbb{Z}_t^g$, respectively. Suppose $\|e_i\|_\infty \leq B (< B_{correct})$ for $i = 1, 2$. Let f be the inherent noise term of $ct = \text{Mult}(ct_1, ct_2)$ designed for $\mathbf{m}_1 \odot \mathbf{m}_2 \in \mathbb{Z}_t^g$. Then,*

$$\|f\|_\infty \leq t\omega(\sqrt{\log \lambda}) \sqrt{mg} s_{key} \cdot B + \omega(\sqrt{\log \lambda}) \sqrt{l_w} \sqrt{mg} w s_{err}.$$

Proof. We prepare two claims.

Claim. Let $e_0 = \frac{1}{\Delta} b_1 b_2$, $c_0 = \frac{1}{\Delta} (a_1 b_2 + a_2 b_1)$, $d_0 = \frac{1}{\Delta} a_1 a_2$. Then,

$$e_0 - c_0 s + d_0 s^2 \equiv \Delta m_\xi + x \pmod{q}$$

with $m_\xi = (m_1)_\xi (m_2)_\xi$ and some $x \in R_Z$ bounded as

$$\|x\|_\infty \leq t\omega(\sqrt{\log \lambda}) \sqrt{mg} s_{key} \cdot B.$$

Proof. By assumption,

$$b_i - a_i s = \Delta (m_i)_\xi + x_i + \alpha_i q \quad (i = 1, 2) \quad (10)$$

with $\|x_i\|_\infty < B$. By Lemma 12 the product $a_i s$ is subgaussian of parameter $\|a_i\|_\infty s_{key} \leq \sqrt{mg} \|a_i\|_\infty s_{key} \leq \sqrt{mg} q s_{key}$. So, $\alpha_i = \lfloor (b_i - a_i s) / q \rfloor$ is bounded as

$$\|\alpha_i\|_\infty \leq \omega(\sqrt{\log \lambda}) \sqrt{mg} s_{key}.$$

By taking product of the two equations (10), we get

$$\begin{aligned}
 e_0 - c_0 s + d_0 s^2 &= \frac{1}{\Delta} \left(b_1 b_2 - (a_1 b_2 + a_2 b_1) s + a_1 a_2 s^2 \right) \\
 &= \Delta (m_1)_\xi (m_2)_\xi + x + qv
 \end{aligned}$$

with some $v \in R_Z$, where

$$x = (m_1)_\xi x_2 + (m_2)_\xi x_1 + \frac{1}{\Delta} x_1 x_2 + t(x_1 \alpha_2 + x_2 \alpha_1).$$

By Lemmas 9 and 11,

$$\begin{aligned} \|(m_i)_\xi x_j\|_\infty &\leq \|(m_i)_\xi\|_\infty \|x_j\|_\infty = \sqrt{mg} \|\mathbf{n}_i\|_\infty \|x_j\|_\infty \leq \sqrt{mgt} B \\ \left\| \frac{1}{\Delta} x_1 x_2 \right\|_\infty &\leq \frac{1}{\Delta} \|x_1\|_\infty \|x_2\|_\infty \leq \frac{1}{\Delta} B_{correct} \cdot \|x_2\|_\infty \leq \frac{\sqrt{m}}{2\sqrt{g}} \cdot B \\ \|tx_i \alpha_j\|_\infty &\leq t \|x_i\|_\infty \|\alpha_j\|_\infty \leq t B \omega(\sqrt{\log \lambda}) \sqrt{mgs_{key}}. \end{aligned}$$

Hence, x is bounded as

$$\begin{aligned} \|x\|_\infty &\leq 2\sqrt{mgt} B + \frac{\sqrt{m}}{2\sqrt{g}} \cdot B + 2\sqrt{mgt} B \omega(\sqrt{\log \lambda}) \sqrt{mgs_{key}} \\ &= (2\sqrt{mgt} + \frac{\sqrt{m}}{2\sqrt{g}} + 2t\omega(\sqrt{\log \lambda}) \sqrt{mgs_{key}}) B \\ &= t\omega(\sqrt{\log \lambda}) \sqrt{mgs_{key}} \cdot B \end{aligned} \quad \square$$

Claim. Let $e = \begin{bmatrix} e_0 \end{bmatrix}$, $c = \begin{bmatrix} c_0 \end{bmatrix}$, $d = \begin{bmatrix} d_0 \end{bmatrix}$. Then,

$$e - cs + ds^2 \equiv e_0 - c_0 s + d_0 s^2 + y \pmod{q}$$

with some $y \in R_Z$ bounded as

$$\|y\|_\infty \leq \omega(\log \lambda) \sqrt{mgs_{key}^2}.$$

Proof. Let $y = (e - e_0) - (c - c_0)s + (d - d_0)s^2 \pmod{q}$.

Using Lemma 11 (1), $\|e - e_0\|_\infty \leq \sqrt{mg} \|e - e_0\|_\infty \leq \sqrt{mg}/2$.

Similarly, $\|c - c_0\|_\infty \leq \sqrt{mg}/2$ and by Lemma 9, $\|(c - c_0)s\|_\infty \leq \|c - c_0\|_\infty \|s\|_\infty \leq \sqrt{mg}/2 \cdot \omega(\sqrt{\log \lambda}) s_{key} = \omega(\sqrt{\log \lambda}) \sqrt{mgs_{key}}$. Similarly, $\|(d - d_0)s^2\|_\infty \leq \omega(\log \lambda) \sqrt{mgs_{key}^2}$.

Thus,

$$\begin{aligned} \|y\|_\infty &\leq \|e - e_0\|_\infty + \|(c - c_0)s\|_\infty + \|(d - d_0)s^2\|_\infty \\ &\leq \sqrt{mg}/2 + \omega(\sqrt{\log \lambda}) \sqrt{mgs_{key}} + \omega(\log \lambda) \sqrt{mgs_{key}^2} \\ &\leq \omega(\log \lambda) \sqrt{mgs_{key}^2} \end{aligned} \quad \square$$

By the two claims we know that

$$e - cs + ds^2 \equiv \Delta m_\xi + z \pmod{q}$$

with $z = x + y$ bounded as

$$\begin{aligned} \|z\|_\infty &\leq \|x\|_\infty + \|y\|_\infty \leq t\omega(\sqrt{\log \lambda})\sqrt{mgs_{key}} \cdot B + \omega(\log \lambda)\sqrt{mgs_{key}^2} \\ &\leq t\omega(\sqrt{\log \lambda})\sqrt{mgs_{key}} \cdot B. \end{aligned}$$

Finally, applying Lemma 16 to our situation, we know that **Mult** will output a ciphertext $ct = (\mathbf{a}, \mathbf{b})$ that has an inherent noise term f designed for $m_\xi = (m_1)_\xi(m_2)_\xi$, satisfying that

$$\begin{aligned} \|f\|_\infty &\leq \|z\|_\infty + \omega(\sqrt{\log \lambda})\sqrt{l_w m g w s_{err}} \\ &\leq t\omega(\sqrt{\log \lambda})\sqrt{mgs_{key}} \cdot B + \omega(\sqrt{\log \lambda})\sqrt{l_w m g w s_{err}}. \quad \square \end{aligned}$$

Proof of Theorem 2. By Lemma 14, a ciphertext ct that encrypts plaintext \mathbf{m} can be correctly decrypted if its inherent noise term e designed for \mathbf{m} satisfies that

$$\frac{\sqrt{g}}{\sqrt{m}} \|e\|_\infty < \frac{1}{2} \Delta = \frac{q}{2t}.$$

By Lemma 17, by one multiplication, $\frac{\sqrt{g}}{\sqrt{m}}$ times of infinity norm of noises under input ciphertexts increases $\log_2(t\omega(\sqrt{\log \lambda})gs_{key}) = O(\log \lambda)$ bits. Hence, to correctly evaluate an arithmetic circuit over \mathbb{Z}_t^g with L levels of multiplications, it suffices that

$$\log q > L \log \lambda.$$

By Lemma 4 of [3], we can implement **Decrypt** algorithm by some circuit of level $L_{dec} = O(\log \lambda)$. Hence by taking $q = O(\lambda^{\log \lambda})$, the subring homomorphic encryption scheme can homomorphically evaluate its own **Decrypt** circuit and will be fully homomorphic under circular security assumption. \square

References

1. Albrecht, M.R.: On dual lattice attacks against small-secret LWE and parameter choices in HELib and SEAL. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10211, pp. 103–129. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56614-6_4
2. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. *J. Math. Cryptol.* **9**(3), 169–203 (2015)
3. Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical GapSVP. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 868–886. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_50
4. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. In: ITCS, pp. 309–325. ACM (2012)

5. Cheon, J.H., Kim, M., Lauter, K.: Homomorphic computation of edit distance. In: Brenner, M., Christin, N., Johnson, B., Rohloff, K. (eds.) FC 2015. LNCS, vol. 8976, pp. 194–212. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48051-9_15
6. Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. IACR Cryptology ePrint Archive, 2012–144 (2012)
7. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC, pp. 169–178. ACM (2009)
8. Gentry, C., Halevi, S., Smart, N.P.: Homomorphic evaluation of the AES circuit. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 850–867. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_49
9. Graepel, T., Lauter, K., Naehrig, M.: ML confidential: machine learning on encrypted data. In: Kwon, T., Lee, M.-K., Kwon, D. (eds.) ICISC 2012. LNCS, vol. 7839, pp. 1–21. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37682-5_1
10. Halevi, S., Shoup, V.: Algorithms in HELib. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 554–571. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_31
11. Liu, J., Li, J., Xu, S., Fung, B.C.M.: Secure outsourced frequent pattern mining by fully homomorphic encryption. In: Madria, S., Hara, T. (eds.) DaWaK 2015. LNCS, vol. 9263, pp. 70–81. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-22729-0_6
12. Lauter, K., López-Alt, A., Naehrig, M.: Private computation on encrypted genomic data. In: Aranha, D.F., Menezes, A. (eds.) LATINCRYPT 2014. LNCS, vol. 8895, pp. 3–27. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16295-9_1
13. Lu, W., Kawasaki, S., Sakuma, J.: Using fully homomorphic encryption for statistical analysis of categorical ordinal and numerical data. In: Network and Distributed System Security Symposium (NDSS), February 2017
14. Laine, K., Player, R.: Simple Encrypted Arithmetic Library - SEAL (v2.0). Technical report, Microsoft Research, MSR-TR-2016-52, September 2016
15. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_1
16. Lyubashevsky, V., Peikert, C., Regev, O.: A toolkit for ring-LWE cryptography. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 35–54. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38348-9_3
17. Smart, N.P., Vercauteren, F.: Fully homomorphic SIMD operations. Des. Codes Crypt. **71**(1), 57–81 (2014)

Side Channel Analysis and Implementation



Novel Leakage Against Realistic Masking and Shuffling Countermeasures

Case Study on PRINCE and SEED

Yoo-Seung Won¹, Aesun Park¹, and Dong-Guk Han^{1,2(✉)}

¹ Department of Financial Information Security, Kookmin University, Seoul, Korea
{mathwys87,aesons,chrsta}@kookmin.ac.kr

² Department of Information Security, Cryptology, and Mathematics,
Kookmin University, Seoul, Korea

Abstract. It is often considered reasonable to combine first-order Boolean masking and shuffling countermeasures. However, shuffling countermeasures can sometimes be applied only to some rounds to improve performance. Herein, we define combinations of partial shuffling and masking countermeasures as **restricted shuffling and masking countermeasures**.

Moreover, we propose a novel leakage on restricted shuffling and masking countermeasures that have low attack complexity and a small correlation-reduction factor. Our novel leakage ignores the confusion layer to prevent shuffling from increasing the attack complexity. To reduce the complexity, we can confirm a partial correlation between the diffusion and confusion layer outputs. We identify that our proposal, which exploits this fact offers an overwhelming advantage compared with existing attacks when applied to the PRINCE and SEED block ciphers. Furthermore, we demonstrate the effectiveness of our proposed scheme using both simulated and realistic traces. In simulations, the number of traces required was reduced by up to 95%. When attacking a realistic device, a few traces were enough to recover the correct key, although existing attacks failed to reveal the correct key.

Keywords: Shuffling · Masking

Second-order correlation power analysis · PRINCE · SEED

1 Introduction

Higher-order Boolean masking [4, 5] is computationally very expensive, whereas shuffling [6, 12] is usually significantly less costly when applied to non-linear layers [12]. Moreover, first-order masking can be defeated with only modest effort [10]. For these reasons, when implementing practical cryptographic algorithms that have to be secure against CPA attacks, it is sometimes considered reasonable to combine first-order Boolean masking and shuffling [6, 12, 16].

Many of the devices, used in IoT environments are very restricted in terms of computing power and memory capacity. For these reasons, it is impractical to

fully prevent CPA attacks using perfect countermeasures as it is impossible in practice to implement cryptographic algorithms to defend against CPA attacks using full masking and shuffling. Thus, typical cryptographic algorithm implementations utilize more practical countermeasures to prevent CPA attacks they use more secure countermeasures for parts wherein the complexity of guessing the key is low and less secure countermeasures for other parts wherein the complexity of guessing the key is significantly higher. For example, the SEED block cipher algorithm [13] can apply first-order Boolean masking [7] and shuffling schemes on G-functions over one or two rounds. At times, however, depending on availability, first-order Boolean masking scheme can be selectively applied over three rounds.

Especially, in the financial IC card of the Republic of Korea, the SEED block cipher is used as encrypting a password. Moreover, to construct the countermeasure implemented in software, the first-order Boolean masking and shuffling countermeasures is sometimes employed.

In fact, it is well-known that these countermeasures are secure against second-order CPA (SOCPA) attacks on confusion layer outputs (*e.g.*, S-box outputs), which are the usual targets for CPA attacks. However, we must still investigate whether other potential target, in addition to confusion layer leaks, exist for SOCPA attacks if these secure cryptographic algorithms are to be used in practice.

Contributions. Some block ciphers, such as SEED [13] and PRINCE [1], utilize bitwise AND operations in a diffusion layer after the confusion layer. When they adopt first-order Boolean masking countermeasures in the diffusion layer, they only utilize bitwise AND and bitwise XOR operations, as in [7]. Moreover, to protect against SOCPA attacks in practice, first-order Boolean masking with partial shuffling is sometimes employed [6] to improve performance; these are defined as the restricted shuffling and first-order Boolean masking countermeasures in this paper. Although the confusion and diffusion layers are shuffled, the diffusion layer outputs are loaded in order to compute next round.

In this paper, we propose a novel leakage on restricted shuffling and first-order Boolean masking countermeasures that have low attack complexity and a smaller correlation-reduction factor. It reduces the complexity of guessing the key from $(2^n)^L$ (for a SOCPA attack) to 2^{2n} , where n indicates the number of blocks and L specifies the number of linear operations required to compute a single block of the diffusion layer. In the case of PRINCE and SEED, the complexity of guessing the key from 2^{16} and 2^{32} can be reduced to 2^8 and 2^{16} , respectively.

Our main target is the diffusion layer outputs, which are composed of bitwise AND and XOR operations. In addition, a partial correlation we prove between the diffusion and confusion layer outputs will facilitate realistic recovery of the correct key. We demonstrate that in simulations, our proposal can reduce the number of traces required by up to 95%. We also confirm that in a more realistic scenario, only a few traces can be enough to retrieve the correct key although the existing SOCPA schemes fail to directly reveal it.

2 Preliminaries

In this section, we define some necessary terminology and symbols, and describe the algorithm used for the restricted shuffling and first-order Boolean masking countermeasures. Then, we discuss the vulnerability in the confusion layer output.

2.1 Notation

Let X denote a random variable comprising T elements X_i , represented as follows:

$$X = X_0 \parallel X_1 \parallel X_2 \parallel \cdots \parallel X_{T-2} \parallel X_{T-1}$$

where X_i is an n -bit string and \parallel indicates concatenation. We also use tildes, *i.e.*, \widetilde{X} , to represent the application of masking. R indicates the total number of cipher rounds. The functions $\kappa(\kappa_m)$, γ , and λ represent respectively key addition (mask addition), confusion, and diffusion layer shuffling, where the shuffle order is generated by the random permutation φ . Here, φ^0 is the identity function, *i.e.*, it specifies that the shuffling countermeasure is not adopted.

2.2 Restricted Shuffling and Masking Countermeasures

In order to implement block ciphers, such as AES or SEED, so that they are secure against power analysis attacks, countermeasures may be required. As discussed above, there are two approaches to protecting cryptographic algorithms against power analysis attacks. Combine masking and shuffling is considered to render effective protection for cryptographic algorithms. In [6], Herbst *et al.* applied first-order Boolean masking and partial shuffling to AES for the use in smart card implementations.

Definition 1 (Restricted shuffling and masking). *When first-order Boolean masking is applied to the entire cipher and shuffling is only applied to **the first and last rounds**, we call these countermeasures restricted shuffling and masking.*

Algorithm 1 shows the application of the restricted shuffling and masking countermeasures to a block cipher that is mainly composed of key addition, confusion, and diffusion layers.

Algorithm 1. Restricted shuffling and masking scheme

Input: Plaintext X , seen as n bits X_i , $i \in \llbracket 0, T - 1 \rrbracket$,
 SubKeys, $R + 1$ ($n \times T$)-bit constants $\text{RoundKey}[r]$, $r \in \llbracket 0, R \rrbracket$
 seen as n bits $\text{RoundKey}[r]_i$, $i \in \llbracket 0, T - 1 \rrbracket$,
Output: Ciphertext X , seen as n bits X_i , $i \in \llbracket 0, T - 1 \rrbracket$

▷ PreComputation

1: $m, m' \leftarrow_R \mathbb{F}_2^n$

2: Generate the masked S-Box using m and m'

```

▷ Encryption
// Start of partial shuffling countermeasure
3: Generate  $\varphi, \widetilde{X} \leftarrow \kappa(\kappa_m(X, m, \varphi^0), \text{RoundKey}[0], \varphi)$ 
4: Generate  $\varphi, \widetilde{X} \leftarrow \gamma(\widetilde{X}, \varphi)$ 
5: Generate  $\varphi, \widetilde{X} \leftarrow \lambda(\widetilde{X}, \varphi)$ 
// End of partial shuffling countermeasure
6:  $\widetilde{X} \leftarrow \kappa(\widetilde{X}, \text{RoundKey}[1], \varphi^0)$ 
7: for  $r = 2$  to  $R - 2$  do
8:    $\widetilde{X} \leftarrow \gamma(\widetilde{X}, \varphi^0)$ 
9:    $\widetilde{X} \leftarrow \lambda(\widetilde{X}, \varphi^0)$ 
10:   $\widetilde{X} \leftarrow \kappa(\widetilde{X}, \text{RoundKey}[r], \varphi^0)$ 
11: end for
12:  $\widetilde{X} \leftarrow \gamma(\widetilde{X}, \varphi^0)$ 
// Start of partial shuffling countermeasure
13: Generate  $\varphi, \widetilde{X} \leftarrow \lambda(\widetilde{X}, \varphi)$ 
14: Generate  $\varphi, \widetilde{X} \leftarrow \kappa(\widetilde{X}, \text{RoundKey}[R - 1], \varphi)$ 
15: Generate  $\varphi, \widetilde{X} \leftarrow \gamma(\widetilde{X}, \varphi)$ 
16: Generate  $\varphi, X \leftarrow \kappa_m(\kappa(\widetilde{X}, \text{RoundKey}[R], \varphi), m, \varphi^0)$ 
// End of partial shuffling countermeasure
17: return  $X$ 

```

2.3 Existing Confusion Layer Vulnerabilities in Block Ciphers

In this section, we describe the existing vulnerabilities, which commonly exploit the confusion layer output. In general, the confusion layer of a block cipher comprises S-boxes whose output is the main target for performing CPA. In order to protect S-box operation against generic CPA attacks, first-order Boolean masking and shuffling countermeasures are typically employed together due to the trade-offs involved in the security features of these countermeasures. Then, these countermeasures cannot be easily broken except by a profiling attack. Enormous numbers of traces are required to successfully break them, and it is well-known that more traces are required to retrieve the correct key when shuffling is applied. There are two main attacks against shuffling countermeasures: a generic CPA and an integrated CPA against shuffling countermeasure. We call the generic and integrated SOCPAs as G-SOCPA and I-SOCPA, respectively.

Remark 1 (G-CPA: Generic CPA against shuffling [11]). The signal, including information about the sensitive variable X , is randomly split over p different signals, reducing the correlation coefficient by the shuffling complexity p . In other words, $\alpha \times p^2$ traces are required to retrieve the correct key when shuffling is applied, where α is the number of traces required to recover the correct key when shuffling is not applied.

Remark 2 (I-CPA: Integrated CPA against shuffling [15]). Herein, although the signal, including information about the sensitive variable X , is still randomly split over p different signals, the correlation coefficient is reduced by a factor of \sqrt{p} instead of the shuffling complexity p . In other words, $\alpha \times p$ traces are required to retrieve the correct key when shuffling is applied.

For example, when the 16 S-boxes of a PRINCE implementation are shuffled, 16^2 and 16 times more traces are required for the G-CPA and I-CPA attacks, respectively. These approaches can be extended to perform SOCPA attacks against combined shuffling and masking countermeasures. In that case, **the shuffling complexity increases from p to $\frac{p \cdot (p-1)}{2}$.**

3 A Novel Leakage Against Restricted Masking and Shuffling Countermeasures

In this section, we describe a novel vulnerability of block ciphers using restricted shuffling and masking countermeasures. In other words, we exploit a new target instead of an existing one to reduce the number of traces required to retrieve the secret key. Our proposed main target is the diffusion layer output, which consists of AND and XOR operations, as opposed to the confusion layer output that is the usual target of attacks. We bypass the confusion and diffusion layers of first round to prevent a further increase of the attack complexity owing to shuffling countermeasure.

Unlike the confusion layer, no one has yet tried to attack the diffusion layer output of a block cipher when restricted shuffling and masking countermeasures are used together, because of the increased attack complexity. For example, in Algorithm 1, the attack complexity is 2^{2n} when the main target is the confusion layer output. In contrast, the attack complexity when targeting the diffusion layer output is generally $(2^n)^L$ when the main target is the output, where L is the number of linear operations required to compute a single block of diffusion output. The diffusion layer output of a block cipher is therefore not commonly considered to be a leakage point.

We propose a novel leakage for the diffusion layer output when block ciphers are implemented to protect against CPA by combining restricted shuffling and masking countermeasure. In particular, we assume that the diffusion layer only comprises AND and XOR operations, as in [7]. The following lemma allows us to retrieve the correct key and reduce the complexity of SOCPA attacks.

Lemma 1. *Let A, B , and C be n bit variables. Then, $(A \wedge B) \oplus (A \wedge C)$ if and only if $A \wedge (B \oplus C)$.*

Proof. By Table 1, $A \oplus B$ corresponds to $(A \vee B) \wedge (A \wedge B)^c$. Based on this fact, we can prove Lemma 1 as follows.

Table 1. Truth table

A	B	$A \oplus B$	$(A \vee B) \wedge (A \wedge B)^c$
F	F	F	F
F	T	T	T
T	F	T	T
T	T	F	F

$$\begin{aligned}
& (A \wedge B) \oplus (A \wedge C) \\
\Leftrightarrow & \{(A \wedge B) \vee (A \wedge C)\} \wedge \{(A \wedge B) \wedge (A \wedge C)\}^c \\
\Leftrightarrow & \{A \wedge (B \vee C)\} \wedge (A \wedge B \wedge C)^c \\
\Leftrightarrow & \{A \wedge (B \vee C)\} \wedge \{A \wedge (B \wedge C)\}^c \\
\Leftrightarrow & \{A \wedge (B \vee C)\} \wedge \{A^c \vee (B \wedge C)^c\} \\
\Leftrightarrow & [\{A \wedge (B \vee C)\} \wedge A^c] \vee [\{A \wedge (B \vee C)\} \wedge (B \wedge C)^c] \\
\Leftrightarrow & A \wedge (B \vee C) \wedge (B \wedge C)^c \\
\Leftrightarrow & A \wedge (B \oplus C)
\end{aligned}$$

□

As previously discussed, our main target is the diffusion layer output, which comprises AND and XOR operations. In other words, by using a SOCPA attack, we can retrieve the correct key from the input to the second round by utilizing Lemma 1. Generalizing Lemma 1 to the diffusion layer output is more complicated so instead we show how to apply it to two block ciphers: PRINCE and SEED. Without loss of generality, we define Y_i as the confusion layer output corresponding to $S[X_i \oplus \text{RoundKey}[r]_i]$, where $S[\cdot]$ refers to an S-box and $0 \leq i < T$.

3.1 Case Study on the PRINCE Block Cipher

Before showing how to utilize Lemma 1, we briefly discuss the diffusion layer of the PRINCE block cipher. For this cipher, the basic operation unit is a nibble and L , the number of blocks used to compute a single diffusion layer output, is 4.

Diffusion Layer. The diffusion layer output can be calculated as follows.

$$\begin{aligned}
Z_{0+i} &= (M_0 \wedge Y_{0+i}) \oplus (M_1 \wedge Y_{1+i}) \oplus (M_2 \wedge Y_{2+i}) \oplus (M_3 \wedge Y_{3+i}) \\
Z_{1+i} &= (M_1 \wedge Y_{0+i}) \oplus (M_2 \wedge Y_{1+i}) \oplus (M_3 \wedge Y_{2+i}) \oplus (M_0 \wedge Y_{3+i}) \\
Z_{2+i} &= (M_2 \wedge Y_{0+i}) \oplus (M_3 \wedge Y_{1+i}) \oplus (M_0 \wedge Y_{2+i}) \oplus (M_1 \wedge Y_{3+i}) \\
Z_{3+i} &= (M_3 \wedge Y_{0+i}) \oplus (M_0 \wedge Y_{1+i}) \oplus (M_1 \wedge Y_{2+i}) \oplus (M_2 \wedge Y_{3+i})
\end{aligned}$$

where $i \in \{0, 4, 8, 12\}$ and $M_0 = 0x7$, $M_1 = 0xB$, $M_2 = 0xD$, $M_3 = 0xE$.

In order to mount a SOCPA attack on our new target against the restricted shuffling and masking countermeasures, we focus on the point wherein the second round input is loaded, which directly gives the diffusion layer output because shuffling countermeasures are only applied to the first and last rounds. To utilize Lemma 1, we have to select one of the four cases $\{0, 4, 8, 12\}$. Subsequently, to perform a SOCPA attack, preprocessing is required after selecting two leakage points, we now provide an example for $\{Z_{0+i}, Z_{1+i}\}$.

$$\begin{aligned}
 & Z_{0+i} \oplus Z_{1+i} \\
 &= \{(M_0 \wedge Y_{0+i}) \oplus (M_1 \wedge Y_{1+i}) \oplus (M_2 \wedge Y_{2+i}) \oplus (M_3 \wedge Y_{3+i})\} \oplus \\
 &\quad \{(M_1 \wedge Y_{0+i}) \oplus (M_2 \wedge Y_{1+i}) \oplus (M_3 \wedge Y_{2+i}) \oplus (M_0 \wedge Y_{3+i})\} \\
 &= \{(M_0 \oplus M_1) \wedge Y_{0+i}\} \oplus \{(M_1 \oplus M_2) \wedge Y_{1+i}\} \oplus \\
 &\quad \{(M_2 \oplus M_3) \wedge Y_{2+i}\} \oplus \{(M_3 \oplus M_0) \wedge Y_{3+i}\} \text{ (By Lemma 1)} \\
 &= (1100_2 \wedge Y_{0+i}) \oplus (0110_2 \wedge Y_{1+i}) \oplus (0011_2 \wedge Y_{2+i}) \oplus (1001_2 \wedge Y_{3+i}) \\
 &= ((Y_{0+i})_{(3)} \oplus (Y_{3+i})_{(3)} \parallel (Y_{0+i})_{(2)} \oplus (Y_{1+i})_{(2)} \parallel \\
 &\quad (Y_{1+i})_{(1)} \oplus (Y_{2+i})_{(1)} \parallel (Y_{2+i})_{(0)} \oplus (Y_{3+i})_{(0)})
 \end{aligned}$$

where $(X)_{(s)}$ means the s -th least significant bit.

By Eq. (1), it can be noted that each bit of diffusion layer outputs is related to two rather than four confusion layer outputs. Therefore, the attack complexity, based on Lemma 1, can be reduced from $(2^4)^4$ to $(2^4)^2$ for a SOCPA attack. For example, $(\widetilde{Z_{0+i}} \oplus \widetilde{Z_{1+i}}) \wedge 0100_2 = (Z_{0+i} \oplus Z_{1+i}) \wedge 0100_2 = (Y_{0+i})_{(2)} \oplus (Y_{1+i})_{(2)}$.

As mentioned earlier, with the restricted shuffling and masking countermeasures, we can mount a SOCPA attack because each byte is masked using the identical value. Consequently, we can perform a SOCPA attack on the diffusion layer output with an attack complexity of $(2^4)^2$.

3.2 Case Study on the SEED Block Cipher

Similar to block cipher PRINCE, we briefly explain the diffusion layer of block cipher SEED. Although block cipher SEED consists of two type S-boxes, we maintain the notation of S-box as previous. By the definition of block cipher SEED, the basic operation unit is byte and L corresponds to 4 which is identical to block cipher PRINCE.

Diffusion Layer. The diffusion layer output of the SEED block cipher can be calculated as follows.

$$\begin{aligned}
 Z_0 &= (M_0 \wedge Y_0) \oplus (M_1 \wedge Y_1) \oplus (M_2 \wedge Y_2) \oplus (M_3 \wedge Y_3) \\
 Z_1 &= (M_1 \wedge Y_0) \oplus (M_2 \wedge Y_1) \oplus (M_3 \wedge Y_2) \oplus (M_0 \wedge Y_3) \\
 Z_2 &= (M_2 \wedge Y_0) \oplus (M_3 \wedge Y_1) \oplus (M_0 \wedge Y_2) \oplus (M_1 \wedge Y_3) \\
 Z_3 &= (M_3 \wedge Y_0) \oplus (M_0 \wedge Y_1) \oplus (M_1 \wedge Y_2) \oplus (M_2 \wedge Y_3)
 \end{aligned}$$

where $M_0 = 0xFC$, $M_1 = 0xF3$, $M_2 = 0xCF$, $M_3 = 0x3F$.

As before, we now give an example for $\{Z_0, Z_1\}$.

$$\begin{aligned}
& Z_0 \oplus Z_1 \\
&= \{(M_0 \wedge Y_0) \oplus (M_1 \wedge Y_1) \oplus (M_2 \wedge Y_2) \oplus (M_3 \wedge Y_3)\} \oplus \\
&\quad \{(M_1 \wedge Y_0) \oplus (M_2 \wedge Y_1) \oplus (M_3 \wedge Y_2) \oplus (M_0 \wedge Y_3)\} \\
&= \{(M_0 \oplus M_1) \wedge Y_0\} \oplus \{(M_1 \oplus M_2) \wedge Y_1\} \oplus \\
&\quad \{(M_2 \oplus M_3) \wedge Y_2\} \oplus \{(M_3 \oplus M_0) \wedge Y_3\} \quad (\text{By Lemma 1}) \quad (1) \\
&= (00001111_2 \wedge Y_0) \oplus (00111100_2 \wedge Y_1) \oplus \\
&\quad (11110000_2 \wedge Y_2) \oplus (11000011_2 \wedge Y_3) \\
&= \{11000000_2 \wedge (Y_2 \oplus Y_3)\} \oplus \{00110000_2 \wedge (Y_1 \oplus Y_2)\} \\
&\quad \oplus \{00001100_2 \wedge (Y_0 \oplus Y_1)\} \oplus \{00000011_2 \wedge (Y_3 \oplus Y_0)\}
\end{aligned}$$

This shows that $\widetilde{Z}_0 \oplus \widetilde{Z}_1$ leaks second-order information that indicates part of $Y_2 \oplus Y_3$, $Y_1 \oplus Y_2$, $Y_0 \oplus Y_1$ or $Y_3 \oplus Y_0$. For example, $(\widetilde{Z}_0 \oplus \widetilde{Z}_1) \wedge 00001100_2 = (Z_0 \oplus Z_1) \wedge 00001100_2 = (Y_0 \oplus Y_1) \wedge 00001100_2$. Consequently, the complexity of the attack required to recover the secret key is 2^{16} instead of 2^{32} .

4 Security Analysis

In this section, we perform our proposed attack, as well as G-SOCPA and I-SOCPA attacks, on simulated and realistic traces. As discussed previously, we use correlations with the combined S-box outputs to guess the intermediate variables. However, unlike existing schemes, our proposed attack point where the diffusion layer output data is loaded as input. That is, the point wherein the diffusion layer output is loaded into the next operation after the diffusion layer has been calculated. To make this clearer, we summarize the attack point, complexities, and correlation-reduction factors of the different attacks in Table 2.

Except for Case 3, the intermediate variables are combinations of S-box outputs. Thus, we do not consider Case 3 in this paper due to its high complexity. Instead, we focus on performing Cases 1, 2, and 4 against restricted shuffling and masking countermeasures. In Cases 1 and 2, the attack point is conventional. Although the restricted shuffling countermeasure is used, the S-boxes are the main targets. Therefore, the main leaks are at the points wherein the S-box outputs are loaded or saved.

In contrast to the usual approach, our proposal regards the point wherein the diffusion layer outputs are loaded as the main leak. The point wherein the diffusion layer outputs are saved cannot be used due to the shuffling countermeasure. In spite of this limitation, our proposal still has an advantage because the loading point has been used for some template attacks [3], indicating that its leakage is sufficient.

Table 2. Summary of targets for experimental SOCPA attacks

Case	Attack scheme	Intermediate variables	Attack point	Attack complexity	Correlation-reduction factor
Case 1	G-SOCPA	S-box outputs	Loading or Saving of S-box outputs	2^{2n}	$1/\binom{T}{2}$
Case 2	I-SOCPA	S-box outputs	Loading or Saving of S-box outputs	2^{2n}	$1/\sqrt{\binom{T}{2}}$
Case 3	G-SOCPA	Diffusion layer outputs	Loading of Diffusion layer outputs	$(2^n)^L$	1
Case 4	Our suggestion	S-box outputs	Loading of Diffusion layer outputs	2^{2n}	$1/\sqrt{\frac{n}{l}}$

The correlation-reduction factor is the factor by which the correlation between the intermediate variable and trace is reduced for SOCPA attack. In Case 1, the correlation is reduced by $1/\binom{T}{2}$ because the shuffling complexity T . Namely, the number of traces required to retrieve the correct key in Case 1 is $\alpha \times \binom{T}{2}^2$, where α is the number of traces required without the shuffling countermeasure [8]. In the same way, the number of traces required in Case 2 is $\alpha \times \binom{T}{2}$ [15].

Our proposal is not affected by shuffling countermeasure since the main target is the point wherein the diffusion layer outputs are loaded. However, the intermediate variable is only partially available as the critical key leakage due to a partial correlation with the combined S-box outputs. More precisely, the correlation-reduction factor for our suggestion is derived from Sect. 4.1 where n indicates the number of bits in the intermediate variable, and l denotes the number of bits in the S-box outputs that are correlated when performing the SOCPA attack. In the case of the PRINCE block cipher, n and l are 4 and 1, respectively.

The number of traces required for our proposal can be roughly determined from the correlation-reduction factor. By [8], the number of traces required depends on the inverse square of the correlation-reduction factor, meaning $\alpha \times \left(\sqrt{\frac{n}{l}}\right)^2$ traces are required.

4.1 Statistical Derivation of the Correlation Reduction Factor

Herein, we derive the correlation-reduction factor for our proposal. We assume that the power consumption conforms to the linear Hamming weight model [2]. Therefore, the basic model for the intermediate variable can be written as:

$$W = H(V) + N \quad (2)$$

where $H(\cdot)$ is the Hamming weight function, V is the intermediate variable (e.g., $S[X_0 \oplus \text{RoundKey}[0]_0] \oplus S[X_1 \oplus \text{RoundKey}[0]_1]$ for SOCPA), W is the power consumption, and N is the noise, defined as $N \sim \mathcal{N}(0, \sigma^2)$, where σ^2 denotes the noise variance.

If the pre-processed power consumption W perfectly conforms to the linear Hamming weight model, then we have the correlation $\rho_{WV} = \frac{\text{cov}(W, H)}{\sigma_W \sigma_H}$ where H is $H(V)$. In our proposal, part of $H(V)$ is utilized to perform the SOCPA attack. This correlation can be written as in Lemma 2.

Lemma 2. *Let $H'(V)$ be the Hamming weight derived from part of the intermediate variable, n be the number of bits in the intermediate variable, and l be the number of bits in the S-box output that are correlated for the SOCPA attack. Then, we have*

$$\rho_{WH'} = \rho_{WH} \rho_{HH'} = \sqrt{\frac{l}{n}} \rho_{WH}$$

Proof. The proof of Lemma 2 is given in Appendix A. □

For example, in the cases where $n = 4$ and $l = 1$ (for the PRINCE block cipher) or $n = 8$ and $l = 2$ (for the SEED block cipher), the correlation is reduced by $\sqrt{\frac{1}{4}} = \sqrt{\frac{2}{8}}$. Therefore, the number of traces required is at least four times more than without the countermeasure.

4.2 Simulation Evaluation

In this section, we evaluate the proposed scheme against restricted shuffling and masking countermeasures when the leakage conforms to Eq. (2). Herein, we generated T intermediate variables at the points $(V_0, V_1, \dots, V_{T-1})$, which specify $(Y_{\varphi(0)}, Y_{\varphi(1)}, \dots, Y_{\varphi(T-1)})$ for Cases 1 and 2 or $(Z_0, Z_1, \dots, Z_{T-1})$ for Case 4, where $\varphi(i)$ means the i^{th} operation after applying the shuffling countermeasure.

More precisely, to provide practical attack scenarios, we considered models with three noise level: low, medium, and high, with standard deviations of $\sigma_{low} = 0.25$, $\sigma_{med} = 1$, and $\sigma_{high} = 4$, respectively.

In addition, we used guessing entropy [14] as the security metric because we consider realistic countermeasures in this paper. In all experiments, 1,000,000 traces were generated, and each attack was repeated 50 times using randomly chosen traces.

4.2.1 PRINCE Block Cipher

The PRINCE block cipher's substitution comprises 16 S-boxes; therefore shuffling complexity for the substitution layer is 16. For our proposal, we generate traces diffusion layer output traces because the data cannot be shuffled when the diffusion layer output is loaded, although restricted shuffling and masking countermeasures were applied to the diffusion layer. Using this setup, we evaluated

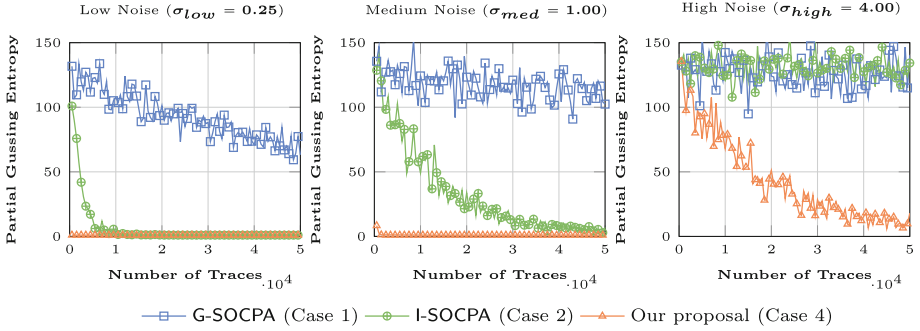


Fig. 1. Simulation results for the PRINCE block cipher

Cases 1, 2, and 4 against restricted shuffling and masking countermeasures for the PRINCE block cipher, and the results are shown in Fig. 1.

The intermediate variable $Y_0 \oplus Y_1$ was used to evaluate Cases 1 and 2. For Case 4, $(Y_0 \oplus Y_1) \wedge 0100_2$ was used to target the combination of Z_0 and Z_1 .

For the low noise model, our proposal and I-SOCPA both recovered the secret key for a few traces. Although there was no full leakage with respect to G-SOCPA, a quarter of the guessing entropy was revealed. In contrast, our proposal achieved full leakage with only 1,000 traces, where 20 times as many traces were required to reach a guessing entropy of 1 with I-SOCPA.

The guessing entropy of I-SOCPA dropped below 5 for the medium noise model although the correct key could not be retrieved directly. In addition, Case 4 (our proposal) was still very effective as it only required 2,500 traces to recover the correct key.

Lastly, for the high noise model, the correct key was not be completely revealed for all cases. However, the guessing entropy for our proposal appeared to converge to a very low value; therefore, it could still be considered to be practical attack scheme against restricted shuffling and masking countermeasures.

4.2.2 SEED Block Cipher

This section is analogous to the previous one for the PRINCE block cipher in terms of experimental method. However, the substitution layer for SEED’s G-function consisted of four S-boxes. The SEED cipher’s security may not be sufficient despite the shuffling countermeasure, owing to the small the number of S-boxes. To reinforce its security, fixed dummy operations can be effective [6]; therefore, four dummy operations (the same as the number of S-boxes) were used in our experiment, indicating that T was eight.

Using this setup, we evaluated Cases 1, 2, and 4 against restricted shuffling and masking countermeasures for the SEED block cipher, and the results are shown in Fig. 2. The intermediate variable $Y_0 \oplus Y_1$ was used to evaluate Cases 1 and 2, and $(Y_0 \oplus Y_1) \wedge 00001100_2$ was used for Case 4.

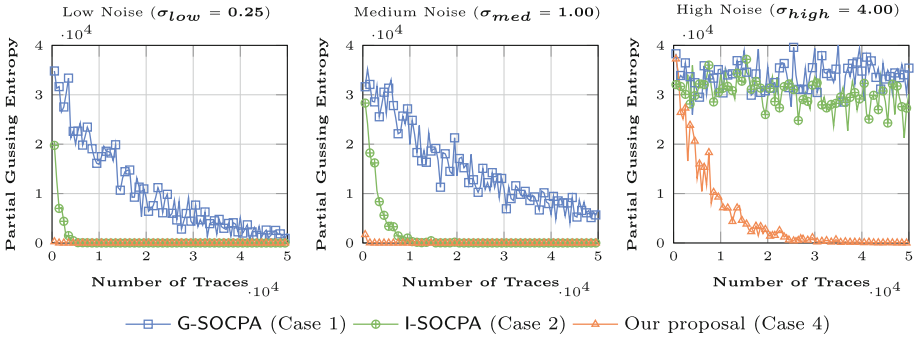


Fig. 2. Simulation results for the SEED block cipher

For the low noise model, several of the attacks revealed the correct key. However, the guessing entropy was over 1,000 for G-SOCPA; therefore, it could not retrieve the correct key despite the low noise. As expected, this demonstrated the effectiveness of the windowing attack, which only required 13,000 traces to reach a guessing entropy of 1. In addition, our proposal only required 1,500 traces (about 1/8 of as many) to recover the correct key. In the medium noise model, even though the I-SOCPA attack only needed about 28,000 traces to confirm the leakage of the correct key, roughly 3,000 traces allow Case 4 (our proposal) to reveal the correct key.

As expected for the high noise, the correct key could not be retrieved. Only our proposal was almost able to find the correct key so it could have broken the countermeasure if more traces had been collected.

4.3 Practical Evaluation

In this section, we present experimental results for a realistic embedded board. For this, we used an XMEGA128D4-I chip embedded on a ChipWhisperer-Lite (CW1173) [9]. The sampling rate was four times the 29.5 MHz clock frequency of the chip. There are two attack targets. For Cases 1 and 2, we acquired traces when the substitution layer was calculated. For our proposal, traces were collected when the diffusion layer outputs were loaded. As with the simulation results, guessing entropy was employed as the security metric, calculated using the same variables as in the simulations.

The results for this experiment are analogous to the medium noise cases in the simulations. The I-SOCPA attack was able to achieve low guessing entropy but could not immediately find the correct key. Our proposal, which utilizes the correlation between the diffusion layer and confusion layer outputs, was more effective than the existing attacks, requiring only 3,000 and 7,000 traces to find the correct key directly for the PRINCE and SEED ciphers, respectively.

In conclusion, our proposed scheme offers an overwhelming improvement in terms of the number of traces required. More precisely, an adversary using our

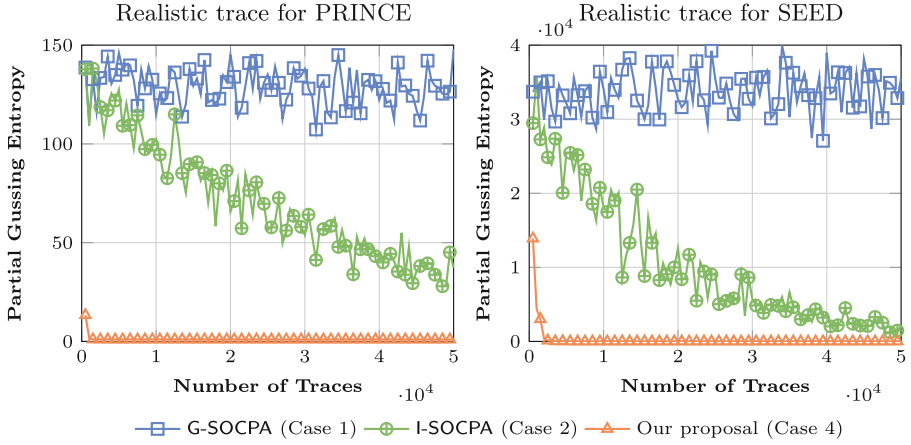


Fig. 3. Realistic results for the PRINCE and SEED block ciphers

proposal would require more than 17 times fewer traces to break the PRINCE block cipher and achieve a guessing entropy of one compared to I-SOCPA.

5 Conclusion

Restricted shuffling and masking countermeasures can provide adequate security and performance in practice. However, this countermeasure can be defeated by our proposed scheme due to a partial correlation between the diffusion layer and confusion layer outputs. Using simulations, we have demonstrated that our proposal offers an overwhelming advantage compared with existing attacks. For instance, the number of traces required can be reduce by 95%, from 21,000 to 1,000, for the PRINCE block cipher. Moreover, for a realistic scenario, we have demonstrated that the correct key can be retrieved using only a few traces even though other attacks such as G-SOCPA and I-SOCPA fail to reveal the correct key directly.

To protect against our proposal in the real-world, the diffusion layer outputs would have to be concealed a series of different masking values with each row. Moreover, this computation would have to be done as part of the shuffling countermeasure and with acceptable overhead.

Acknowledgements. This work was supported by the Institute for Information and Communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No. 20170005200011001, Development of SCR-Friendly Symmetric Key Cryptosystem and Its Application Modes).

A Proof of Lemma 2

Proof. First, we prove the equation $\rho_{WH'} = \rho_{WH}\rho_{HH'}$, which can be derived as in [6].

$$\begin{aligned}
 \rho_{WH'} &= \frac{\text{Cov}(W, H')}{\sigma_W \sigma_{H'}} = \frac{\text{Cov}(H + N, H')}{\sigma_W \sigma_{H'}} \\
 &= \frac{E[(H + N)H'] - E[H + N]E[H']}{\sigma_W \sigma_{H'}} \\
 &= \frac{E[HH'] - E[H]E[H']}{\sigma_W \sigma_{H'}} = \frac{\text{Cov}(H, H')}{\sigma_W \sigma_{H'}} \\
 &= \frac{\text{Cov}(H, H')}{\sigma_H \sigma_{H'}} \cdot \frac{\sigma_H^2}{\sigma_W \sigma_H} \\
 &= \rho_{HH'} \cdot \frac{\text{Cov}(H + N, H)}{\sigma_W \sigma_H} = \rho_{WH} \cdot \rho_{HH'}
 \end{aligned}$$

Before we prove the equation $\rho_{HH'} = \sqrt{\frac{l}{n}}$, we define H_l as the Hamming weight for l bits out of a total of n bits. Then, by [11], $E[H_l] = \frac{l}{2}$, $\text{Var}[H_l] = \frac{l}{4}$, and hence

$$\begin{aligned}
 \rho_{HH'} &= \frac{\text{Cov}(H_n, H_l)}{\sigma_{H_n} \sigma_{H_l}} \\
 &= \frac{\text{Cov}(H_{n-l} + H_l, H_l)}{\sigma_{H_n} \sigma_{H_l}} \\
 &= \frac{E[(H_{n-l} + H_l)H_l] - E[H_{n-l} + H_l]E[H_l]}{\sigma_{H_n} \sigma_{H_l}} \\
 &= \frac{E[H_{n-l}H_l] + E[H_l^2] - E[H_{n-l}]E[H_l] - (E[H_l])^2}{\sigma_{H_n} \sigma_{H_l}} \\
 &= \frac{E[H_l^2] - (E[H_l])^2}{\sigma_{H_n} \sigma_{H_l}} (\because H_{n-l} \text{ and } H_l \text{ are independent}) \\
 &= \frac{\text{Var}[H_l]}{\sigma_{H_n} \sigma_{H_l}} = \frac{\frac{l}{4}}{\sqrt{\frac{n}{4}}\sqrt{\frac{l}{4}}} = \sqrt{\frac{l}{n}}.
 \end{aligned}$$

□

References

1. Borghoff, J., et al.: PRINCE – a low-latency block cipher for pervasive computing applications. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 208–225. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34961-4_14
2. Brier, E., Clavier, C., Olivier, F.: Correlation power analysis with a leakage model. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 16–29. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28632-5_2
3. Choudary, O., Kuhn, M.G.: Efficient template attacks. In: Francillon, A., Rohatgi, P. (eds.) CARDIS 2013. LNCS, vol. 8419, pp. 253–270. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08302-5_17
4. Chari, S., Jutla, C.S., Rao, J.R., Rohatgi, P.: Towards sound approaches to counter-act power-analysis attacks. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 398–412. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_26
5. Goubin, L., Patarin, J.: DES and differential power analysis the “Duplication” method. In: Koç, Ç.K., Paar, C. (eds.) CHES 1999. LNCS, vol. 1717, pp. 158–172. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48059-5_15
6. Herbst, C., Oswald, E., Mangard, S.: An AES smart card implementation resistant to power analysis attacks. In: Zhou, J., Yung, M., Bao, F. (eds.) ACNS 2006. LNCS, vol. 3989, pp. 239–252. Springer, Heidelberg (2006). https://doi.org/10.1007/11767480_16
7. Kim, H., Cho, Y.I., Choi, D., Han, D.G., Hong, S.: Efficient masked implementation for SEED based on combined masking. *ETRI J.* **33**(2), 267–274 (2011)
8. Mangard, S., Oswald, E., Popp, T.: *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer, Heidelberg (2007). <https://doi.org/10.1007/978-0-387-38162-6>
9. O’Flynn, C., Chen, Z.D.: ChipWhisperer: an open-source platform for hardware embedded security research. In: Prouff, E. (ed.) COSADE 2014. LNCS, vol. 8622, pp. 243–260. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10175-0_17
10. Oswald, E., Mangard, S., Herbst, C., Tillich, S.: Practical second-order DPA attacks for masked smart card implementations of block ciphers. In: Pointcheval, D. (ed.) CT-RSA 2006. LNCS, vol. 3860, pp. 192–207. Springer, Heidelberg (2006). https://doi.org/10.1007/11605805_13
11. Prouff, E., Rivain, M., Bevan, R.: Statistical analysis of second order differential power analysis. *IEEE Trans. Comput.* **58**(6), 799–8141 (2009)
12. Rivain, M., Prouff, E., Doget, J.: Higher-order masking and shuffling for software implementations of block ciphers. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 171–188. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04138-9_13
13. *Information Technology – Security Techniques – Encryption Algorithms – Part 3: Block Ciphers*, ISO/IEC 18033–3:2005 (2005)
14. Standaert, F.-X., Malkin, T.G., Yung, M.: A unified framework for the analysis of side-channel key recovery attacks. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 443–461. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01001-9_26

15. Tillich, S., Herbst, C.: Attacking state-of-the-art software countermeasures—a case study for AES. In: Oswald, E., Rohatgi, P. (eds.) CHES 2008. LNCS, vol. 5154, pp. 228–243. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85053-3_15
16. Tillich, S., Herbst, C., Mangard, S.: Protecting AES software implementations on 32-bit processors against power analysis. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 141–157. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72738-5_10



Detecting Similar Code Segments Through Side Channel Leakage in Microcontrollers

Peter Samarin^{1,2}(✉) and Kerstin Lemke-Rust¹(✉)

¹ Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin, Germany

{peter.samarin,kerstin.lemke-rust}@h-brs.de

² Ruhr-Universität Bochum, Bochum, Germany

Abstract. We present new methods for detecting plagiarized code segments using side-channel leakage of microcontrollers. Our approach uses the dependency of side-channel leakage on processed data and requires that the implementation under test accepts varying known input data. Detection tools are built upon a similarity matrix that contains the absolute correlation coefficient for each combination of time samples of the two possibly different implementations as result of side channel measurements. These methods are evaluated on smartcards with ATMega163 microcontroller using different test applications written in assembly language. We show that our methods are highly robust even against a skilled adversary who modifies the original assembly code in various ways. Our approach is non-intrusive, so that the application does not need to be additionally watermarked in order to be protected—the resulting pattern of data leakage of the microcontroller executing the code is considered as its own watermark.

Keywords: Side-channel watermarking · IP protection
Code similarity analysis · Similarity matrix
Software reverse engineering · Embedded software

1 Introduction

Intellectual property (IP) theft of embedded software for microcontrollers with effective read-out protection of memories constitutes a hard problem for IP protection nowadays. A direct binary analysis is not feasible as the suspicious program code needs to be physically extracted from the internal code memory first which constitutes a hard and cost-intensive problem in practice. To address this problem, the use of side channel leakage was first proposed by Becker et al. [1] by implementing additional watermarking code for a leakage generator. In their followup work, Becker et al. [2] have used the coincidence that ATMega8 microcontrollers leak the Hamming weight of the opcode when an instruction is fetched. The power traces are converted into strings, and the strings are

compared using string matching algorithms, such as edit distance and Boyer-Moore-Horspool algorithm. However, this approach relies on processor-specific properties that do not in general apply—not all microcontrollers leak Hamming weights of the opcodes. In addition, Hamming weights of opcodes are not unique, such that many different instructions can have the same Hamming weight. Also, the method is not robust against code-transformation attacks that exchange assembly instructions by others leading to the same output, replace registers and RAM and flash addresses of variables and data.

A different approach was taken by Strobel et al. [3], where the authors tried to disassemble instructions from electromagnetic (EM) traces of PIC16F687 microcontroller by training a classifier that is able to distinguish 87.60% of instruction classes correctly. This approach, however, has been only tested on a small range of microcontrollers.

Another way to prove ownership is to look for dependency between the power consumption and executed code. Durvaux et al. [4] compute similarity of two power traces using Pearson’s correlation along the time axis. The intrinsic side channel leakage of different implementations is compared without the need of additional watermarking code. However, their proposed method shows low robustness if the adversary adds dummy instructions to the IP protected code.

In this work we propose a new method that addresses the software plagiarism problem on a much finer level—our method can not only identify whether the whole program is a plagiarism, but also which code segments have been plagiarized. In addition, our method shows a significantly better robustness against additions of dummy code when compared to the current state of the art approach.

2 Our Approach

Since the pioneering work of Kocher et al. [5] it is well known that during code execution data dependencies in microcontroller programs induce side-channel leakage that is measurable in the power consumption of the microcontroller or in its electromagnetic (EM) emanation. And side-channel leakage discloses the positions in time where a targeted intermediate data item is processed, such that repeatedly processing the same data using the same implementation will result in a very similar physical leakage. Our approach builds upon this finding for the use in IP protection. Considering the framework for IP protection of [6], our contribution introduces new detection tools for IP protection using side channel leakage. Our main approach requires that an equivalent input data channel is available for the configuration of the program code of the genuine implementation and the second unknown and possibly suspicious implementation. The existence of an equivalent input data channel is a reasonable assumption when different program code is used for the same purpose. Our objective is to provide robust similarity detection tools using code and data characteristics in two given implementations.

2.1 Extraction: Data Acquisition

The objective of the data acquisition step is the collection of N side channel traces of the genuine program and N side channel traces of the unknown program, both processing the same list of N varying input data. This can be achieved either in a chosen-input scenario but also in a known-input scenario. In the latter case the unknown program has to be measured first and a chosen-input variant of the genuine program has to be available that is fed with the same input data afterwards. Each set of measurements and their corresponding input data represent a soft physical hash value [6] or a fingerprint of the genuine and unknown code, respectively. To make comparison of similarity easier, it is recommended to use an identical measurement setup for both implementations. Details on side-channel measurement setups can be found in [5, 7].

2.2 Extraction: Preprocessing

Before comparing the two fingerprints, the number of samples per trace is reduced by compressing them, e.g., by extracting the mean of each clock cycle. In principle, this preprocessing step is optional, but useful in practice to reduce the computation time in the following detection phase.

2.3 Detection: Similarity Matrix

As result of the extraction phase, we build two matrices:

1. An $N \times M_1$ matrix $T_{genuine}$ that contains N rows of compressed side channel traces with M_1 samples each. $T_{genuine}$ contains the extracted measurement data from the genuine implementation.
2. An $N \times M_2$ matrix $T_{unknown}$ that contains N rows of compressed side channel traces with M_2 samples each. $T_{unknown}$ contains the extracted measurement data from the unknown implementation.

In the following, we denote an entry of a matrix T as T_{ij} , the j -th column vector as $T_{:,j}$, and the i -th row vector as $T_{i,:}$. An entry of a vector \mathbf{x} is denoted by x_i and \mathbf{x}^T is the transpose of \mathbf{x} .

For the computation of similarity between the data leakage of the genuine and the unknown code the sample Pearson correlation coefficient

$$\hat{\rho}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^N (x_i - \hat{x})(y_i - \hat{y})}{\hat{\sigma}_x \hat{\sigma}_y}$$

is used, where \mathbf{x} and \mathbf{y} are vectors of length N , \hat{x} and \hat{y} are the sample means of \mathbf{x} and \mathbf{y} , respectively, and $\hat{\sigma}_x, \hat{\sigma}_y$ are their respective sample standard deviations.

The basis for our detection methods is the $M_1 \times M_2$ similarity matrix S . Each entry S_{ij} is the absolute correlation coefficient of the column vector $T_{genuine,:,i}$ and the column vector $T_{unknown,:,j}$.

$$S = \begin{pmatrix} |\hat{\rho}(T_{genuine_{:,1}}, T_{unknown_{:,1}})| & \cdots & |\hat{\rho}(T_{genuine_{:,1}}, T_{unknown_{:,M_2}})| \\ \vdots & \ddots & \vdots \\ \vdots & \ddots & \vdots \\ |\hat{\rho}(T_{genuine_{:,M_1}}, T_{unknown_{:,1}})| & \cdots & |\hat{\rho}(T_{genuine_{:,M_1}}, T_{unknown_{:,M_2}})| \end{pmatrix}$$

It holds $0 \leq S_{ij} \leq 1$ for all entries of S . If S_{ij} is close to 1 this indicates a high similarity between sample point i in the genuine implementation and sample point j in the unknown implementation, whereas entries with S_{ij} close to 0 indicate that there is no similarity at these two sample points.

In the final step, we need to make a decision whether the two implementations are similar or not as a whole, and whether they contain highly similar subparts.

Detection Tool: Visual Inspection. The similarity matrix pinpoints the sequence of data processing in the unknown implementation and its analysis discloses the structure of the unknown program code. To quickly find code similarities, we plot the resulting similarity matrix and inspect it visually. Visual inspection is an extremely powerful tool when the unknown program contains a one-to-one copy of the original code, in which case the matrix will contain many diagonal lines with absolute correlation coefficients close to 1.

Detection Tool: Maximum Projection. Rows and columns of the similarity matrix S represent the time measured in samples or clock cycles. To analyze the similarity matrix computationally, we project the matrix either onto the rows or onto the columns by using the maximum function. By projecting along the rows we can see which clock cycles of the genuine implementation are covered by the unknown implementation. Concretely, the maximum projection of S onto the rows results in an M_2 -dimensional vector

$$\mathbf{p}_{row}^T = \left(\max_{i=1, \dots, M_1} S_{i1}, \dots, \max_{i=1, \dots, M_1} S_{iM_2} \right)$$

On the other hand, projecting the matrix onto the columns will reveal the clock cycles where the genuine implementation processes the same data as the unknown one. The M_1 -dimensional vector \mathbf{p}_{col} is computed as

$$\mathbf{p}_{col} = \begin{pmatrix} \max_{j=1, \dots, M_2} S_{1j} \\ \vdots \\ \max_{j=1, \dots, M_2} S_{M_1j} \end{pmatrix}$$

For both projection vectors it holds that sub-parts with high correlation values in succession suggest that the same intermediate data is processed using the same code segment by the microcontroller. For the quantification of similarity we use the mean absolute correlation coefficient calculated over all entries of \mathbf{p}_{row} and \mathbf{p}_{col}

$$\rho_{row} = \frac{1}{M_2} \sum_{i=1}^{M_2} p_{row_i} \quad \text{and} \quad \rho_{col} = \frac{1}{M_1} \sum_{i=1}^{M_1} p_{col_i}.$$

Accordingly, local similarity is assessed by computing the mean absolute correlation coefficient over pre-selected sub-parts of the vectors \mathbf{p}_{row} and \mathbf{p}_{col} .

3 Experiments

We evaluate our approach on different smartcards with ATmega163 microcontroller [8]. It is an 8-bit RISC microcontroller based on the AVR architecture running at 4 MHz with 16 K bytes of flash memory, 1024 bytes internal SRAM, and 32 general-purpose registers [9]. The processor uses a two-stage pipeline, where one instruction is executed while the next instruction is fetched and decoded. We measure the voltage variations of the smartcard over a resistor inserted between the ground path from the smartcard socket to the power supply using a digital oscilloscope (PicoScope 6402C) running at the sampling frequency of 325 MHz. For each implementation we recorded 10.000 power traces.

We test our approach on five implementations of AES encryption written in assembly language. The AES implementations serve as an example of several implementations of the same data-dependent algorithm. Thereby we target the problem of distinguishing several implementations of the same algorithm or application. The purpose of our experiments is not only to show that identical implementations can be detected as a whole, but also which of their code segments are similar or even identical to each other.

Our test implementations differ in their overall duration and structure. An overview is shown in Fig. 1. *AES-0* was written by us, *AES Labor* is available from [10], *Fast*, *Furious*, and *Fantastic* are available from [11]. *Fast* uses two sbx tables, which results in data leakage that is very different from the leakage of all other implementations. Some code parts of different implementations are the same. For example, the key expansion of *Fast* and *Furious* are identical. The MixColumns operation of *AES-0* and *AES Fantastic* follow the same low-level specification of the algorithm but utilize different registers and data addresses.

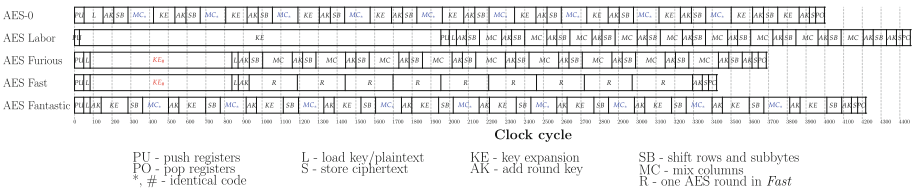


Fig. 1. AES implementations used in our experiments.

We explore two cases of plagiarism: (i) The passive adversary copies the entire machine code or parts thereof into his own implementation. This case is

considered in Sect. 3.1. (ii) The active adversary modifies the machine code in various ways before copying it or parts thereof into his own implementation. This case is covered in Sect. 3.2.

3.1 Plagiarized Code Without Modification

Visual Inspection. By visually inspecting a similarity matrix, we can detect identical and similar code. Figure 2 shows segments of similarity matrices for all implementations and the following AES operations: ExpandKey_1 , AddRoundKey_2 , ShiftRows_2 and SubBytes_2 , and MixColumns_2 , where subscripts denote the round of the corresponding function. When comparing each implementation to itself, we see long lines of high correlation along the diagonal in the similarity matrices, as shown in Fig. 10. This is expected, since the same implementations process the same data using the same sequence of instructions.

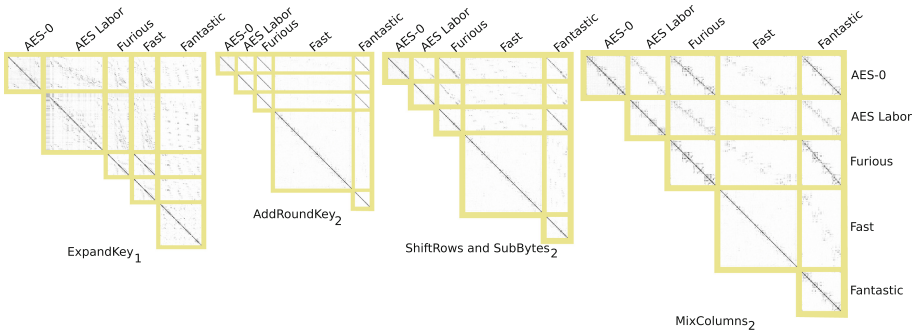


Fig. 2. Segments of similarity matrices for selected subparts of all AES implementations. Black points signify high absolute correlation (close to 1), and white points signify low correlation (close to 0). Subscripts of selected AES subparts denote the AES round: ExpandKey_1 is the key expansion in the first AES round (the key will be used by AddRoundKey_2). *RijndaelFast* has no clear distinction between AddRoundKey , ShiftRows , SubBytes , and MixColumns , so that we consider its entire second round when we compare it to the other implementations.

Several observations can be made. For example, key expansion is different in all implementations except for *Fast* and *Furious*, where it is identical. AddRoundKey_2 is similar for all implementations except for *Fast*. MixColumns_2 is identical in *AES-0* and *Fantastic*, except for the registers used. The other operations are similar but not identical, and we can see that the similar values are processed at slightly different times.

Maximum Projection. Figure 3 shows the results of projecting the maximum values onto the rows of the similarity matrices for *Furious* and all the other tested AES implementations. When comparing *Furious* with *Furious*, we have used

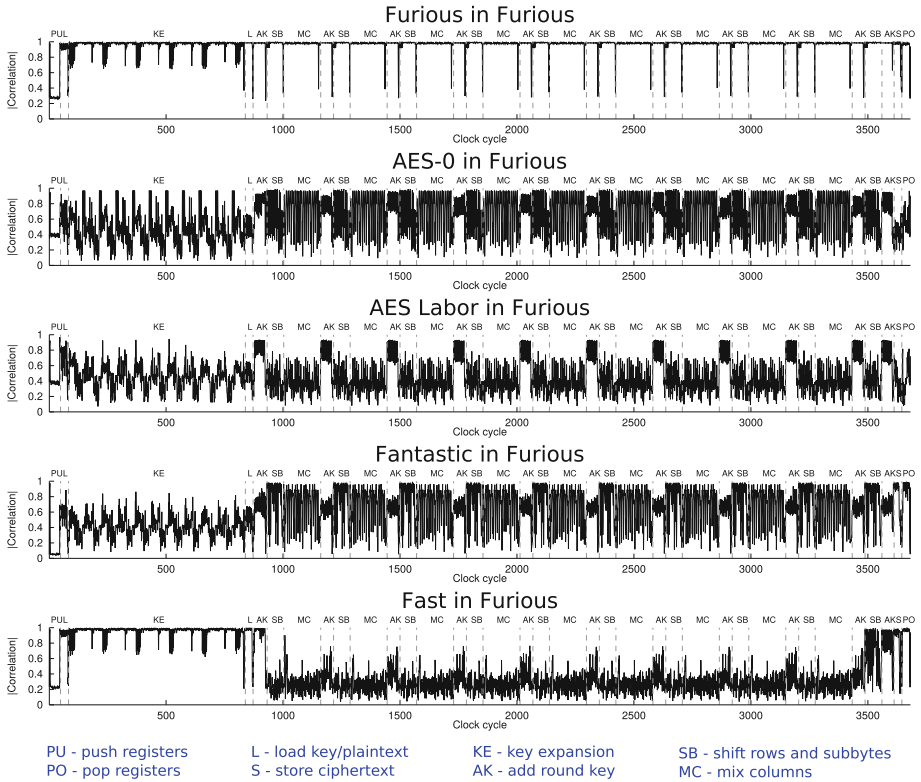


Fig. 3. Maximum projection of all AES implementations onto *Furious*.

two different sets of traces. The maximum projection for *Furious* is the highest when it is compared to itself. The graphs also suggest that *Fast* and *Furious* implementations have identical key expansion algorithm. The high plateaus in *AES Labor* are executions of AddRoundKey. The correlation at the same times is still significant for all other implementations except for *Fast*, for which only the very first and the very last key additions shows high correlated sequences. By projecting a known implementation onto unknown implementation, we can uncover the structure of the unknown implementations, as shown in Fig. 7.

Maximum projection graphs can be summarized by computing the mean absolute correlation coefficient. The resulting number indicates the similarity between the implementations. Table 1 shows the mean absolute correlation coefficients for all our test AES implementations. Identical implementations have the mean correlation close to 1.0. Similar implementations, such as *AES-0*, *Furious*, and *Fantastic* result in high mean correlation coefficients. Even though *Fast* is very different from the other implementations, its intermediate values are similar enough to produce a significant correlation, so that we can conclude that *Fast*

Table 1. Mean absolute correlation of projecting traces of each implementation (row) into traces of each other implementation (column).

	AES-0	AES Labor	Furious	Fast	Fantastic
AES-0	0.97	0.41	0.63	0.33	0.53
AES Labor	0.42	0.91	0.46	0.29	0.39
Furious	0.61	0.44	0.96	0.45	0.54
Fast	0.35	0.32	0.46	0.96	0.29
Fantastic	0.58	0.40	0.62	0.30	0.93

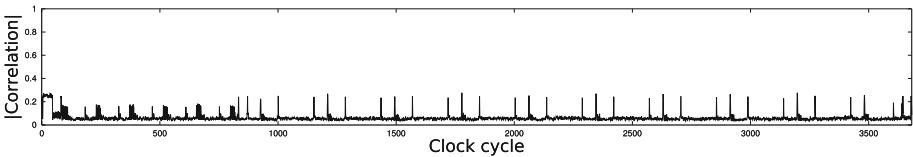


Fig. 4. Absolute correlation of *Furious* with *Furious* when non-matching data was used. Total mean absolute correlation equals to 0.061. High peaks happen when constants are processed (e.g. loading a constant into a register).

is performing an AES encryption. In contrast, when different data is processed, the mean absolute correlation is low, as shown in Fig. 4.

To reveal more details about a suspicious implementation at hand, the maximum projection graph can also be summarized by computing the mean absolute correlation for each known operation, as shown in Table 2. Here we can see which

Table 2. Mean absolute correlation of projecting similarity matrices of all AES implementations into the implementations denoted by symbol “→”. Numbers in bold signify identical code. AK-*AddRoundKey*, SB-*SubBytes*, MC-*MixColumns*, KE-*KeyExpansion*.

	AK	SB	MC	KE	AK	SB	MC	KE	AK	SB	MC	KE
AES-0	0.96	0.97	0.98	0.97	0.68	0.31	0.38	0.40	0.71	0.65	0.71	0.46
AES Labor	0.64	0.33	0.36	0.43	0.96	0.97	0.96	0.88	0.75	0.40	0.37	0.45
Furious	0.68	0.65	0.73	0.46	0.73	0.38	0.40	0.41	0.95	0.98	0.98	0.96
Fast	0.45	0.31	0.26	0.44	0.48	0.24	0.19	0.39	0.47	0.31	0.27	0.95
Fantastic	0.64	0.58	0.75	0.41	0.62	0.31	0.37	0.43	0.65	0.72	0.68	0.41

(a) → *AES-0*

(b) → *AES Labor*

(c) → *Furious*

	AK	KE	R	AK	SB	MC	KE
AES-0	0.69	0.46	0.28	0.66	0.57	0.75	0.33
AES Labor	0.73	0.45	0.23	0.62	0.32	0.35	0.40
Furious	0.85	0.95	0.27	0.62	0.71	0.70	0.32
Fast	0.97	0.95	0.98	0.43	0.27	0.25	0.31
Fantastic	0.64	0.40	0.25	0.96	0.96	0.97	0.90

(d) → *Fast*

(e) → *Fantastic*

subparts of the code are similar or even identical. For example, we know that the key expansion of *Fast* and *Furious* is identical; and that *AES-0* and *Fantastic* have similar *MixColumns*.

3.2 Modified Plagiarized Code

To simulate an attacker who actively manipulates plagiarized code, we have selected *Furious* as our genuine implementation and modified it in various ways, cf. [4, 12].

Address modification (addr). In this attack, we change the usage of all registers of the *Furious* implementation. For example, register `r8` is consistently used instead of `r0`. In addition, the variables for the key, the expanded key, the plaintext, and the ciphertext are stored in different locations in the SRAM. Finally, the constants `sbox` and `xtime` are moved to different flash addresses.

Instruction reordering (swap). In this attack, we change the order of individual instructions, and also swap entire blocks of instruction without changing the number of clock cycles required to perform the encryption. Concretely, we change the order of loading and saving the key, plaintext, and ciphertext; the order of applying XOR in `AddRoundKey`; the order of rows when computing `ShiftRows` and `SubBytes`; and the order of applying XORs in the key scheduler. On several occasions, in order to change the sequence of loading variables, we change instructions such as, e.g., `LD R0, Z+` (opcode 9001) into instructions like `LDD R0, Z+0` (opcode 8000).

Combination of addr and swap (addr+swap). Here, we combine the address changes and the swapping of the instructions.

Insert dummy NOPs (dummy). This version has NOP instructions inserted throughout the code: in the key expansion, in `SubBytes`, `ShiftRows`, and `MixColumns`. As a result, the encryption needs 792 additional clock cycles to finish.

Insert other dummy instructions (dummy smart). The problem with NOP instructions is that on the ATmega163 they can be easily distinguished because they have low power consumption in comparison to the other instructions. To make the power consumption appear more genuine, we insert dummy instructions that manipulate the state for a short amount of time, and change it back before resuming with encryption. Figure 8 shows several assembly macros that we spread throughout the code that reuse intermediate values of the implementation. This version also has 792 additional clock cycles.

All attacks combined (dummy smart+addr+swap). This attack introduces 792 smart dummy cycles and combines them with the `add+swap` attack.

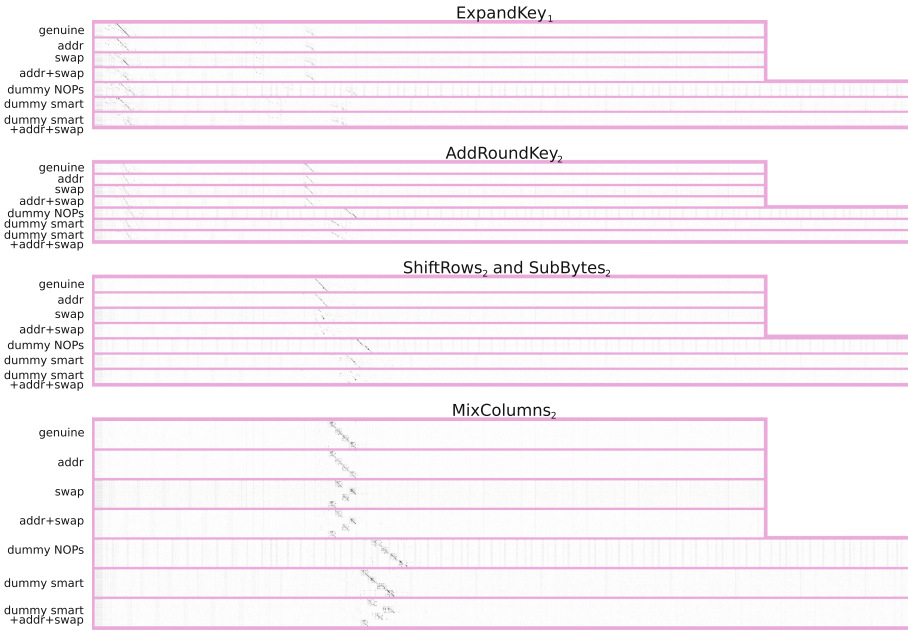


Fig. 5. Similarity matrices of plagiarism attacks on RijndaelFurious (denoted by “genuine”). X and Y axes are measured in clock cycles.

Visual Inspection. Figure 5 shows excerpts of the similarity matrices that correspond to the genuine implementation along the Y-axis, and the manipulated implementations along the X-axis for selected AES functions. Figures 11, 12, 13, 14, 15 and 16 show the full similarity matrices. Code with manipulations that do not change the order of instructions can be well recognized because of the long lines along the diagonals of the matrices.

Maximum Projection. Figures 6 and 9 show the maximum projection method applied on similarity matrices of *Furious* with all the other implementations. The versions *addr* and *swap* produce the largest impact on the maximum projection method, however, we are still able to distinguish MixColumns quite well. On the other hand, despite introducing 792 dummy clock cycles, we can argue with a high confidence that the presented system contains large portions of our genuine implementation.

Table 3 shows the total mean absolute correlation coefficients and the operation-wise absolute mean correlation coefficient for *Furious* and its modified versions. Here we can see that our method is well-suited for finding versions of our code with added dummy instructions—even 792 additional dummy clock cycles have little effect on the mean absolute correlation, which, at >0.8 is high

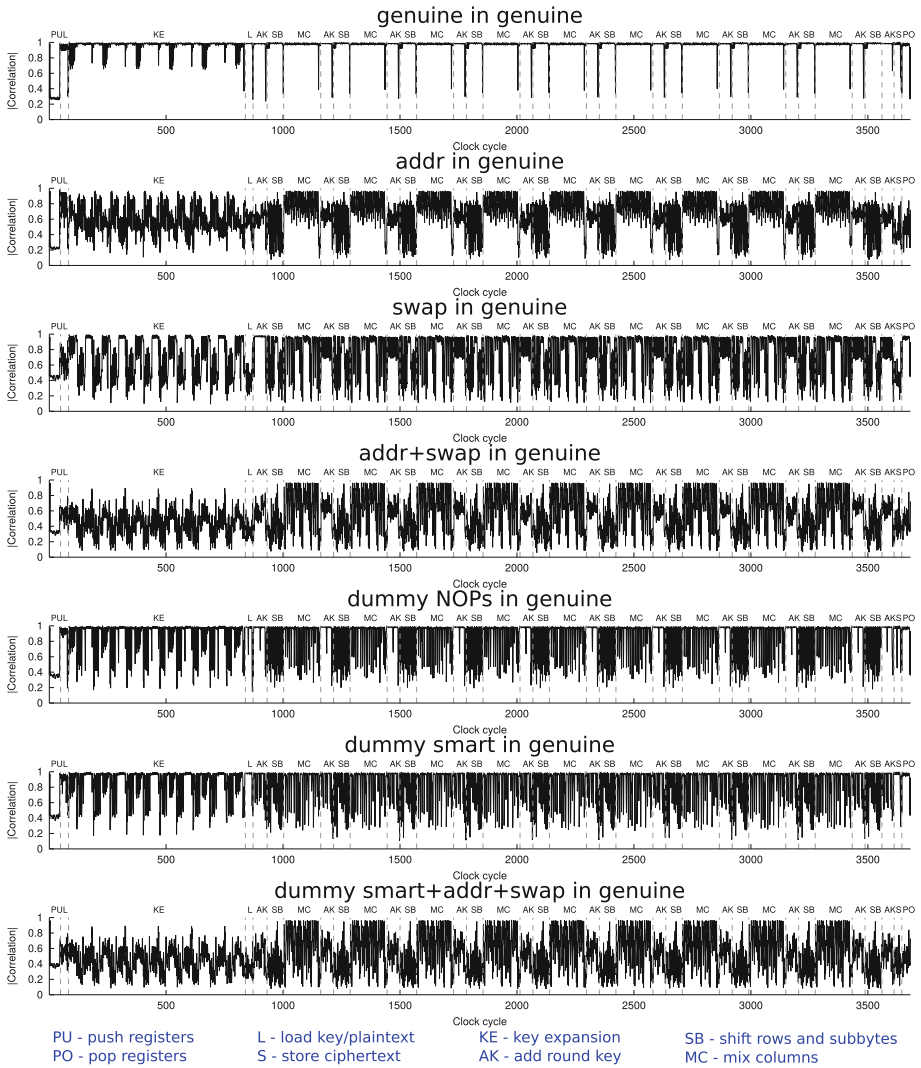


Fig. 6. Maximum projection of modified implementations onto genuine *Furious*.

enough to suspect a plagiarism. On the other hand, by exchanging registers and choosing different addresses for variables and constants in the memory, as in *addr*, the attacker can decrease the mean absolute correlation by a large margin. However, after cross-referencing the similarity matrix, as shown in Fig. 11, we see an almost continuous line along its diagonal, implying an identical implementation. Swapping code chunks does not help much, since the individual parts are still well-recognizable in the operation-wise mean correlation table and in

Table 3. (a) Total mean absolute correlation of modified *Furious* implementations projected into the genuine *Furious* implementation. (b) Operation-wise mean absolute correlation of modified *Furious* implementations projected into the genuine *Furious* implementation.

	genuine	AK	SB	MC	KE
genuine	0.96	0.95	0.98	0.98	0.96
addr	0.64	0.61	0.52	0.76	0.60
swap	0.73	0.84	0.62	0.78	0.80
addr+swap	0.52	0.59	0.37	0.64	0.45
dummy NOPs	0.84	0.92	0.72	0.87	0.86
dummy smart	0.83	0.82	0.75	0.85	0.85
dummy smart+addr+swap	0.51	0.54	0.36	0.63	0.44

(a) (b)

the similarity matrix. Thus, in order to successfully avoid raising suspicion, an attacker will have to use a plethora of countermeasures at a much finer level, which requires a considerable amount of effort to obfuscate the code without introducing bugs and unintended side effects.

4 Discussion

Our work is built upon the assumption that identical code produces almost identical side channel leakage on an identical (but physically distinct) microcontroller platform. In our experiments we studied five different AES implementations and observed significant correlation coefficients in each similarity matrix. This is due to the fact that all test programs are implementations of the same task and process the same data. However, as exemplarily shown in Fig. 3, the maximum projection of identical implementations clearly stands out when compared to different implementations, and thereby underlines that the same sequence of instructions is clearly highlighted and allows us to detect plagiarized software of a passive adversary with high accuracy.

In the case when an adversary has spent some effort modifying the original code, we come to the conclusion that our method achieves a good robustness against changed registers and added dummy cycles. The second case is a clear advantage compared to the method of Durvaux et al. [4] that turned out to be very sensitive to the addition of 57 dummy clock cycles, whereas we worked with 792 additional dummy cycles. In addition, in contrast to the method of Durvaux et al., we are not forced to cut the recorded traces to have the same length and are able to use all recorded information. The robustness of our approach to adding smart dummy instructions is only slightly worse compared to dummy NOPs and for many code segments the original sequence is still revealed in the maximum projection. Similar observations hold when instructions are swapped.

In this paper we have used static, time-aligned, and constant-time applications to test our approach. However, we are confident that our methods are also applicable in a more general setting, e.g., with code branches and nondeterministic parts. A further adversary strategy could be to implement dynamic code generation on the microcontroller as, e.g., proposed by [12] which will make our approach more difficult if the frequency of run-time code generation is high. As result, an adversary is forced to combine many kinds of modifications and put a considerable amount of effort in order to reduce successful plagiarism detection that usually lead to enhanced code complexity and execution time.

Further promising properties of our approach can be seen in Fig. 4. Even if the data of the two implementations do not match, our approach reveals smaller but still significant correlation signals. These correlation signals are assumed to be due to code similarities and lead to the conjecture that our methods are still able to detect the similarity of two programs if there is a lack of any input data channel. Interestingly, this observation gives reasons to assume that even the implementation of an intrinsic data masking scheme by the adversary might be not sufficient.

Thinking from a higher-level perspective, our tools detect whether a program processes identical data or not. Hereby, patent infringements on the algorithm level may be possible to detect. For example, an unlicensed use of a patented algorithm with a specific data processing can be proven using the data dependent side-channel leakage based on our approach.

5 Conclusions

We have presented and evaluated new methods based on characteristic data leakage for detecting software plagiarism on a microcontroller platform. The conducted experiments give evidence that these methods are highly robust to many different code transformations and that the resulting pattern of data leakage of the microcontroller executing the code can be considered as its own watermark. Promising research directions are opened for connecting horizontal and vertical dimensions for code sequence analysis through side channel traces.

Acknowledgement. This work has been supported in parts by the German Federal Ministry of Education and Research (BMBF) through the project DePlagEmSoft, FKZ 03FH015I3.

6 Appendix

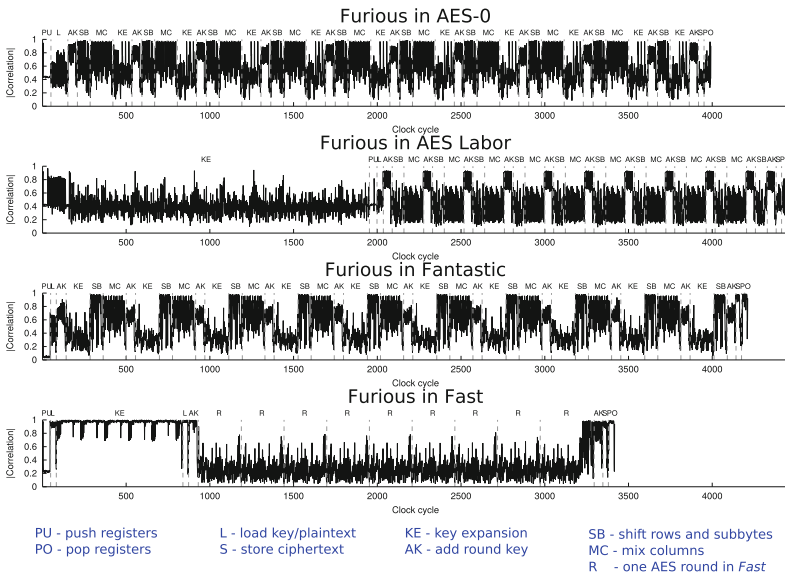


Fig. 7. Maximum projection of *Furious* onto all other AES implementations.

```

① INC \reg
   DEC \reg

② NEG \reg
   NEG \reg

③ ROL \reg
   ROR \reg

④ PUSH \tmp
   LDI \tmp, \c
   EOR \reg, \tmp
   POP \tmp

⑤ LDI ZL, 0x00
   LPM \tmp, Z

⑥ EOR \tmp, \tmp

⑦ PUSH \reg1
   PUSH \reg2
   PUSH \reg3
   EOR \reg1, \reg2
   EOR \reg2, \reg3
   EOR \reg3, \reg1
   POP \reg3
   POP \reg2
   POP \reg1

⑧ MOV \tmp, \reg ;; save register
   LDI ZH, hi8(hd_tmp)
   LDI ZL, lo8(hd_tmp)
   LD \reg, z
   MOV \reg, \tmp ;; restore register
    
```

Fig. 8. Assembly macros used to insert dummy smart instructions. Macros 1,2,3 change the content of a chosen register in one clock cycle, and change it back in the next one. Macro 8 is used to remove Hamming-distance leakage between consecutive SRAM reads or writes by performing a dummy read in the SRAM at some constant address. Macro 5 is used before some of the sbx lookups in the flash memory. Macro 6 is applied to an unused register and leaks data from preceding operations that have use the ALU (arithmetic-logic unit). Macro 4 loads a random constant value chosen at compile time into a register and restores the register right after that. Macro 7 uses XORs on three selected registers and immediately restores them to their respective original values.

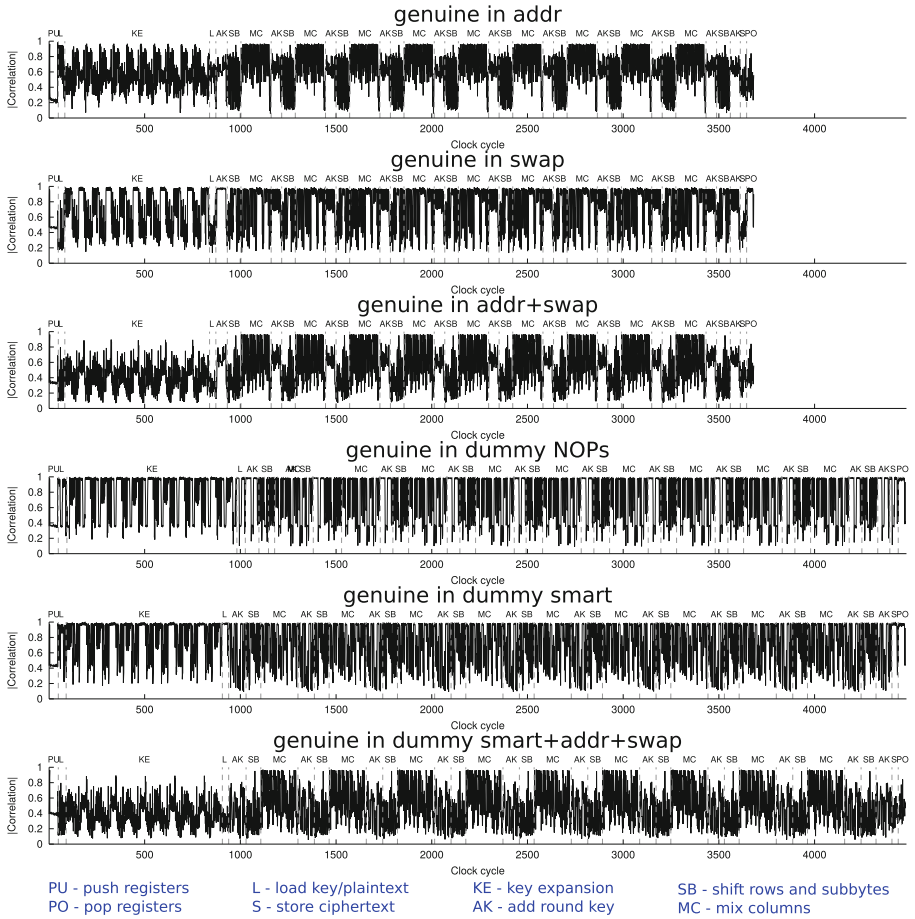


Fig. 9. Maximum projection of genuine *Furious* onto all modified AES implementations.

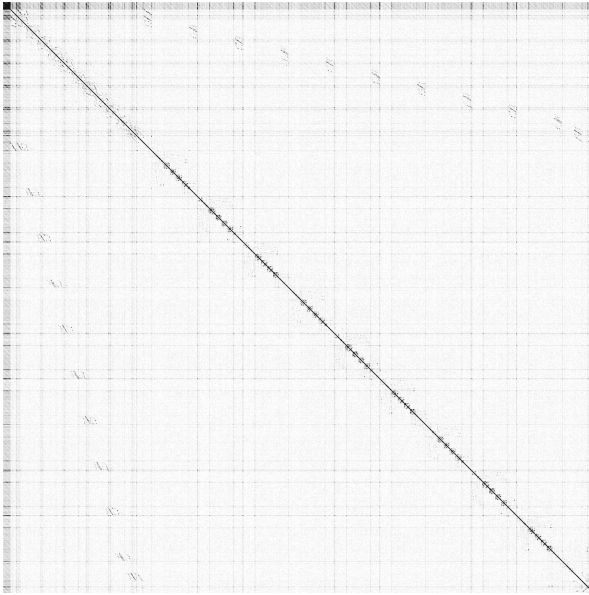


Fig. 10. Similarity matrix of *Furious* with itself.

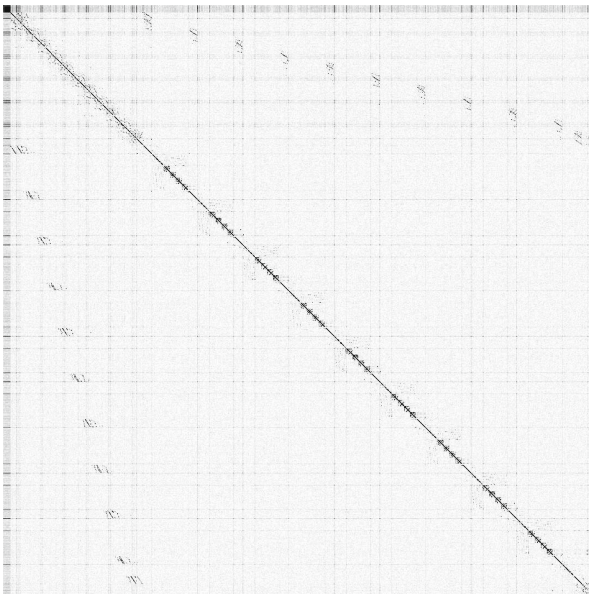


Fig. 11. Similarity matrix of *addr* and the genuine *Furious* AES implementations.

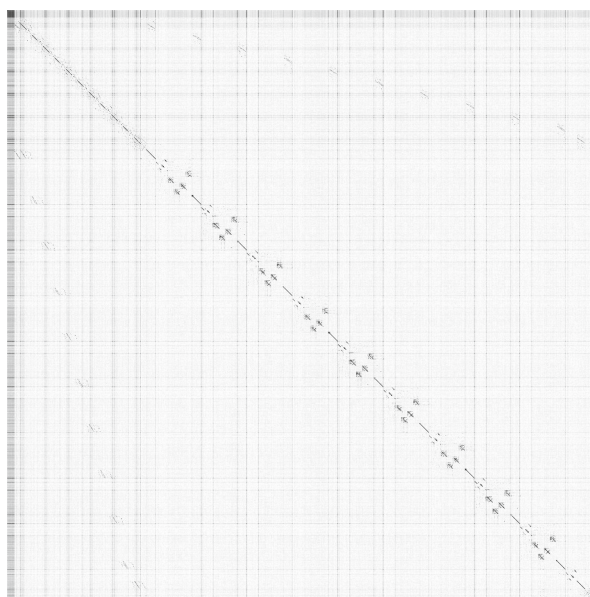


Fig. 12. Similarity matrix of *swap* and the genuine *Furious* AES implementations.

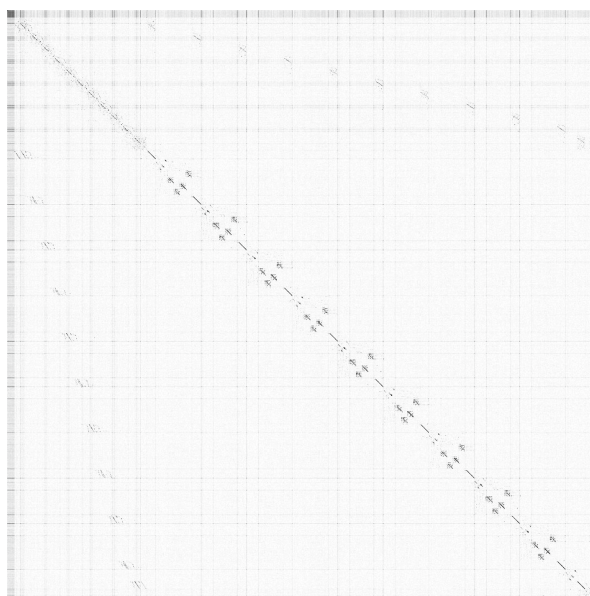


Fig. 13. Similarity matrix of *addr+swap* and the genuine *Furious* AES implementations.

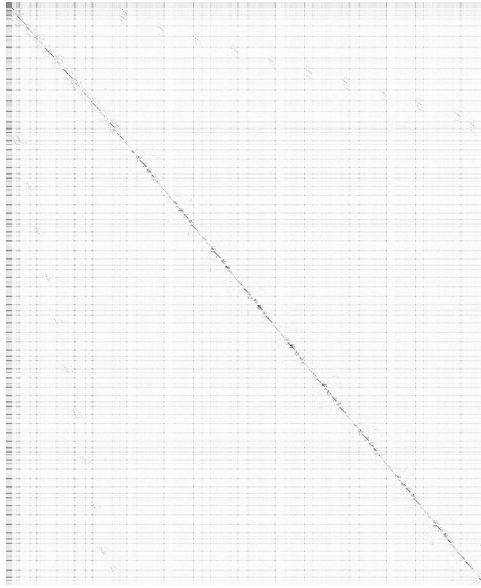


Fig. 14. Similarity matrix of *dummy NOPs* and the genuine *Furious* AES implementations.

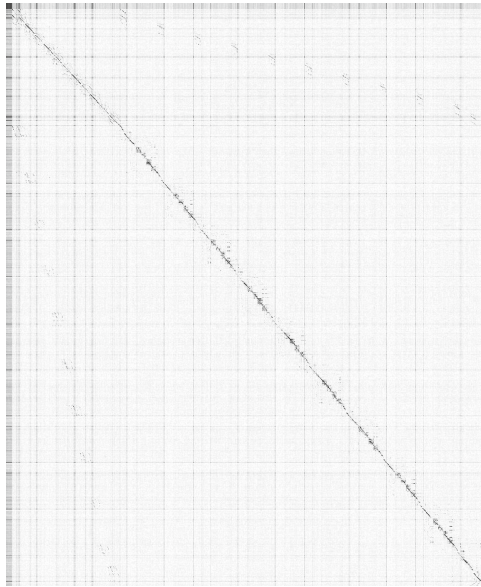


Fig. 15. Similarity matrix of *dummy smart* and the genuine *Furious* AES implementations.

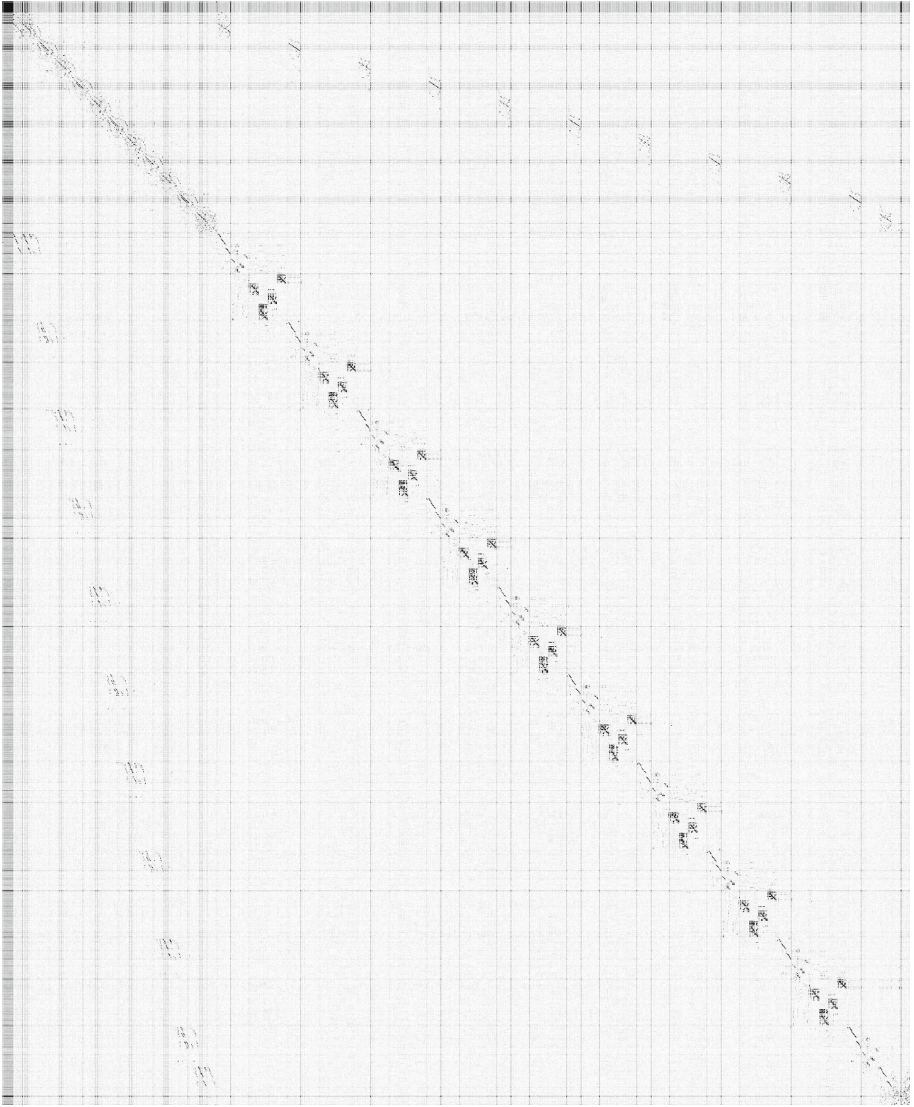


Fig. 16. Similarity matrix of *dummy smart+addr+swap* and the genuine *Furious* AES implementations.

References

1. Becker, G.T., Burleson, W., Paar, C.: Side-channel watermarks for embedded software. In: 9th IEEE NEWCAS Conference (NEWCAS 2011) (2011)
2. Becker, G., Strobel, D., Paar, C., Burleson, W.: Detecting software theft in embedded systems: a side-channel approach. *IEEE Trans. Inf. Forensics Secur.* **7**(4), 1144–1154 (2012)

3. Strobel, D., Bache, F., Oswald, D., Schellenberg, F., Paar, C.: SCANDALee: a side-ChANnel-based DisAssemblEr using local electromagnetic emanations. In: Design, Automation, and Test in Europe (DATE), 9–13 March 2015 (2015)
4. Durvaux, F., Gérard, B., Kerckhof, S., Koeune, F., Standaert, F.-X.: Intellectual property protection for integrated systems using soft physical hash functions. In: Lee, D.H., Yung, M. (eds.) WISA 2012. LNCS, vol. 7690, pp. 208–225. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35416-8_15
5. Kocher, P., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_25
6. Kerckhof, S., Durvaux, F., Standaert, F.-X., Gerard, B.: Intellectual property protection for FPGA designs with soft physical hash functions: first experimental results. In: 2013 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp. 7–12, June 2013
7. Mangard, S., Oswald, E., Popp, T.: Power Analysis Attacks Revealing the Secrets of Smart Cards. Springer, New York (2007). <https://doi.org/10.1007/978-0-387-38162-6>
8. Atmel: ATmega163(L) Datasheet (revision E), February 2003
9. Atmel: Atmel AVR 8-bit Instruction Set Manual (revision 0856J), July 2014
10. Otte, D.: Avr-crypto-lib. <https://www.das-labor.org/wiki/AVR-Crypto-Lib/en>. Accessed Sept 2017
11. Poettering, B.: AVRAES: the AES block cipher on AVR controllers. <http://point-at-infinity.org/avraes/>. Accessed Sept 2017
12. Couroussé, D., Barry, T., Robisson, B., Jaillon, P., Potin, O., Lanet, J.-L.: Runtime code polymorphism as a protection against side channel attacks. In: Foresti, S., Lopez, J. (eds.) WISTP 2016. LNCS, vol. 9895, pp. 136–152. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45931-8_9



Secure Number Theoretic Transform and Speed Record for Ring-LWE Encryption on Embedded Processors

Hwajeong Seo¹, Zhe Liu², Taehwan Park³, Hyeokchan Kwon⁴, Sokjoon Lee⁴,
and Howon Kim³(✉)

¹ Department of IT, Hansung University, 116 Samseongyoro-16gil,
Seongbuk-gu, Seoul 136-792, Republic of Korea

hwajeong@hansung.ac.kr

² APSIA, Interdisciplinary Centre for Security, Reliability and Trust (SnT),
University of Luxembourg, Luxembourg City, Luxembourg

sduliuzhe@gmail.com

³ School of Computer Science and Engineering, Pusan National University,
San-30, Jangjeon-Dong, Geumjeong-Gu, Busan 609-735, Republic of Korea

{pth5804, howonkim}@pusan.ac.kr

⁴ System Security Research Group,
Electronics and Telecommunications Research Institute, Daejeon 34129, Korea
{hckwon, junny}@etri.re.kr

Abstract. Compact implementations of the ring variant of the Learning with Errors (Ring-LWE) on the embedded processors have been actively studied due to potential quantum threats. Various Ring-LWE implementation works mainly focused on optimization techniques to reduce the execution timing and memory consumptions for high availability. For this reason, they failed to provide secure implementations against general side channel attacks, such as timing attack. In this paper, we present secure and fastest Ring-LWE encryption implementation on low-end 8-bit AVR processors. We targeted the most expensive operation, i.e. Number Theoretic Transform (NTT) based polynomial multiplication, to provide countermeasures against timing attacks and best performance among similar implementations till now. Our contributions for optimizations are concluded as follows: (1) we propose the Look-Up Table (LUT) based fast

This research of Hwajeong Seo was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2017-2014-0-00743) supervised by the IITP (Institute for Information & communications Technology Promotion). This work of Hyeokchan Kwon and Sokjoon Lee was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT). [B0717-16-0097, Development of V2X Service Integrated Security Technology for Autonomous Driving Vehicle]. This research of Taehwan Park and Howon Kim was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2012-0-00265, Development of high performance IoT device and Open Platform with Intelligent Software).

reduction techniques for speeding up the modular coefficient multiplication in regular fashion, (2) we use the modular addition and subtraction operations, which are performed in constant timing. With these optimization techniques, the proposed NTT implementation enhances the performance by 18.3–22% than previous works. Finally, our Ring-LWE encryption implementations require only 680,796 and 1,754,064 clock cycles for 128-bit and 256-bit security levels, respectively.

Keywords: Ring learning with errors · Software implementation
Public key encryption · 8-bit AVR · Number theoretic transform
Discrete gaussian sampling · Timing attack

1 Introduction

Classic public key cryptography algorithms such as RSA and Elliptic Curve Cryptography (ECC) are built based on integer factorization and discrete logarithm problems, which are believed to be secure against classical computer environments with properly chosen parameters. For this reason, a number of works focused on compact implementations of RSA and ECC [5, 6, 8–10, 12, 13, 17, 21–24]. However, such hard problems can be solved using Shor’s algorithm on a sufficient large quantum computer in polynomial time [25]. To defeat potential attacks and threats, lattice-based cryptography is considered as one of the most promising candidates for post-quantum cryptography. Lattice-based cryptography is built based on worst-case computational assumptions in lattices that would remain hard even for quantum computers. Furthermore, the emerging Internet of Things (IoT) technology introduces new computing environments including all kinds of sensors, actuators, meters, consumer electronics, medical monitors, household appliances and vehicles. Since these devices are very resource-constrained in terms of computing power, power supply and memory resources, implementing public-key cryptographic algorithms on low-end 8-bit processors poses a big challenge. Therefore, it is necessary to further study the post-quantum cryptosystems on the low-end IoT devices.

The introduction of Learning with Errors (LWE) problem and its ring variant (Ring-LWE) [14, 18] provide efficient ways to build lattice-based public key cryptosystems. The following software implementations of Ring-LWE based public-key encryption or digital signature schemes improved performance and memory requirements: Oder et al. presented an efficient implementation of Bimodal Lattice Signature Scheme (BLISS) on a 32-bit ARM Cortex-M4F microcontroller [15]. De Clercq et al. implemented Ring-LWE encryption scheme on the identical ARM processors [3]. They utilized 32-bit registers to retain two 13–14 coefficients. Boorghany et al. implemented a lattice-based cryptographic scheme on an 8-bit processor for the first time in [1, 2]. The authors evaluated four lattice-based authentication protocols on both 8-bit AVR and 32-bit ARM processors. In particular, Fast Fourier Transform (FFT) transform and Gaussian sampler function are implemented. In LATINCRYPT’15, Pöppelmann et al. studied and compared implementations of Ring-LWE encryption and BLISS on an 8-bit Atmel

ATxmega128 microcontroller [16]. In CHES'15, Liu et al. optimized implementations of Ring-LWE encryption by presenting efficient modular multiplication, NTT computation and refined memory access schemes to achieve high performance and low memory consumption [11]. They presented two implementations of Ring-LWE encryption scheme for both medium-term and long-term security levels on an 8-bit AVR processor. Liu et al. presented the first secure Ring-LWE encryption and BLISS signature implementations against timing attack [7]. NTT and sampling computations are implemented in constant time to prevent timing attack. Particularly, modular reduction is performed in Montgomery reduction to reduce computation complexity. Recently, in [4,9], high efficient implementations on ARM-NEON and MSP430 processors are also covered.

1.1 Contributions

This paper continues the line of research on the secure and compact implementations of the Ring-LWE encryption scheme on low-end 8-bit AVR processor. Core contributions are the techniques to prevent information leakage and optimizations to improve real-world performance of Ring-LWE encryption scheme.

In particular, we focused on the optimization of Number Theoretic Transform (NTT) based polynomial multiplication, which is the most expensive computation in the Ring-LWE. In NTT computation, a number of modular arithmetic operations are required and optimization of modular reduction is highly related with performance. To accelerate performance, we use Look Up Table (LUT) based fast reduction techniques for modular coefficient multiplication. Modular addition and subtraction operations are also implemented in constant time and incomplete representation. To optimize the performance in assembly level, NTT routines fully utilize general purpose registers in the target processors.

Based on the above NTT optimization techniques, we present secure and compact implementations of Ring-LWE encryption scheme on an low-end 8-bit AVR processor. All operations are designed to prevent the timing attack. The implementation only requires 681K and 1,754K clock cycles for 128-bit and 256-bit security level encryption respectively.

The rest of this paper is organized as follows. In Sect. 2, we recall background of Ring-LWE encryption scheme, NTT algorithm, and previous implementation techniques for NTT algorithm. In Sect. 3, we present optimization techniques for NTT on low-end 8-bit AVR processors. In particular, we propose techniques to prevent information leakage through timing and reduce execution time of NTT algorithm. In Sect. 4, we report performance of our implementation and compare with the state-of-the-art NTT and Ring-LWE encryption on the low-end 8-bit AVR platforms. Finally, we conclude the paper in Sect. 5.

2 Background

2.1 Ring-LWE Encryption Scheme

In 2010, Lyubashevsky et al. proposed an encryption scheme based on a more practical algebraic variant of LWE problem defined over polynomial rings

$R_q = \mathbb{Z}_q[\mathbf{x}]/\langle f \rangle$ with an irreducible polynomial $f(x)$ and a modulus q . In Ring-LWE problem, elements a, s and t are polynomials in the ring R_q . Ring-LWE encryption scheme proposed by Lyubashevsky et al. was later optimized in [20]. Roy et al.'s variant aims at reducing the cost of polynomial arithmetic. In particular, the polynomial arithmetic during a decryption operation requires only one Number Theoretic Transform (NTT) operation. Beside this computational optimization, the scheme performs sampling from the discrete Gaussian distribution using a Knuth-Yao sampler. In next subsection, we will first present mathematical concepts of NTT and Knuth-Yao sampling operations, then we will describe the steps used in the Roy et al.'s version of the encryption scheme.

Now, we describe steps applied in the encryption scheme proposed by Roy et al. [20]. We denote the NTT of a polynomial a by \tilde{a} .

- Key generation stage **Gen**(\tilde{a}): Two error polynomials $r_1, r_2 \in R_q$ are sampled from the discrete Gaussian distribution \mathcal{X}_σ by applying the Knuth-Yao sampler twice.

$$\tilde{r}_1 = NTT(r_1), \tilde{r}_2 = NTT(r_2)$$

and then an operation $\tilde{p} = \tilde{r}_1 - \tilde{a} \cdot \tilde{r}_2 \in R_q$ is performed. Public key is polynomial pair (\tilde{a}, \tilde{p}) and private key is polynomial \tilde{r}_2 .

- Encryption stage **Enc**(\tilde{a}, \tilde{p}, M): The input message $M \in \{0, 1\}^n$ is a binary vector of n bits. This message is first encoded into a polynomial in the ring R_q by multiplying the bits of message by $q/2$. Three error polynomials $e_1, e_2, e_3 \in R_q$ are sampled from \mathcal{X}_σ . The ciphertext is computed as a set of two polynomials $(\tilde{C}_1, \tilde{C}_2)$:

$$(\tilde{C}_1, \tilde{C}_2) = (\tilde{a} \cdot \tilde{e}_1 + \tilde{e}_2, \tilde{p} \cdot \tilde{e}_1 + NTT(e_3 + M'))$$

- Decryption stage **Dec**($\tilde{C}_1, \tilde{C}_2, \tilde{r}_2$): One inverse NTT is performed to recover M' :

$$M' = INTT(\tilde{r}_2 \cdot \tilde{C}_1 + \tilde{C}_2)$$

and then a decoder is used to recover the original message M from M' .

2.2 Number Theoretic Transform

We use the Number Theoretic Transform (NTT) to perform polynomial multiplication. NTT can be seen as a discrete variant of Fast Fourier Transform (FFT) but performs in a finite ring \mathbb{Z}_q . Instead of using the complex roots of unity, NTT evaluates a polynomial multiplication $a(x) = \sum_{i=0}^{n-1} a_i x^i \in \mathbb{Z}_q$ in the n -th roots of unity ω_n^i for $i = 0, \dots, n - 1$, where ω_n denotes a primitive n -th root of unity. Algorithm 1 shows the iterative version of NTT algorithm.

The iterative NTT algorithm consists of three nested loops. The outermost loop (i -loop) starts from $i = 2$ and increases by doubling i , and the loop stops when $i = n$, thus it has only $\log_2 n$ iterations. In each iteration, the value of twiddle factor ω_i are computed by executing a power operation $\omega_i = \omega_n^{n/i}$, and the

Algorithm 1. Iterative Number Theoretic Transform

Require: A polynomial $a(x) \in \mathbb{Z}_q[x]$ of degree $n - 1$ and n -th primitive $\omega \in \mathbb{Z}_q$ of unity

Ensure: Polynomial $a(x) = NTT(a) \in \mathbb{Z}_q[x]$

```

1:  $a = BitReverse(a)$ 
2: for  $i$  from 2 by  $i = 2i$  to  $n$  do
3:    $\omega_i = \omega_n^{n/i}, \omega = 1$ 
4:   for  $j$  from 0 by 1 to  $i/2 - 1$  do
5:     for  $k$  from 0 by  $i$  to  $n - 1$  do
6:        $U = a[k + j]$ 
7:        $V = \omega \cdot a[k + j + i/2]$ 
8:        $a[k + j] = U + V$ 
9:        $a[k + j + i/2] = U - V$ 
10:     $\omega = \omega \cdot \omega_i$ 
11: return  $a$ 

```

value of ω is initialized by 1. Compared to i -loop, the j -loop executes more iterations, the number of iteration can be seen as a sum of a geometric progression for 2^i where i starts from 0 and has a maximum value of $\log_2(n - 1)$, thus, the j -loop has $n - 1$ iterations. In each iteration of j -loop, the twiddle factor ω is updated by performing a coefficient modular multiplication. Apparently, the innermost loop (k -loop) occupies most part of the execution time of NTT algorithm since it is executed roughly $\frac{n}{2} \log_2 n$ times. In each iteration of the innermost loop, two coefficients $a[i + j]$ and $a[i + j + i/2]$ are loaded from memory into registers, and then $a[i + j + i/2]$ are multiplied by the twiddle factor ω , after that, the value of $a[k + j]$ and $a[k + j + i/2]$ are updated and stored in the memory.

2.3 Previous Implementations of NTT

In LATINCRYPT'15, Pöppelmann et al. optimized the NTT operation by merging inverse NTT and multiplication by powers of ψ^{-1} . Furthermore, bit-reversal step is removed by the manipulation of the standard iterative algorithms. In CHES'15, Liu et al. suggested the high-speed NTT operations with efficient coefficient modular multiplication [11]. They presented the Move-and-Add (MA) method to perform the 16-bit wise coefficient multiplication and the Shift-Add-Multiply-Subtract-Subtract (SAMS2) techniques to replace the expensive reduction operations with the MUL instructions by cheaper shift and addition instructions. In TECS'17, Liu et al. improved the modular reduction by using Montgomery reduction [7]. This improves the previous SAMS2 techniques when the case requires a number of shift and addition operations on low-end devices. The new technique ensures the constant time computation together with high performance.

3 Proposed Methods

NTT computation takes up the majority of the execution time on modular multiplication operation since it is performed in the innermost k -loop. The 16-bit wise multiplication requires only 4 8-bit wise multiplication operations and this is already well covered in previous works [11]. Thus, the optimization of fast reduction operation is a prerequisite for high-speed implementation of NTT algorithm. We chose the prime modulus $q = 7681$ (i.e. `0x1e01` in hexadecimal representation) and $q = 12289$ (i.e. `0x3001` in hexadecimal representation) for the target parameters, which are used in previous works [7, 11].

Unlike previous SAMS2 method by [7, 11], we propose an optimized Look-Up Table (LUT) based fast reduction technique for performing the mod 7681 and mod 12289 operations. The main idea is to first reduce the result by using the 8-bit wise pre-computed reduced results, and then perform the tiny fast reduction steps on short coefficients. The results are kept in the incomplete representation in order to optimize the number of subtraction in the reduction step. For the case of prime modulus $q = 7681$, the variables are always kept in range of $(0, 2^{14} - 1)$ in incomplete representations and the intermediate results (IR) of multiplication are kept in $(0, 2^{28} - 1)$. We set two pre-computed LUTs with (mod 7681) operation. One input variable are ranging from 17-th bit to 24-th bit, which are the values located in $(x \times 2^{16})$ where x is ranging from 0 to 2^8 . Afterwards, the variable is reduced to 13-bit wise results through (mod 7681) operation ($\approx ((IR \text{ div } 2^{16}) \text{ mod } 2^8) \text{ mod } 7681$). The other input variable is from 25-th bit to 28-th bit, which are values $(x \times 2^{24})$ where x is ranging from 0 to 2^{41} . The LUT ensures that the variable is reduced to the 13-bit results ($\approx (IR \text{ div } 2^{24}) \text{ mod } 7681$). After two times of LUT based reduction operations, the two 13-bit wise outputs are added to the remaining 16-bit wise intermediate results (1-st–16-th bits), which output 17-bit intermediate results. Afterwards, the tiny fast reduction is performed on the intermediate results. Observing that $2^{13} \equiv 2^9 - 1 \pmod{7681}$, the fast reduction can be performed with 16-bit wise addition (2^9) and 8-bit wise subtraction operations (-1).

The detailed method is described in Fig. 1. We keep the product in four registers ($r3, r2, r1, r0$), which has been marked by different colors. Each of register ($r3, r2, r1, r0$) is 8-bit long. The colorful parts mean that this bit has been occupied while the white part means the current bit is empty. The reduction with 7681 using LUT approach can be performed as follows:

1. LUT access. We first perform the LUT access with variable ($r2$) to get the 13-bit wise reduced results ($s1$ and $s0$). Then, the variable ($r3$) is also reduced to the results ($t1$ and $t0$). Both results are 13-bit wise long and stored in 2 8-bit registers.

¹ Two LUTs only require 1 KB ($2^8 \times 2 + 2^8 \times 2$) and the LUTs are stored in the ROM. Considering that AVR platforms support ROM size in 128, 256, and 384 KB, the ROM consumption of LUT is negligible.

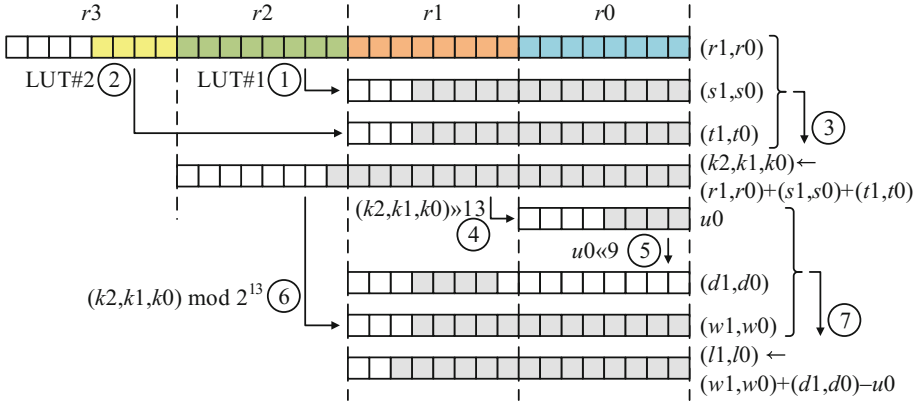


Fig. 1. Look-Up Table based fast reduction for $q = 7681$, ①②: LUT access; ③: addition; ④⑤: shifting; ⑥: modulo; ⑦: addition and subtraction.

2. Addition. We then perform the addition of $(r1, r0) + (s1, s0) + (t1, t0)$. Apparently, the sum result is less than 18-bit, which can be kept in three registers $(k2, k1, k0)$.
3. Shifting. We right shift $(k2, k1, k0)$ by 13-bit to get the result $(u0)$. Afterwards, the value $(u0)$ is left shifted by 9-bit to get the $(d1, d0)$.
4. Modulo. Thereafter, the intermediate results $(k2, k1, k0)$ below 13-bit are extracted and we obtain the $(w1, w0)$.
5. Addition and Subtraction. Finally, we perform the addition and subtraction operations of $(w1, w0) + (d1, d0) - u0$.

In Algorithm 2, the LUT based modular reduction in source code level is described. In Step 1–13, MOV-and-ADD multiplication is used to perform the 16-bit wise multiplication. The 32-bit intermediate results are stored in 4 8-bit registers (R18, R19, R20, R21). In Step 14–15, the address of LUT_1 is loaded to 2 registers (R30, R31). Then, the 17–24-th bits (R20) is added to the address. When the address pointer is ready, the LUT access is performed. From Step 22 to 29, the 25–28-th bits (R21) are used to access the LUT_2. Afterwards the results are reduced. In Step 30–31, two 13-bit LUT results are added. Afterwards, the summation is added to the intermediate results. From Step 35 to 45, tiny fast reduction is performed on 17-bit intermediate results with 16-bit wise addition and 8-bit wise subtraction operations.

Since the LUT approach is generic approach for any primes, proposed LUT based approach is also available in the case of mod 12289. Two differences are LUT value and final step (tiny fast reduction). We need to construct the (mod 12289)'s LUT. For the final step, we perform the tiny fast reduction with modulus equation $(2^{14} \equiv 2^{12} - 1 \text{ mod } 12289)$. The detailed descriptions are drawn in Fig. 2. We execute two LUT and one tiny final reduction. After the tiny fast reduction, it outputs 16-bit results and this can incur the overflow in following operations. We perform the fast reduction once again to fit the results within

Algorithm 2. LUT based modular reduction in source code (mod 7681)

Input: operands R22, R23, R24, R25		24: ADD R30, R21
Output: results {R24, R25}		25: ADC R31, R26
1: CLR R26	{MOV-and-ADD}	26: ADD R30, R21
2: MUL R24, R22		27: ADC R31, R26
3: MOVW R18, R0		
4: MUL R25, R23		28: LPM R24, Z+
5: MOVW R20, R0		29: LPM R25, Z+
6: MUL R24, R23		30: ADD R24, R22
7: ADD R19, R0		31: ADC R25, R23
8: ADC R20, R1		
9: ADC R21, R26		32: ADD R24, R18
		33: ADC R25, R19
		34: ADC R26, R26
10: MUL R25, R22		
11: ADD R19, R0		
12: ADC R20, R1		35: MOV R20, R25{tiny fast reduction}
13: ADC R21, R26		36: ANDI R25, 0X1F
14: LDI R30, lo8(LUT.1)	{LUT access}	37: LSR R26
15: LDI R31, hi8(LUT.1)		38: ROR R20
16: ADD R30, R20		39: SWAP R20
17: ADC R31, R26		40: ANDI R20, 0X0F
18: ADD R30, R20		
19: ADC R31, R26		41: SUB R24, R20
		42: SBC R25, R26
20: LPM R22, Z+		
21: LPM R23, Z+		43: LSL R20
		44: ADD R25, R20
22: LDI R30, lo8(LUT.2)	{LUT access}	45: CLR R1
23: LDI R31, hi8(LUT.2)		

15-bit. By leaving the most significant bit in the register, addition and subtraction operations do not need to check whether the intermediate results generate the overflow/underflow or not.

Constant Modular Addition and Subtraction. To prevent timing attacks, modular addition and subtraction operations should be implemented in constant time. We used the incomplete representation and unsigned type for variable format. The results are always kept in 2 bytes and positive values. The detailed descriptions are available in Algorithm 3. First addition or subtraction operation is performed. In particular, subtraction operation is performed with addition of variable (q_2) to avoid underflow condition. From Step 6 to 9, the tiny fast

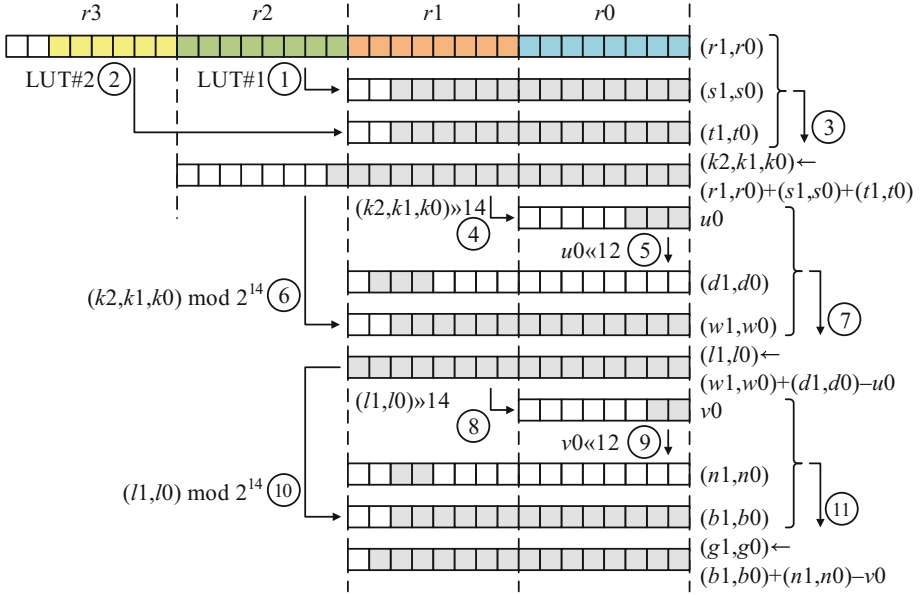


Fig. 2. Look-Up Table based fast reduction for $q = 12289$, ①②: LUT access; ③: addition; ④⑤: shifting; ⑥: modulo; ⑦: addition and subtraction; ⑧⑨: shifting; ⑩: modulo; ⑪: addition and subtraction.

reduction operation is performed. However, the result we get in Step 9 may still be larger than modulus ($q = 7681$), thus, we do the correction by subtracting the modulus (q). If the underflow condition occurs, we perform the addition with modulus (q) with the mask variable (P). Finally, the results (R) are always kept within $0x2000$ in the incomplete representation.

For the case of 12289, we can adopt the constant modular addition and subtraction techniques in Algorithm 3. Only the parameters are different. The detailed descriptions are given in Algorithm 4. Firstly, the addition and subtraction operations are performed. Afterwards, the fast reduction is performed. The obtained results (R) are always kept within $0x4000$ in the incomplete representation.

4 Performance Evaluation

This section presents performance results of our implementation. We first give the experimental platform in Sect. 4.1. Afterwards, we show a comparison with the previous modular multiplication and NTT implementations in Sect. 4.2. Finally, we show a comparison with the previous Ring-LWE implementation in Sect. 4.3.

Algorithm 3. Constant modular addition/subtraction for $q = 7681$ (0x1E01)

Require: Two 2-word operands A and B in $[0, 2^{14} - 1]$.**Ensure:** The incomplete result $R = A(+, -)B \bmod 0x2000$ $[0, 2^{14} - 1]$ or $0x1E01$.

```

1: if addition then
2:    $R \leftarrow A + B$                                 {addition operation}
3: else if subtraction then
4:    $q_2 \leftarrow q \ll 2$                             {subtraction operation}
5:    $R \leftarrow q_2 + A - B$                           {underflow prevention}
6:    $R_1 \leftarrow R \gg 13$                             {tiny fast reduction}
7:    $R_2 \leftarrow (R \gg 4) \& 0x1E00$ 
8:    $R \leftarrow R \& 0x1FFF$ 
9:    $R \leftarrow R - R_1 + R_2$ 
10: if complete then
11:    $\{Borrow, R\} \leftarrow R - 0x1E01$                 {last correction}
12:    $P \leftarrow 0x0000 - Borrow$ 
13:    $R \leftarrow R + (0x1E01 \& P)$ 
14: return  $R$ 

```

Algorithm 4. Constant modular addition/subtraction for $q = 12289$ (0x3001)

Require: Two 2-word operands A and B in $[0, 2^{15} - 1]$.**Ensure:** The incomplete or complete result $R = A(+, -)B \bmod 0x4000 \in [0, 2^{15} - 1]$ or $0x3001$.

```

1: if addition then
2:    $R \leftarrow A + B$                                 {addition operation}
3: else if subtraction then
4:    $q_2 \leftarrow q \ll 2$                             {subtraction operation}
5:    $R \leftarrow q_2 + A - B$                           {underflow prevention}
6:    $R_1 \leftarrow R \gg 14$                             {tiny fast reduction}
7:    $R_2 \leftarrow (R \gg 2) \& 0x7000$ 
8:    $R \leftarrow R \& 0x3FFF$ 
9:    $R \leftarrow R - R_1 + R_2$ 
10: if complete then
11:    $\{Borrow, R\} \leftarrow R - 0x3001$                 {last correction}
12:    $P \leftarrow 0x0000 - Borrow$ 
13:    $R \leftarrow R + (0x3001 \& P)$ 
14: return  $R$ 

```

4.1 Experimental Platform

Our implementation uses ATxmega128A1 processor on an Xplain board as target platform. This processor has a maximum frequency of 32 MHz, 128 KB flash program memory, and 8 KB SRAM. It supports an AES crypto-accelerator and can be used in a wide range of applications, such as industrial, hand-held battery applications as well as some medical devices. The implementation is written using a mixed ANSI C and Assembly languages. In particular, the main structure and interface are written in C while the core operations such as modular arithmetic is implemented in Assembly. For the LUT based approach, the con-

Table 1. Execution time of modular multiplication and NTT (in clock cycles), where 128-bit security represents ($n : 256, q : 7681$) and 256-bit security represents ($n : 512, q : 12289$) on 8-bit AVR processors, e.g., ATxmega64, ATxmega128.

Implementation	128-bit security			256-bit security		
	MOD MUL	NTT	Const	Mod MUL	NTT	Const
Boorghany and Jalili [2]	N/A	1,216,000	–	N/A	2,207,787	–
Boorghany et al. [1]	N/A	754,668	–	N/A	N/A	–
Pöppelmann et al. [16]	N/A	334,646	–	N/A	855,595	–
Liu et al. [11]	N/A	193,731	–	N/A	441,572	–
Liu et al. [7]	73	194,145	✓	70	516,971	✓
This work	57	158,607	✓	66	403,224	✓

stant LUT variables are stored in flash program memory, which requires 0.5 KB for saving the parameters and 3 clock cycles for each byte access. We compiled our implementation with speed optimization option ‘-O3’ on Atmel Studio 6.2. In order to obtain accurate timing, we execute each operation for at least 1000 times and report average cycle count for each operation.

4.2 Comparison of Modular Multiplication and NTT

Table 1 summarizes execution time of modular multiplication and NTT for both of medium-term and long-term security levels. First, various works including [1, 2, 11, 16] are not constant-time solutions, which means the attackers can perform timing attack to extract the secret information. Recent work by Liu et al. introduced the secure approach with tiny Montgomery reduction [7]. They perform the Montgomery reduction to reduce the 28/30-bit variables to 14/15-bit results. However, the complexity of n -word Montgomery reduction is generally $n^2 + n$, which is still high overheads on the low-end devices. Unlike previous approaches, we used LUT based approach to achieve high performance and secure implementation.

As shown in the Table 1, the proposed modular multiplication with 7681 and 12289 only requires 57 and 66 clock cycles, which are 16 and 4 clock cycles smaller than previous approaches, respectively [7]. The proposed NTT operation also shows higher performance than previous works. NTT operation only requires 158,607 clock cycles for 128-bit security implementation and 403,224 cycles for 256-bit security implementation. Results of NTT for medium and long-term security are 18.3% and 22.0% faster than previous works, respectively.

4.3 Comparison of Ring-LWE

With optimized NTT implementation, we evaluated the Ring-LWE encryption scheme with parameter sets (n, q, σ) with $(256, 7681, 11.31/\sqrt{2\pi})$ and $(512, 12289, 12.18/\sqrt{2\pi})$ for security levels of 128-bit and 256-bit. The tailcut

Table 2. Performance comparison of software implementation of 128-bit and 256-bit security level lattice-based cryptosystems on 8-bit AVR processors, e.g., ATxmega64, ATxmega128.

Implementation	NTT/FFT	Sampling	Enc	Secure
<i>Implementations of 128-bit security level</i>				
Boorghany and Jalili [2]	1,216,000	N/A	5,024,000	–
Boorghany et al. [1]	754,668	N/A	3,042,675	–
Pöppelmann et al. [16]	334,646	N/A	1,314,977	–
Liu et al. [11]	193,731	26,763	671,628	–
Liu et al. [7]	194,145	53,023	796,872	✓
This work	158,607	35,409	680,796	✓
<i>Implementations of 256-bit security level</i>				
Boorghany et al. [1]	2,207,787	617,600	N/A	–
Pöppelmann et al. [16]	855,595	N/A	3,279,142	–
Liu et al. [11]	441,572	255,218	2,617,459	–
Liu et al. [7]	516,971	105,153	1,975,806	✓
This work	403,224	69,062	1,754,064	✓

of discrete Gaussian sampler is limited to 12σ to achieve a high precision statistical difference from the theoretical distribution, which is less than 2^{-90} . These parameter sets were also used in most of the previous software implementations, e.g., [1–3, 7, 11].

Discrete Gaussian sampling is an integral part of Ring-LWE algorithm. However, previous implementations are not secure against timing and simple power analysis, since the Knuth-Yao sampler uses a bit/byte scanning operation in which the sample generated is related to the number of probability-bits/bytes scanned during a sampling operation and its timing provides secret information to an adversary about the value of the sample. In [19], Roy et al. suggested a random shuffling method to protect the Gaussian distributed polynomial against such attacks. The random permutation is performed after generating all samples. The random shuffle operation swaps all samples randomly, which removes any timing information from samplings. In the implementation, we adopt the previous Knuth-Yao sampler with byte-scanning [11, 19]. Afterwards, all generated samples are randomly mixed with the random numbers.

Table 2 compares software implementations of 128-bit and 256-bit security lattice-based cryptosystems on the 8-bit AVR processors. We compare the previous work [1, 2, 7, 11, 16] with ours. Proposed 128-bit security implementation requires 159K, 35K, and 681K cycles for NTT, sampling and encryption, respectively. Compared to the recent work [7], the NTT operation is significantly improved because we used compact modular multiplication routine. For the secure sampling, we adopted lightweight random shuffling technique, which shows better performance than previous works. The proposed implementations

are constant timing, which ensures a secure computation against simple power analysis and timing attacks. The similar performance enhancement is observed in 256-bit case.

5 Conclusion

This paper presents optimization techniques for efficient and secure implementation of NTT and its application Ring-LWE encryption on the low-end 8-bit AVR platform. For the secure KY sampler, we use the random shuffling technique to prevent the side channel attack. A combination of both NTT and KY sampler implementation achieved new speed records for secure 128-bit and 256-bit Ring-LWE encryption implementation on low-end 8-bit AVR platforms.

Our future works are applying the proposed techniques to the other low-end IoT devices, such as 8-bit PIC and 16-bit MSP processors. Similarly, these platforms also support very limited Arithmetic Logic Unit (ALU) and memory consumptions. Second, we will further investigate side channel attacks on the implementation of Ring-LWE. Unlike traditional RSA and ECC, only few works explored potential threats on the implementation of Ring-LWE.

References

1. Sarmadi, S.B., Boorghany, A., Jalili, R.: On constrained implementation of lattice-based cryptographic primitives and schemes on smart cards. *Cryptology ePrint Archive*, Report 2014/514 (2014). <https://eprint.iacr.org/2014/514.pdf>
2. Boorghany, A., Jalili, R.: Implementation and comparison of lattice-based identification protocols on smart cards and microcontrollers. *Cryptology ePrint Archive*, Report 2014/078 (2014)
3. De Clercq, R., Roy, S.S., Vercauteren, F., Verbauwhede, I.: Efficient software implementation of Ring-LWE encryption. In: 18th Design, Automation & Test in Europe Conference & Exhibition, DATE 2015 (2015)
4. Liu, Z., Azarderakhsh, R., Kim, H., Seo, H.: Efficient software implementation of Ring-LWE encryption on IoT processors. *IEEE Trans. Comput.* (2017)
5. Liu, Z., Huang, X., Hu, Z., Khan, M.K., Seo, H., Zhou, L.: On emerging family of elliptic curves to secure internet of things: ECC comes of age. *IEEE Trans. Dependable Secure Comput.* **14**(3), 237–248 (2017)
6. Liu, Z., Longa, P., Pereira, G., Reparaz, O., Seo, H.: FourQ on embedded devices with strong countermeasures against side-channel attacks. Technical report, *Cryptology ePrint Archive*, Report 2017/434 (2017). 28, 29
7. Liu, Z., Pöppelmann, T., Oder, T., Seo, H., Roy, S.S., Güneysu, T., Großschädl, J., Kim, H., Verbauwhede, I.: High-performance ideal lattice-based cryptography on 8-bit AVR microcontrollers. *ACM Trans. Embed. Comput. Syst. (TECS)* **16**(4), 117 (2017)
8. Liu, Z., Seo, H., Großschädl, J., Kim, H.: Efficient implementation of NIST-compliant elliptic curve cryptography for sensor nodes. In: Qing, S., Zhou, J., Liu, D. (eds.) *ICICS 2013*. LNCS, vol. 8233, pp. 302–317. Springer, Cham (2013). https://doi.org/10.1007/978-3-319-02726-5_22

9. Liu, Z., Seo, H., Großschädl, J., Kim, H.: Efficient implementation of NIST-compliant elliptic curve cryptography for 8-bit AVR-based sensor nodes. *IEEE Trans. Inf. Forensics Secur.* **11**(7), 1385–1397 (2016)
10. Liu, Z., Seo, H., Hu, Z., Hunag, X., Großschädl, J.: Efficient implementation of ECDH key exchange for MSP430-based wireless sensor networks. In: *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*, pp. 145–153. ACM (2015)
11. Liu, Z., Seo, H., Roy, S.S., Großschädl, J., Kim, H., Verbauwhede, I.: Efficient Ring-LWE encryption on 8-Bit AVR processors. In: Güneysu, T., Handschuh, H. (eds.) *CHES 2015*. LNCS, vol. 9293, pp. 663–682. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48324-4_33
12. Liu, Z., Seo, H., Xu, Q.: Performance evaluation of twisted edwards-form elliptic curve cryptography for wireless sensor nodes. *Secur. Commun. Netw.* **8**(18), 3301–3310 (2015)
13. Liu, Z., Weng, J., Hu, Z., Seo, H.: Efficient elliptic curve cryptography for embedded devices. *ACM Trans. Embed. Comput. Syst. (TECS)* **16**(2), 53 (2016)
14. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. *Cryptology ePrint Archive, Report 2012/230* (2012)
15. Oder, T., Pöppelmann, T., Güneysu, T.: Beyond ECDSA and RSA: lattice-based digital signatures on constrained devices. In: *51st Annual Design Automation Conference, DAC 2014* (2014)
16. Pöppelmann, T., Oder, T., Güneysu, T.: High-performance ideal lattice-based cryptography on 8-Bit ATxmega microcontrollers. In: Lauter, K., Rodríguez-Henríquez, F. (eds.) *LATINCRYPT 2015*. LNCS, vol. 9230, pp. 346–365. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-22174-8_19
17. Qiu, L., Liu, Z., Pereira, G.C., Seo, H.: Implementing RSA for sensor nodes in smart cities. *Pers. Ubiquit. Comput.* **21**(5), 807–813 (2017)
18. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: *37th Annual ACM Symposium on Theory of Computing*, pp. 84–93 (2005)
19. Roy, S.S., Reparaz, O., Vercauteren, F., Verbauwhede, I.: Compact and side channel secure discrete Gaussian sampling (2014)
20. Roy, S.S., Vercauteren, F., Mentens, N., Chen, D.D., Verbauwhede, I.: Compact Ring-LWE cryptoprocessor. In: Batina, L., Robshaw, M. (eds.) *CHES 2014*. LNCS, vol. 8731, pp. 371–391. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44709-3_21
21. Seo, H.: Faster (feat. ECC PMULL) over F2571. In: *A Systems Approach to Cyber Security: Proceedings of the 2nd Singapore Cyber-Security R&D Conference (SG-CRC 2017)*, vol. 15, p. 97. IOS Press (2017)
22. Seo, H., Kim, H.: MoTE-ECC based encryption on MSP430. *J. Inf. Commun. Converg. Eng.* **15**(3), 160–164 (2017)
23. Seo, H., Liu, Z., Großschädl, J., Kim, H.: Efficient arithmetic on ARM-NEON and its application for high-speed RSA implementation. *Secur. Commun. Netw.* **9**(18), 5401–5411 (2016)
24. Seo, H., Liu, Z., Nogami, Y., Park, T., Choi, J., Zhou, L., Kim, H.: Faster ECC over $\mathbb{F}_{2^{521-1}}$ (feat. NEON). In: Kwon, S., Yun, A. (eds.) *ICISC 2015*. LNCS, vol. 9558, pp. 169–181. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-30840-1_11
25. Shor, P.: Algorithms for quantum computation: discrete logarithms and factoring. In: *1994 Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, pp. 124–134, November 1994

Broadcast Encryption



Recipient Revocable Broadcast Encryption Schemes Without Random Oracles

Kamalesh Acharya^(✉) and Ratna Dutta

Department of Mathematics, Indian Institute of Technology Kharagpur,
Kharagpur 721302, India
kamaleshiitkgp@gmail.com, ratna@maths.iitkgp.ernet.in

Abstract. Public key broadcast encryption system is a fundamental cryptographic primitive that enables a broadcaster to transmit encrypted content to a set of users allowing only a privileged subset of users to decrypt the content. Traditionally, it is not possible to remove any receiver from the encrypted content without decryption. Recipient revocable broadcast encryption (RRBE) is an useful cryptographic primitive whereby a trusted third party can revoke a set of users from the encrypted content without having the ability to decrypt it. This property is not achievable in traditional broadcast encryption (BE) schemes. However, the currently existing RRBE schemes are secure only in the random oracle model. In this paper, we propose two new constructions for RRBE with *constant* number of pairing, linear exponentiation operations and analyze their security in the *standard model*. Our first construction achieves *adaptive security* in the standard model with *constant* communication cost as opposed to the existing adaptively secure RRBE schemes all of which use random oracles and have linear communication cost. The storage and computation complexity are linear to the total number of users and the number of subscribed users respectively.

Our second construction attains *selective security* in the standard model with *constant* size public parameter and secret key. The communication and computation overhead are linear to the number of revoked users. We emphasize that, this scheme is *flexible* in a sense that constant size public parameter allows to encrypt any number of users in the system.

The proposed constructions are highly comparable with the existing similar schemes, exhibits better performance over them and practically more efficient.

Keywords: Recipient revocable broadcast encryption
Chosen plaintext attack · Adaptive security

1 Introduction

With the rapid development of technology, the application of the Internet already extended to each aspect of people's life, and has gradually become a necessary

part of our lives. Availability of internet makes a threat in copyright protection and abuse of intellectual properties. Broadcast encryption (BE) provides skilful techniques to handle the issue. BE introduced by Fiat and Naor [11], supports a broadcaster to send an encrypted content to a group of users in such a way that only the subscribed users can recover the message. Due to its numerous application in real life expanding from pay TV to digital rights management, many BE schemes have been developed. However, traditional BE schemes do not support revocation of recipients from the encrypted content. Recipient revocable broadcast encryption (RRBE) is a new broadcast encryption (e.g., [1-4, 7-9, 11]) primitive, introduced by Susilo et al. [17], skillfully revokes users from the encrypted content without decrypting it. In RRBE, a content provider and a broadcaster agree upon a common set of recipients. The broadcaster broadcasts a ciphertext by modifying the original encrypted content received from the content provider in such a way that enables the broadcaster to revoke the intended subscribers without having the ability to decrypt the original encrypted content. A subscribed user can decrypt this modified content using its secret key and can recover the message.

Applications: Consider the following applications where RRBE is useful.

- Suppose in an academic institute, the head wants to send an important message (or file) to the current students. He makes an encrypted message for all the enrolled students and securely sends to an academic staff. Some students may leave the institute as they get jobs or for other reasons. The academic staff makes a list of current students and wants to revoke the students who are currently not present in the institute without having eligibility of recovering the message.
- To increase the business, let Internet service provider, such as CenturyLink, collaborates with movie content provider CinemaNow and provides free access of CinemaNow to the subscribers. CinemaNow sends the content securely (else any body can act as a broadcaster) to CenturyLink. CenturyLink wants to distribute the content to as many users as possible to increase its profit. Therefore, CinemaNow cannot send the content in a plaintext form and sends in encrypted form. CenturyLink wants to revoke a user who has not paid previous month subscription charge. Now the challenge for CenturyLink comes to revoke users using the encrypted content provided by CinemaNow.
- A study centre provides online material to the students. It releases 12 month material on monthly basis. It provides first month content on the monthly agreement and if the students are satisfied then they need to pay for remaining 11 months contract amount to get the remaining materials. The teacher provides (to study centre) all the content for the students who have registered in the first month. He does not want to send the material in plaintext form as there is the opportunity for misuse of study materials by the study centre. As some students do not continue after the first month, study centre needs to revoke those students.

The key challenge in designing an RRBE comes from the difficulty of revoking users by the broadcaster from the encrypted content without recovering the

content. One may think that RRBE can be developed by “decrypt then re-encrypt” technique. But in this case, the broadcaster can recover the message and can rebroadcast the content to other users, thereby damages the copyright issue or business opportunity of the content provider.

The performance of an RRBE scheme is measured by the storage size, communication and computation overhead. The computation cost includes the encryption, revocation and decryption complexity ignoring the computations that are offline and performed one time only. The communication cost is determined by the number of bits in the ciphertext. Security is another issue. Security of an RRBE can be achieved in the selective, semi-static or adaptive security model. In the selective security, the adversary publishes a challenge set in the initialization phase at the beginning of the security game. The challenger can generate the public parameter, secret key and challenge ciphertext according to the initially committed challenge set. In the semi-static model, the adversary sets an initial set in the initialization phase and declares a subset of the initial set as the challenge set in the challenge phase. The hardness of achieving semi-static security is in between of the selective and adaptive security. The adaptive security is the strongest security model where the challenge set is fixed in the challenge phase only and the challenger needs to set the public parameter, secret key without utilising the challenge set.

Related works: Susilo et al. [17] introduced RRBE in AsiaCCS 2016 using the identity-based broadcast encryption scheme of Delerablée [10]. The scheme is selectively secure in the random oracle model under the (\hat{f}, ϕ, F) -General Decisional Diffie-Hellman Exponent $((\hat{f}, \phi, F)$ -GDDHE) assumption. The scheme has constant ciphertext and secret key size. The public parameter size is linear to the maximum number of users supported by the system. The encrypted content size is linear to the maximum possible number of revoked users. Lai et al. [13] proposed an adaptively secure scheme with constant storage in 2016 using Lagrange’s interpolation polynomial. It needs $O(n^2)$ exponentiations and $O(n)$ pairings for encryption and the ciphertext size is $O(n)$, where n is the number of users selected by the content provider to prepare the encrypted content. Moreover, the adaptive security under the Bilinear Diffie-Hellman Exponent (BDHE) problem is achieved in the random oracle model. Recently, Lai et al. [12] propose another scheme employing an authorisation set which has the same storage size, encryption cost and security property. It needs $O(m^2)$ exponentiations in revocation phase, m being the number of elements in the authorized set where m is greater than the size of the subscribed user set after revocation. Scheme [13] achieves anonymity in terms of ciphertext privacy. Very recently Lai et al. [14] proposed another scheme in selective security model extending [13], it achieves both ciphertext and user privacy as [12] at the cost of pairing computations and communication bandwidth of [13].

Our Contribution: A security proof in the random oracle model can be treated as a heuristic argument as all parties get black box access to a truly random function. On the other hand, there is no assumption for the existence of such random looking function in the standard model and security of the scheme relies on the

Table 1. Comparison of storage, communication bandwidth, security.

Scheme	PP	SK	CT'	CT	SM	RO	SA
[17]	$(2N+1) \mathbb{G} + 1 \mathbb{G}_1 $	$(1) \mathbb{G} $	$(k+2) \mathbb{G} + 1 \mathbb{G}_1 $	$2 \mathbb{G} + 1 \mathbb{G}_1 $	Sel	Yes	(\hat{f}, g, F) -GDDHE
[13]	$(2) \mathbb{G} $	$(1) \mathbb{G} $	$(2n+3) \mathbb{G} $	$(2n+3) \mathbb{G} $	Adp	Yes	BDHE
[12]	$(2) \mathbb{G} $	$(1) \mathbb{G} $	$(n+3) \mathbb{G} $	$(m+2) \mathbb{G} $	Adp	Yes	BDHE
[14]	$(2) \mathbb{G} $	$(1) \mathbb{G} $	$(2n+6) \mathbb{G} + 1 \mathbb{Z}_p $	$(2n+3) \mathbb{G} + (l-1) \mathbb{Z}_p $	Sel	Yes	BDHE
RRBE-I	$(3N+3) \mathbb{G} + 2 \mathbb{G}_1 $	$(N+3) \mathbb{G} + 1 \mathbb{Z}_p $	$(k+2) \mathbb{G} + 2 \mathbb{G}_1 $	$2 \mathbb{G} + 2 \mathbb{G}_1 $	Adp	No	q -wDABDHE
RRBE-II	$(4) \mathbb{G} + 1 \mathbb{G}_1 $	$3 \mathbb{G} $	$(2r_1 + 1) \mathbb{G} + 1 \mathbb{G}_1 $	$(2r+1) \mathbb{G} + 1 \mathbb{G}_1 $	Sel	No	q -DMEBDH

$|CT'|$ = encrypted content size, $|CT|$ = ciphertext size, N = total number of users, k = maximum number of revoked users fixed in encryption phase, m = size of authorisation set, r_1 = number of revoked users in Encrypt phase of RRBE-II, r_2 = number of newly revoked users in Revoke phase of RRBE-II, $r = r_1 + r_2$, $|\mathbb{G}|$ = bit size of an element of \mathbb{G} , SM = security model, SA = security assumption, RO = random oracle, Sel = Selective, Adp = Adaptive.

Table 2. Comparison of computation cost.

Scheme	PP		SK		Enc			Revoke			Dec		
	#exp	#p	#exp	#in	#exp	#p	# in	#exp	#p	# inv	#exp	#p	# in
[17]	$2N$ in \mathbb{G}	1	1 in \mathbb{G}	0	$n+k+2$ in \mathbb{G} , 1 in \mathbb{G}_1	0	0	$2l+1$ in \mathbb{G}	1	0	$n'-1$ in \mathbb{G} , 1 in \mathbb{G}_1	2	1 in \mathbb{G}_1
[13]	1 in \mathbb{G}	0	1 in \mathbb{G}	0	$2n^2+2$ in \mathbb{G} , $2n$ in \mathbb{G}_1 ,		0	l in \mathbb{G}	1	0	$2n-2$ in \mathbb{G}	2	3 in \mathbb{G}
[12]	1 in \mathbb{G}	0	1 in \mathbb{G}	0	n^2+3n+2 in \mathbb{G} , $n+1$ in \mathbb{G}_1	$n+1$	0	$nm+m^2$ in \mathbb{G}	0	0	$m-1$ in \mathbb{G}	1	2 in \mathbb{G}
[14]	1 in \mathbb{G}	0	1 in \mathbb{G}	0	$2n^2+3$ in \mathbb{G} , $3n+1$ in \mathbb{G}_1 ,	$4n+2$	0	l in \mathbb{G}	$l+1$	0	$2n-2$ in \mathbb{G}	3	3 in \mathbb{G}
RRBE-I	$3N+1$ in \mathbb{G}	2	$N+4$ in \mathbb{G}	0	$n+k+2$ in \mathbb{G} , 2 in \mathbb{G}_1	0	1 in \mathbb{G} , 1 in \mathbb{G}_1	$2l+1$ in \mathbb{G}	2	3 in \mathbb{G}	$4n'-2$ in \mathbb{G} , 4 in \mathbb{G}_1	4	1 in \mathbb{G}_1
RRBE-II	3 in \mathbb{G} , 1 in \mathbb{G}_1	1	5 in \mathbb{G} , 1 in \mathbb{G}_1	1 in \mathbb{G}	$3r_1+1$ in \mathbb{G} , 1 in \mathbb{G}_1	0	0	$3r_2+1$ in \mathbb{G} , 1 in \mathbb{G}_1	1	0	$2r$ in \mathbb{G}	3	2 in \mathbb{G}_1

l = actual number of revoked users in revocation phase, n = number of users used in encryption phase, $n' = n - l$, #exp = number of exponentiations in \mathbb{G} and \mathbb{G}_1 , #p = number of pairings, #in = number of inversions.

hardness of mathematical problem. This paper addresses the challenging task to design efficient RRBE in the *standard model* and introduces two such RRBE constructions using *constant* number of pairing and linear exponentiation computations, namely, RRBE-I and RRBE-II. More precisely, the proposed schemes provide the followings features.

- Our RRBE-I uses the identity-based broadcast encryption scheme of Ren et al. [16] and achieves *adaptive security* in the *standard model* under the q -weaker Decisional Augmented Bilinear Diffie-Hellman Exponent (q -wDABDHE) assumption. More positively, this scheme achieves *constant* ciphertext size while storage and encrypted content size are linear to the number of users supported by the system and maximum revocation number respectively.

- Our second construction RRBE-II is the first RRBE where the encryption is performed using the set of revoked user identities. It follows the identity-based encryption scheme of Lewko et al. [15]. RRBE-II is more practical in following scenario. Suppose a broadcast system involves a set \mathcal{U} of N users and the content provider needs to generate encrypted content by involving all but a smaller set $R \subset \mathcal{U}$ of r users where $r \ll N$. The encryption phase in [12–14, 17] and our RRBE-I involves the set $\mathcal{U} \setminus R$ which is large compared to that in RRBE-II which involves a smaller set R . Consequently, RRBE-II outperforms the existing RRBE schemes where $r \ll N$. More interestingly, the public parameter is of constant size and *independent* of user set in RRBE-II, therefore is *flexible* in the sense that the system can accommodate any number of users without altering the existing public parameter provided the chosen prime number (bilinear group order) is greater than total number of users. Furthermore, the scheme achieves *selective semantic security* under the hardness of the q -Decisional Multi-Exponent Bilinear Diffie-Hellman (q -DMEBDH) problem.

Additionally, in both the proposed schemes, new users can join any time without any updation of the pre-existing PP, SK provided the number of subscribed users in the system does not exceed the maximum number of users allowed in the system. More interestingly, the broadcaster has the control to revoke any user of the group for which original content was created by the content provider without changing the existing setup. Besides, the schemes are non-interactive in the sense that the private key generation centre does not need to interact with the subscribed users after issuing users' secret key. We have compare our constructions with existing similar schemes in Tables 1 and 2.

2 Preliminaries

Notation: Let $[m]$, $[a, b]$ denote integers from 1 to m , a to b respectively. We use the notation $x \in_R S$ to denote x is a random element of S and λ to represent bit size of prime integer p . Let $\epsilon : \mathbb{N} \rightarrow \mathbb{R}$ be a function, where \mathbb{N} and \mathbb{R} are the sets of natural and real numbers respectively. The function $\epsilon(\lambda)$ is said to be a *negligible function* if for every positive integer c , \exists an integer N_c such that for every $\lambda > N_c$, $\epsilon(\lambda) \leq \frac{1}{\lambda^c}$. Let $|G|$ denotes the number of elements of group G .

2.1 Recipient Revocable Broadcast Encryption (RRBE)

RRBE introduced by Susilo et al. [17] in 2016, enables a trusted third party to generate public parameter and secret key, a content provider to provide encrypted content to a broadcaster, and the broadcaster to revoke certain users from the system without having the ability to decrypt the encrypted content issued by the content provider. Formal description of RRBE in identity based setting is provided below:

Syntax of RRBE:

A RRBE=(Setup,KeyGen,Encrypt,Revoke,Decrypt) consists of the followings:

- $(PP, MK) \leftarrow \text{Setup}(N, \lambda)$: A trusted authority, called private key generation centre (PKGc) takes as input the total number of users N supported by the system and a security parameter λ and creates the public parameter PP and a master key MK . The public parameter PP includes the message space \mathcal{M} and publicly available while the master key MK is kept secret by PKGc.
- $(sk_i) \leftarrow \text{KeyGen}(PP, MK, ID_i)$: The PKGc receives as input the public parameter PP , master key MK and user identity ID_i from user i and returns a secret key sk_i associated with identity ID_i to user i through a secure communication channel between the PKGc and user i .
- $(CT', S) \leftarrow \text{Encrypt}(S, PP, M)$: The broadcaster and the content provider agree upon a set S of n ($\leq N$) user identities. The content provider takes as input public parameter PP , S , message $M \in \mathcal{M}$ and produces as output an encrypted content CT' associated with the set S . The content provider sends CT' together with the description of set S securely to the broadcaster. Note that using the size of CT' it is easy to understand the maximum revocation number k . To allow maximum revocation capability to the broadcaster, content provider should set $k = n - 1$.
- $(CT, G) \leftarrow \text{Revoke}(PP, CT', S, R)$: On input PP , S , encrypted content CT' for the set S , and a set R of l ($\leq k$) identities of users to be revoked, the broadcaster broadcast CT and makes $G = S \setminus R$ public.
- $(M) \leftarrow \text{Decrypt}(PP, sk_i, CT, G)$: A subscribed user i with identity ID_i , outputs the message M using PP , CT , sk_i and the subscribed identity set G .

Correctness: We say that a RRBE is correct if for all λ, M

$$Pr \left[\begin{array}{l} \text{Decrypt}(PP, sk_i, CT, G) = M : (PP, MK) \leftarrow \text{Setup}(N, \lambda) \\ (sk_i) \leftarrow \text{KeyGen}(PP, MK, ID_i) \\ (CT', S) \leftarrow \text{Encrypt}(S, PP, M) \\ (CT, G) \leftarrow \text{Revoke}(PP, CT', S, R) \end{array} \right] = 1$$

2.2 Security Framework**Message Indistinguishability of RRBE Under CPA:**

Following Susilo et al. [17], we define the selective security of the scheme RRBE as a message indistinguishability game played between a challenger \mathcal{C} and an adversary \mathcal{A} as follows:

Initialization: The adversary \mathcal{A} selects a recipient set (i.e., the set of subscribed users) G and provides it to \mathcal{C} .

Setup: The challenger \mathcal{C} generates (PP, MK) by executing $\text{Setup}(N, \lambda)$, provides PP to \mathcal{A} , and keeps MK secret to itself.

Phase 1: The adversary \mathcal{A} is allowed to make key generation queries for user identities $ID_{i_1}, \dots, ID_{i_m}$, where $ID_i \notin G$, $i \in \{i_1, \dots, i_m\}$. On input of user

identity ID_i , $i \in \{i_1, \dots, i_m\}$, \mathcal{C} obtains sk_i by executing $\text{KeyGen}(\text{PP}, \text{MK}, ID_i)$ and sends to \mathcal{A} .

Challenge: At this stage, the adversary \mathcal{A} submits to \mathcal{C} two equal length plaintexts $M_0, M_1 \in \mathcal{M}$, a set R of user identities to be revoked with no identity of R lies in G . The challenger \mathcal{C} picks a bit $\mu \in_R \{0, 1\}$ and generates CT' , CT by executing $\text{Encrypt}(S, \text{PP}, M_\mu)$, $\text{Revoke}(\text{PP}, \text{CT}', S, R)$ respectively. The adversary \mathcal{A} gets a challenge ciphertext CT^* as

$$\text{CT}^* = \begin{cases} \text{CT}', & \text{if } R = \phi \\ \text{CT}, & \text{if } R \neq \phi. \end{cases}$$

Phase 2: The adversary \mathcal{A} can issue additional key generation queries as in Phase 1 with a restriction that queried user identities does not lie in G .

Guess: The adversary \mathcal{A} outputs a guess $\mu' \in \{0, 1\}$ of μ and wins if $\mu' = \mu$. The adversary \mathcal{A} 's advantage in the above security game is defined as $\text{Adv}_{\mathcal{A}, \text{RRBE}}^{\text{IND-sCPA}} = |\text{Pr}(\mu' = \mu) - \frac{1}{2}|$. The probability is taken over random bits used by \mathcal{C} and \mathcal{A} . Let the adversary \mathcal{A} is allowed to get reply up to t key generation queries.

Definition 1. *The broadcast encryption scheme RRBE is said to be (T, t, ϵ) -secure if for every probabilistic polynomial time (PPT) adversary \mathcal{A} with running time T and making at most t key generation queries $\text{Adv}_{\mathcal{A}, \text{RRBE}}^{\text{IND-sCPA}} = \epsilon$.*

In adaptive security, there is no Initialization phase and Phase 1 has no restrictions on key generation query.

2.3 Complexity Assumptions

Definition 2. (Bilinear Map). *Let \mathbb{G} and \mathbb{G}_1 be two multiplicative groups of prime order p . Let g be a generator of \mathbb{G} . A function $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_1$ is said to be bilinear mapping if it has the following properties:*

1. $e(u^a, v^b) = e(u, v)^{ab}$, $\forall u, v \in \mathbb{G}$ and $\forall a, b \in \mathbb{Z}_p$.
2. The function is non-degenerate, i.e., $e(g, g)$ is a generator of \mathbb{G}_1 .
3. e is efficiently computable.

The tuple $\mathbb{S} = (p, \mathbb{G}, \mathbb{G}_1, e)$ is called a prime order bilinear group system.

- **The q -Decisional Multi-exponent Bilinear Diffie-Hellman (q -DMEBDH) Assumption [15]:**

Input: $\langle Z, K \rangle$, where

$$Z = \left\{ \begin{array}{l} \mathbb{S}, g, g^s, e(g, g)^\alpha, \\ g^{a_i}, g^{a_i s}, g^{a_i a_j}, g^{\alpha/a_i^2}, \quad \forall 1 \leq i, j \leq q \\ g^{a_i a_j s}, g^{\alpha a_j/a_i^2}, g^{\alpha a_i a_j/a_k^2}, g^{\alpha a_i^2/a_j^2}, \quad \forall 1 \leq i, j, k \leq q, i \neq j \end{array} \right\}$$

for some $s, \alpha, a_1, \dots, a_q \in \mathbb{Z}_p$, and K is either $e(g, g)^{\alpha s}$ or $X \in_R \mathbb{G}_1$.

Output: 0 if $K = e(g, g)^{\alpha s}$; 1 otherwise.

Definition 3. *The q -DMEBDH assumption holds with (T, ϵ) if for every PPT adversary \mathcal{A} with running time at most T , the advantage of solving the above problem is at most ϵ , i.e.,*

$$Adv_{\mathcal{A}}^{q\text{-DMEBDH}} = |Pr[\mathcal{A}(Z, K = e(g, g)^{\alpha s}) = 0] - Pr[\mathcal{A}(Z, K = X) = 0]| \leq \epsilon.$$

q -DMEBDH assumption: $Adv_{\mathcal{A}}^{q\text{-DMEBDH}}$ is negligible.

- **The l -weaker Decisional Augmented Bilinear Diffie-Hellman Exponent (l -wDABDHE) Assumption [16]:**

Input: $\langle Z = (\mathbb{S}, h, h^{\alpha^{l+2}}, \dots, h^{\alpha^{2l}}, g, g^{\alpha}, \dots, g^{\alpha^l}), K \rangle$, where g is a generator of \mathbb{G} , $h \in_R \mathbb{G}$, $\alpha \in_R \mathbb{Z}_p$, K is either $e(g, h)^{\alpha^{l+1}}$ or a random element $X \in \mathbb{G}_1$.
Output: 0 if $K = e(g, h)^{\alpha^{l+1}}$; 1 otherwise.

Definition 4. *The l -wDABDHE assumption holds with (T, ϵ) if for every PPT adversary \mathcal{A} with running time at most T , the advantage of solving the above problem is at most ϵ , i.e.,*

$$Adv_{\mathcal{A}}^{l\text{-wDABDHE}} = |Pr[\mathcal{A}(Z, K = e(g, h)^{\alpha^{l+1}}) = 0] - Pr[\mathcal{A}(Z, K = X) = 0]| \leq \epsilon.$$

l -wDABDHE assumption: $Adv_{\mathcal{A}}^{l\text{-wDABDHE}}$ is negligible.

The hardness of the q -DMEBDH and l -wDABDHE assumptions follow from the General Decisional Diffie-Hellman Exponent (GDDHE) problem [6] introduced by Boneh et al. [6].

3 Our RRBE-I

3.1 Protocol Description

Our RRBE-I=(Setup,KeyGen,Encrypt,Revoke,Decrypt) involves a PKGC, a content provider, a broadcaster and several users. The PKGC runs Setup to generate public parameter PP and master key MK and runs KeyGen to generate sk_i of user i with identity ID_i . A content provider and a broadcaster agree upon a common set of users. The content provider invokes Encrypt to encrypt a content for a set S using PP and securely sends the encrypted content to the broadcaster. From this encrypted content, the broadcaster produces CT running Revoke by revoking a set of users $R \subset S$ of its choice without getting any information about the original content. Legal users use their secret keys to recover the content. The algorithms are detailed below.

- $(PP, MK) \leftarrow \text{Setup}(N, \lambda)$: Given the security parameter λ , the PKGC generates the public parameter PP and a master key MK as follows:
 - Chooses a prime order bilinear group system $\mathbb{S} = (p, \mathbb{G}, \mathbb{G}_1, e)$, where \mathbb{G}, \mathbb{G}_1 are groups of prime order ($p > N$) and $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_1$ is a bilinear mapping.

– Selects $\alpha, \beta \in_R \mathbb{Z}_p$, and sets MK, PP as $\text{MK} = (\alpha, \beta)$,

$$\text{PP} = (\mathbb{S}, l_0, l_0^\alpha, \dots, l_0^{\alpha^N}, g, g^\alpha, \dots, g^{\alpha^N}, g^{\alpha\beta}, \dots, g^{\alpha^{N+1}\beta}, e(g, g), e(g, l_0)),$$

where g is a generator of \mathbb{G} and l_0 is random non-identity element of \mathbb{G} .

– Keeps MK secret to itself and makes PP public.

- $(sk_i) \leftarrow \text{KeyGen}(\text{PP}, \text{MK}, \text{ID}_i)$: On receiving user identity $\text{ID}_i \in \mathbb{Z}_p$ from user i , the PKGC selects $h_i \in_R \mathbb{G}$, $r_i \in_R \mathbb{Z}_p$ and generates a secret key sk_i , using $\text{MK} = (\alpha, \beta)$ and extracting g, l_0 from PP as $sk_i = (d_{1,i}, d_{2,i}, d_{3,i}, \text{label}_i)$

$$d_{1,i} = (h_i g^{r_i})^{\frac{1}{\alpha\beta(\alpha+\text{ID}_i)}}, d_{2,i} = r_i, d_{3,i} = (h_i l_0^{r_i})^{\frac{1}{\alpha\beta}}, \text{label}_i = (h_i, h_i^\alpha, \dots, h_i^{\alpha^N}).$$

It sends sk_i to user i through a secure communication channel between them. Note that identities are taken from \mathbb{Z}_p . In practice, identity may be arbitrary string but it is possible to map any string to an element of \mathbb{Z}_p using a collision resistance hash function.

- $(\text{CT}', S) \leftarrow \text{Encrypt}(S, \text{PP}, M)$: The content provider and the broadcaster agree upon a set S of n ($\leq N$) user identities. The content provider selects maximum revocation number k ($< n$) and performs the following to produce an encrypted content for S , using PP, message (or content) $M \in \mathbb{G}_1 (= \mathcal{M})$:

– Sets a polynomial $F(x)$ as $F(x) = \prod_{\text{ID}_j \in S} (x + \text{ID}_j) = \sum_{i=0}^n F_i x^i$, where F_i 's

are function of $\text{ID}_j \in S$.

– Picks $\gamma \in_R \mathbb{Z}_p$ and generates the encrypted content $\text{CT}' = (c_1, c_2, \hat{c}_1, \dots, \hat{c}_{k+1}, c_M)$ by setting

$$c_1 = \prod_{i=0}^n (g^{\alpha^{i+1}\beta})^{\gamma F_i} = g^{\sum_{i=0}^n \beta \alpha^{i+1} \gamma F_i} = g^{\alpha \beta F(\alpha) \gamma}, c_2 = e(g, g)^{-\gamma},$$

$$\hat{c}_i = (g^\alpha)^{-\gamma} = g^{-\gamma \alpha}, \hat{c}_i = (g^{\alpha^i})^\gamma = g^{\alpha^i \gamma} \text{ for } 2 \leq i \leq k+1, c_M = M \cdot e(g, l_0)^\gamma.$$

– Sends CT' together with the information of S to the broadcaster through a secure communication channel between the broadcaster and the content provider.

- $(\text{CT}, G) \leftarrow \text{Revoke}(\text{PP}, \text{CT}', S, R)$: Using PP, S , $\text{CT}' = (c_1, c_2, \hat{c}_1, \dots, \hat{c}_{k+1}, c_M)$, revocation identity set $R = \{\text{ID}_{i_1}, \text{ID}_{i_2}, \dots, \text{ID}_{i_l}\} \subset S$, ($l \leq k$) the broadcaster generates ciphertext for the set $G = S \setminus R$ as:

1. If $R = \phi$, sets $C_1 = c_1, C_2 = c_2, \hat{C}_1 = \hat{c}_1, C_M = c_M$.

2. If $R \neq \phi$, then

– Computes $\frac{\prod_{\text{ID}_j \in R} (x + \text{ID}_j)}{\prod_{\text{ID}_j \in R} \text{ID}_j} = \sum_{i=0}^l f_i x^i$, where $f_i, 0 \leq i \leq l$ are function of $\text{ID}_j \in R$.

Note that $f_0 = 1$. Also computes $X = \prod_{i=2}^l \hat{c}_i^{f_i} = g^{\gamma \sum_{i=2}^l f_i \alpha^i}$.

– Implicitly sets $s = \gamma \sum_{i=0}^l f_i \alpha^i$ and generates $C_1, C_2, \widehat{C}_1, C_M$ as

$$C_1 = c_1 \left\{ \frac{1}{\prod_{\text{ID}_j \in R} \text{ID}_j} \right\} = g^{\gamma \alpha \beta \left\{ \frac{\prod_{\text{ID}_j \in R} (\alpha + \text{ID}_j) \prod_{\text{ID}_j \in G} (\alpha + \text{ID}_j)}{\prod_{\text{ID}_j \in R} \text{ID}_j} \right\}} = g^{\alpha \beta \left\{ s \prod_{\text{ID}_j \in G} (\alpha + \text{ID}_j) \right\}},$$

$$C_2 = c_2 e(g, \widehat{c}_1^{f_1} X^{-1}) = e(g, g)^{-\gamma} e(g, g)^{-\gamma(\alpha f_1 + \alpha^2 f_2 + \dots + \alpha^l f_l)} = e(g, g)^{-s},$$

$$\widehat{C}_1 = \widehat{c}_1 \left(\prod_{i=2}^{l+1} \widehat{c}_i^{f_i - 1} \right)^{-1} = (g^{-\alpha \gamma}) (g^{-\alpha^2 \gamma})^{f_1} (g^{-\alpha^3 \gamma})^{f_2} \dots (g^{-\alpha^{l+1} \gamma})^{f_l} = g^{-\alpha s},$$

$$C_M = c_M e(\widehat{c}_1^{\{-f_1\}} X, l_0) = M \cdot e(g, l_0)^\gamma e(g, l_0)^{\gamma(\alpha f_1 + \dots + \alpha^l f_l)} = M \cdot e(g, l_0)^s.$$

In this computation, g, l_0 are extracted from PP.

3. Finally, publishes CT = $(C_1, C_2, \widehat{C}_1, C_M)$ together with the set $G = S \setminus R$.

- $(M) \leftarrow \text{Decrypt}(\text{PP}, sk_i, \text{CT}, G)$: A subscribed user i with identity ID_i , having the secret key $sk_i = (d_{1,i}, d_{2,i}, d_{3,i}, \text{label}_i = (h_i, h_i^\alpha, \dots, h_i^{\alpha^N}))$ uses PP, G , the ciphertext CT = $(C_1, C_2, \widehat{C}_1, C_M)$ and recovers the message M as follows:
 - Computes $e(g, h_i g^{r_i})^s, e(g, h_i)^s$ as

$$e(g, h_i g^{r_i})^s = \begin{cases} \left[e(C_1, d_{1,i}) e(\widehat{C}_1, (h_i g^{d_{2,i}})^{A_{i,G,\alpha}}) \right] \left\{ \frac{1}{\prod_{\text{ID}_j \in G, j \neq i} \text{ID}_j} \right\} & \text{if } |G| > 1 \\ e(C_1, d_{1,i}) & \text{if } |G| = 1 \end{cases}$$

$$e(g, h_i)^s = e(g, h_i g^{r_i})^s C_2^{d_{2,i}},$$

$$e(g, h_i l_0^{r_i})^s = \left[e(C_1, d_{3,i}) e(\widehat{C}_1, (h_i l_0^{d_{2,i}})^{B_{G,\alpha}}) \right] \left\{ \frac{1}{\prod_{\text{ID}_j \in G} \text{ID}_j} \right\}, \text{ where}$$

$$A_{i,G,\alpha} = \frac{1}{\alpha} \left\{ \prod_{\substack{\text{ID}_j \in G, \\ j \neq i}} (\alpha + \text{ID}_j) - \prod_{\substack{\text{ID}_j \in G, \\ j \neq i}} \text{ID}_j \right\}, B_{G,\alpha} = \frac{1}{\alpha} \left\{ \prod_{\text{ID}_j \in G} (\alpha + \text{ID}_j) - \prod_{\text{ID}_j \in G} \text{ID}_j \right\}.$$

[Computation of $(h_i g^{d_{2,i}})^{A_{i,G,\alpha}}$ and $(h_i l_0^{d_{2,i}})^{B_{G,\alpha}}$ have been explained in Note 1.]

- Obtain $K = e(g, l_0)^s$ as $\left\{ \frac{e(g, h_i l_0^{r_i})^s}{e(g, h_i)^s} \right\}^{\frac{1}{d_{2,i}}}$.
- Recovers the message M by computing $\frac{C_M}{K}$.

Note 1. The polynomial $\frac{1}{x} \left\{ \prod_{\text{ID}_j \in G, j \neq i} (x + \text{ID}_j) - \prod_{\text{ID}_j \in G, j \neq i} (\text{ID}_j) \right\} = \sum_{k=0}^{n-l-2} a_k x^k$

(say) is of degree $(n - l - 2)$ in x , when $|G| = n - l > 1$. With the knowledge of G , the user i can compute the co-efficients $a_k, k \in [0, n - l - 2]$ which are functions of $\text{ID}_j \in G, j \neq i$. Also g^{α^j} for $j \in [0, n - l - 2]$ are available in the public parameter PP and $h_i^{\alpha^j}$ for $j \in [0, n - l - 2]$ are extractable from label_i in sk_i . Consequently, user i having identity $\text{ID}_i \in G$ and its secret key sk_i

can compute $\prod_{j=0}^{n-l-2} (h_i^{\alpha^j})^{a_j} \prod_{j=0}^{n-l-2} (g^{\alpha^j})^{d_{2,i} a_j} = h_i^{\left\{ \sum_{j=0}^{n-l-2} a_j \alpha^j \right\}} g^{d_{2,i} \left\{ \sum_{j=0}^{n-l-2} a_j \alpha^j \right\}} = (h_i g^{d_{2,i}})^{A_{i,G,\alpha}}$ without the knowledge of α . Here $d_{2,i} = r$ is obtained from sk_i .

Similarly $(h_i l_0^{d_{2,i}})^{B_{G,\alpha}}$ is also computable. Thus a subscribed user can compute $(h_i g^{d_{2,i}})^{A_{i,G,\alpha}}$ and $(h_i l_0^{d_{2,i}})^{B_{G,\alpha}}$ (in $|G| > 1$ case).

We have given the correctness of RRBE-I in Appendix A.

Remark 1. Suppose that the content provider has generated a content for a group S of n user identities and provided it to the broadcaster as

$$CT' = (c_1, c_2, \hat{c}_1, \dots, \hat{c}_{k+1}, c_M) \text{ where } c_1 = \prod_{i=0}^n (g^{\alpha^{i+1}\beta})^{\gamma F_i} = g^{\sum_{i=0}^n \beta \alpha^{i+1} \gamma F_i} = g^{\alpha \beta F(\alpha) \gamma}, c_2 = e(g, g)^{-\gamma}, \hat{c}_1 = (g^\alpha)^{-\gamma} = g^{-\gamma \alpha}, \hat{c}_i = (g^{\alpha^i})^\gamma = g^{\alpha^i \gamma} \text{ for } 2 \leq i \leq k+1, c_M = M \cdot e(g, l_0)^\gamma.$$

Here $F(x) = \prod_{ID_j \in S} (x + ID_j) = \sum_{i=0}^n F_i x^i$, where F_i 's are function of $ID_j \in S$, $\gamma \in_R \mathbb{Z}_p$, $g^\alpha, \dots, g^{\alpha^k}, g^{\alpha^{k+1}}, g^{\alpha\beta}, \dots, g^{\alpha^n\beta}, g^{\alpha^{n+1}\beta}, e(g, g), e(g, l_0)$ are extracted from PP. It involves n users. In order to add more users, the broadcaster needs to create the ciphertext component c_1 as $c_1 = \prod_{i=0}^{n'} (g^{\alpha^{i+1}\beta})^{\gamma F_i}$, ($n' > n$) for which the explicit knowledge of γ is required. However, γ is chosen by the content provider. Consequently, the broadcaster will not be able to modify the encrypted content to involve more users.

Remark 2. The ciphertext can be modified by an attacker. To prevent this we need to use an unforgeable signature scheme which will sign on ciphertext and will also make verification key public. Each subscribed user will verify the signature and if the verification succeeds then it will decrypt the content else output \perp , indicating that the ciphertext has been modified.

Remark 3. In CT, C_1 is generated using the set G and any one can publicly verify this by checking the following- $e(C_1, g^\alpha) = e(g^{\alpha\beta \prod_{ID_j \in G} (\alpha + ID_j)}, \hat{C}_1^{(-1)})$. Note that $g^{\alpha\beta \prod_{ID_j \in G} (\alpha + ID_j)}$ is publicly computable (similar to c_1) using set G .

Remark 4. If a content provider generates a content having $k+4$ components supporting at most k revocation, the broadcaster will not be able to revoke more than k user as in this case in Revoke phase, computation of X will need more components.

3.2 Security

Theorem 1. *Our proposed scheme RRBE-I described in Sect. 3.1 achieves adaptive semantic (indistinguishability against CPA) security as per the message indistinguishability security game of Sect. 2.2 under the q -wDABDHE ($q \geq 2N$) assumption.*

We have given the proof of Theorem 1 in Appendix A.

4 Our RRBE-II

4.1 Protocol Description

Our scheme RRBE-II works as follows:

- $(PP, MK) \leftarrow \text{Setup}(N, \lambda)$: Given the security parameter λ and a set of users \mathcal{U} of size N , the PKGC generates the public parameter PP, master key MK as follows:
 - Chooses a prime order bilinear group system $\mathbb{S} = (p, \mathbb{G}, \mathbb{G}_1, e)$, where \mathbb{G}, \mathbb{G}_1 are groups of prime order p ($> N$) and $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_1$ is a bilinear mapping.
 - Selects $\alpha, b \in_R \mathbb{Z}_p$, and sets MK, PP as $MK = (\alpha, b), PP = (\mathbb{S}, g, g^b, g^{b^2}, h^b, e(g, g)^\alpha)$, where g is a generator of \mathbb{G} and h is random element of \mathbb{G} .
 - Keeps MK secret to itself and makes PP public.
- $(sk_u) \leftarrow \text{KeyGen}(PP, MK, ID_u)$: On receiving user identity $ID_u \in \mathbb{Z}_p$ from user u , the PKGC selects $t \in_R \mathbb{Z}_p$ and generates a secret key $sk_u = (d_0, d_1, d_2)$, using $MK = (\alpha, b)$, and extracting g, g^b, g^{b^2} from PP as

$$d_0 = g^\alpha (g^{b^2})^t, d_1 = [g^{b(ID_u)} h]^t, d_2 = g^{-t}.$$

It sends sk_u to user u through a secure communication channel between them.

- $(CT', S) \leftarrow \text{Encrypt}(S, PP, M)$: The content provider and the broadcaster agree on a set of subscribed user identities $S \subseteq \mathcal{U}$. Let $R_1 = \mathcal{U} \setminus S = \{ID_{i_1}, ID_{i_2}, \dots, ID_{i_{r_1}}\}$ be the set of identities of revoked users and $I_{R_1} = \{i_1, i_2, \dots, i_{r_1}\}$ be the index set of R_1 . The content provider executes the following steps to encrypt $M \in \mathbb{G}_1 (= \mathcal{M})$ for the user set S , using PP, R_1 .
 - Picks $s \in_R \mathbb{Z}_p$ and divides s into r_1 components $s_{i_1}, s_{i_2}, \dots, s_{i_{r_1}} \in \mathbb{Z}_p$ such that $s = s_{i_1} + s_{i_2} + \dots + s_{i_{r_1}}$ and generates the encrypted content $CT' = (c_0, \{c_{i,1}\}_{i \in I_{R_1}}, \{c_{i,2}\}_{i \in I_{R_1}}, c_M)$ by setting

$$c_0 = g^s, c_{i,1} = (g^b)^{s_{i_1}}, c_{i,2} = (g^{b^2 ID_i} h^b)^{s_{i_1}}, c_M = M \cdot [e(g, g)^\alpha]^s.$$

Here $g, g^b, g^{b^2}, h^b, e(g, g)^\alpha$ are extracted from PP.

- Sends CT' together with S to the broadcaster through a secure communication channel between the broadcaster and the content provider.
- $(CT, G) \leftarrow \text{Revoke}(PP, CT', S, R_2)$: Let the broadcaster wants to revoke the set $R_2 = \{ID_{i_{r_1+1}}, \dots, ID_{i_{r_1+r_2}}\}$ from CT' . Let $I_{R_2} = \{i_{r_1+1}, i_{r_1+2}, \dots, i_{r_1+r_2}\}$ be the index set of R_2 and $I_R = I_{R_1} \cup I_{R_2}$. The broadcaster generates ciphertext for the set $G = S \setminus R_2$ using PP, $CT' = (c_0, \{c_{i,1}\}_{i \in I_{R_1}}, \{c_{i,2}\}_{i \in I_{R_1}}, c_M)$, R_2 as follows
 - Selects $s_{i_{r_1+1}}, \dots, s_{i_r} \in_R \mathbb{Z}_p$ and sets $\rho = s + \sum_{i \in I_{R_2}} s_i$ where $r = r_1 + r_2$.

– Computes $C_0 = c_0 g^{\sum_{i \in I_{R_2}} s_i} = g^{(s + \sum_{i \in I_{R_2}} s_i)} = g^\rho$,

$$C_{i,1} = \begin{cases} c_{i,1}, & \text{if } i \in I_{R_1}, \\ g^{bs_i}, & \text{if } i \in I_{R_2}, \end{cases} \quad C_{i,2} = \begin{cases} c_{i,2}, & \text{if } i \in I_{R_1}, \\ (g^{b^2 ID_i} h^b)^{s_i}, & \text{if } i \in I_{R_2}, \end{cases}$$

$$C_M = c_M (e(g, g)^\alpha)^{\sum_{i \in I_{R_2}} s_i} = M \cdot e(g, g)^{\alpha(s + \sum_{i \in I_{R_2}} s_i)} = M \cdot e(g, g)^{\alpha\rho}.$$

– Broadcasts the ciphertext $\text{CT} = (C_0, \{C_{i,1}\}_{i \in I_R}, \{C_{i,2}\}_{i \in I_R}, C_M)$ together with $G = S \setminus R_2$.

- $(M) \leftarrow \text{Decrypt}(\text{PP}, sk_u, \text{CT}, G)$: A subscribed user u with identity ID_u , having the secret key $sk_u = (d_0, d_1, d_2)$, uses PP, the ciphertext $\text{CT} = (C_0, \{C_{i,1}\}_{i \in I_R}, \{C_{i,2}\}_{i \in I_R}, C_M)$, $R = R_1 \cup R_2 = \mathcal{U} \setminus G$ and recovers the message M as follows:

$$\text{– Obtain } e(g, g)^{\alpha\rho} = \frac{e(C_0, d_0)}{e(d_1, \prod_{i \in I_R} C_{i,1}^{1/(ID_u - ID_i)}) \cdot e(d_2, \prod_{i \in I_R} C_{i,2}^{1/(ID_u - ID_i)})}.$$

Here $I_R = I_{R_1} \cup I_{R_2}$.

– Recovers the message M by computing $C_M / e(g, g)^{\alpha\rho}$.

We have given the correctness of RRBE-II in Appendix B.

Remark 5. Note that in our scheme RRBE-II, we have used Lewko-Sahai-Waters Identity-Based Revocation (IBR) [15] scheme which has ciphertext size linear to the number of revoked users. Consequently, the communication cost in our RRBE-II is linear to number of revoked users. The cost can be reduced further if IBR scheme of Lewko-Sahai-Waters is replaced by a suitable IBR with constant size ciphertext. Attrapadung-Libert-Panafieu proposed an IBR [5] with constant size ciphertext. However this scheme cannot be employed in place of Lewko-Sahai-Waters IBR as it cannot extend the revocation set as illustrated below.

The public parameter PP and master key MK for the IBR of [5] are respectively given by $\text{PP} = (A = e(g, g)^\alpha, g, g^{\bar{\alpha}})$, $\text{MK} = \alpha$, where $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_1$ is a bilinear mapping, \mathbb{G}, \mathbb{G}_1 are two multiplicative groups of prime order p where discrete logarithm problem is hard, g is a generator of \mathbb{G} , $\alpha \in_R \mathbb{Z}_p^$, $\bar{\alpha} = (\alpha_1, \dots, \alpha_{r+1}) \in_R (\mathbb{Z}_p^*)^{r+1}$, r is maximum revocation number that the system can support, $g^{\bar{\alpha}} = (g^{\alpha_1}, \dots, g^{\alpha_{r+1}})$. The content generated by the content provider is given by $\text{CT} = (c_0, c_1, c_2) = (MA^s, g^s, g^{s \langle Y_{R_1}, \bar{\alpha} \rangle})$, where $M \in \mathbb{G}_1$ is the content to be encrypted, $s \in_R \mathbb{Z}_p$, $Y_{R_1} = (a_0, a_1, \dots, a_{r_1-1}, a_{r_1} = 1)$ is the vector co-efficient of $f_{R_1} = \prod_{ID_i \in R_1} (Z - ID_i)$ with $R_1 = (ID_1, ID_2, \dots, ID_{r_1}) \in (\mathbb{Z}_p^*)^{r_1}$. Here $\langle Y_{R_1}, \bar{\alpha} \rangle = \sum_{i=0}^r a_i \alpha_{i+1}$ is the inner product. If $r_1 < r$ then set $a_{r_1+1} = a_{r_1+2} = \dots = a_r = 0$ to compute c_2 . In revocation phase the broadcaster wants to revoke $R_2 = (ID_{r_1+1}, ID_{r_1+2}, \dots, ID_{r_1+r_2}) \in (\mathbb{Z}_p^*)^{r_2}$, where $R_1 \cap R_2 = \emptyset$ and $|R_1 \cup R_2| \leq r$ for which he needs to generate $\text{CT} = (C_0 = c_0, C_1 = c_1, C_2) = (MA^s, g^s, g^{s \langle Y_{R_1 \cup R_2}, \bar{\alpha} \rangle})$, where $Y_{R_1 \cup R_2} = (b_0, b_1, \dots, b_{r_1+r_2-1}, b_{r_1+r_2} = 1)$ is the vector co-efficient of $f_{R_1 \cup R_2}(Z) = \prod_{ID_i \in R_1 \cup R_2} (Z - ID_i)$. The broadcaster will not be able to extend the revocation set without explicitly knowing s or α_i values as $C_2 = g^{s \langle Y_{R_1 \cup R_2}, \bar{\alpha} \rangle}$ is not computable with the knowledge of $g^s, g^{\bar{\alpha}}, Y_{R_1 \cup R_2}$ only.*

Remark 6. If no user revoke in revocation phase and $\mathcal{U} = \mathcal{S}$ in encryption phase, then encryption, decryption are not defined. To avoid this case, we can add a dummy user in \mathcal{U} which will be always revoked.

Remark 7. As in RRBE-I scheme, the broadcaster in our RRBE-II also will be unable to modify the encrypted content in order to allow more users to recover the original message from the encrypted content as $s, \{s_{i_j}\}_{j=1}^r$ are chosen by the content provider.

Remark 8. We can remove the secure communication between the content provider and the broadcaster as follows. In the key generation phase, the PKGC can select a random number $\eta \in \mathbb{G}_1$ and sends it securely to both the content provider and the broadcaster. The content provider publishes the following encrypted content $\text{CT}' = (c_0, \{c_{i,1}\}_{i \in I_{R_1}}, \{c_{i,2}\}_{i \in I_{R_1}}, E_\eta(c_M))$, by encrypting c_M using a symmetric key encryption with η as the symmetric key. As the broadcaster has the secret symmetric key η , it can recover c_M by decrypting $E_\eta(c_M)$ and run the revocation algorithm as in RRBE-II. Note that only the ciphertext component c_M involves the content M , thereby other components $c_0, \{c_{i,1}\}_{i \in I_{R_1}}, \{c_{i,2}\}_{i \in I_{R_1}}$ can be made public. Similar mechanism is applicable for RRBE-I.

Remark 9. In CT , $C_{i,1}$ and $C_{i,2}$ are generated involving the set R_1, R_2 and any one can check by verifying the followings- $e(C_{i,1}, (g^{b^{2\text{ID}_i} h^b})) = e(C_{i,2}, g^b)$ for $i \in I_R = I_{R_1} \cup I_{R_2}$. This is verifiable using public parameters.

4.2 Security

Theorem 2. Our proposed scheme RRBE-II described in Sect. 4.1 achieves selective semantic (indistinguishability against CPA) security as per the message indistinguishability security game of Sect. 2.2 under the q -DMEBDH ($q \geq r$) assumption where r is the number of revoked users.

We have given the proof of Theorem 2 in Appendix B.

5 Conclusion

We have proposed two RRBE schemes, one achieves adaptive security whereas other is selective secure. Both of them are secure in standard model and compare well with existing similar schemes. Our RRBE-II is the first RRBE where encryption is done using the revocation identity set.

Appendix

A Correctness and Security of RRBE-I

A.1 Correctness:

Let $ID_i \in G$. Now if $|G| > 1$, we have

$$\begin{aligned}
 & \left[e(C_1, d_{1,i}) e\left(\widehat{C}_1, (h_i g^{d_{2,i}})^{A_{i,G,\alpha}}\right) \right]^{\left\{ \frac{1}{\prod_{\substack{ID_j \in G, \\ j \neq i}} ID_j} \right\}} \\
 &= \left[e(g, h_i g^{r_i})^{s \left\{ \prod_{\substack{ID_j \in G, \\ j \neq i}} (\alpha + ID_j) \right\}} \times e(g, h_i g^{r_i})^{-s \left\{ \prod_{\substack{ID_j \in G, \\ j \neq i}} (\alpha + ID_j) - \prod_{\substack{ID_j \in G, \\ j \neq i}} ID_j \right\}} \right]^{\left\{ \frac{1}{\prod_{\substack{ID_j \in G, \\ j \neq i}} ID_j} \right\}} \\
 &= \left[e(g, h_i g^{r_i})^{s \prod_{\substack{ID_j \in G, \\ j \neq i}} ID_j} \right]^{\left\{ \frac{1}{\prod_{\substack{ID_j \in G, \\ j \neq i}} ID_j} \right\}} = e(g, h_i g^{r_i})^s,
 \end{aligned}$$

and if $|G| = 1$, i.e., all but 1 user revoked, we have

$$e(C_1, d_{1,i}) = e\left(g^{s\alpha\beta(\alpha+ID_i)}, (h_i g^{r_i})^{\frac{1}{\alpha\beta(\alpha+ID_i)}}\right) = e(g, h_i g^{r_i})^s.$$

Consequently, $e(g, h_i g^{r_i})^s C_2^{d_{2,i}} = e(g, h_i g^{r_i})^s e(g, g)^{-sr_i} = e(g, h_i)^s$.

Similarly, $\left[e(C_1, d_{3,i}) e\left(\widehat{C}_1, (h_i l_0^{d_{2,i}})^{B_{G,\alpha}}\right) \right]^{\left\{ \frac{1}{\prod_{ID_j \in G} ID_j} \right\}} = e(g, h_i l_0^{r_i})^s$,

and, $\left\{ \frac{e(g, h_i l_0^{r_i})^s}{e(g, h_i)^s} \right\}^{\frac{1}{d_{2,i}}} = \left\{ \frac{e(g, h_i)^s e(g, l_0^{r_i})^s}{e(g, h_i)^s} \right\}^{\frac{1}{r_i}} = e(g, l_0)^s = K$.

The message is then recovered by computing $\frac{C_M}{K} = \frac{M \cdot e(g, l_0)^s}{e(g, l_0)^s} = M$.

A.2 Security

Theorem 1. *Our proposed scheme RRBE-I described in Sect. 3.1 achieves adaptive semantic (indistinguishability against CPA) security as per the message indistinguishability security game of Sect. 2.2 under the q -wDABDHE ($q \geq 2N$) assumption.*

Proof. Assume that there is a PPT adversary \mathcal{A} that breaks the adaptive semantic security of our proposed RRBE-I scheme with a non-negligible advantage. We construct a PPT distinguisher \mathcal{C} that attempts to solve the q -wDABDHE problem using \mathcal{A} as a subroutine. Let \mathcal{C} be given a q -wDABDHE ($q \geq 2N$) instance $\langle Z, K \rangle$ with $Z = (\mathbb{S}, \hat{g}, \hat{g}^{\alpha^{q+2}}, \dots, \hat{g}^{\alpha^{2q}}, g, g^\alpha, \dots, g^{\alpha^q})$, where $\mathbb{S} = (p, \mathbb{G}, \mathbb{G}_1, e)$ is a prime order bilinear group system, g is generator of group \mathbb{G} , $\hat{g} \in_R \mathbb{G}$, $\alpha \in_R \mathbb{Z}_p$, K is either $e(\hat{g}, g)^{\alpha^{q+1}}$ or a random element X of \mathbb{G}_1 . We describe below the interaction of \mathcal{A} with the distinguisher \mathcal{C} who attempts to output 0 if $K = e(\hat{g}, g)^{\alpha^{q+1}}$ and 1 otherwise.

Setup: The challenger \mathcal{C} generates the public parameter PP and master key MK as follows:

- Chooses $b_{0,j} \in_R \mathbb{Z}_p, j \in [0, N-1]$ and sets the polynomials $P^0(x), Q^0(x)$ as

$$P^0(x) = \sum_{j=0}^{N-1} b_{0,j} x^j, Q^0(x) = xP^0(x) + 1.$$

- Using $g, g^\alpha, \dots, g^{\alpha^q}$ sets $l_0^{\alpha^i} = g^{\alpha^i} \prod_{j=0}^{N-1} (g^{\alpha^{j+i+1}})^{b_{0,j}} = g^{\alpha^i Q^0(\alpha)}, i \in [0, N]$.
- Picks $\beta \in_R \mathbb{Z}_p$ and sets $\text{MK} = (\alpha, \beta)$, where α is not known to \mathcal{C} explicitly.
- Sets $\text{PP} = (\mathbb{S}, l_0, l_0^\alpha, \dots, l_0^{\alpha^N}, g, g^\alpha, \dots, g^{\alpha^N}, g^{\alpha^\beta}, \dots, g^{\alpha^{N+1}\beta}, e(g, g), e(g, l_0))$, and sends it to the adversary \mathcal{A} .

As $Q^0(x), \beta$ are random, the distribution of the public parameter PP is identical to that in the original scheme.

Phase 1: The adversary \mathcal{A} issues m key generation queries on $\{\text{ID}_{i_j}\}_{j=1}^m$. The challenger \mathcal{C} generates the private key sk_i for $i \in \{i_1, \dots, i_m\}$ as follows:

- Chooses $b_i, b_{i,j} \in_R \mathbb{Z}_p, j \in [0, N-2]$ and sets

$$P^i(x) = \sum_{j=0}^{N-2} b_{i,j} x^j, Q^i(x) = x(x + \text{ID}_i)P^i(x) + b_i.$$

- Computes $d_{1,i} = \left(\prod_{j=0}^{N-2} (g^{\alpha^j})^{b_{i,j}} \right)^{\frac{1}{\beta}} = \left(g^{\sum_{j=0}^{N-2} b_{i,j} \alpha^j} \right)^{\frac{1}{\beta}} = g^{\frac{P^i(\alpha)}{\beta}},$

$$d_{2,i} = -Q^i(-\text{ID}_i) = \text{ID}_i(-\text{ID}_i + \text{ID}_i)P^i(-\text{ID}_i) - b_i = -b_i,$$

$$\begin{aligned} d_{3,i} &= \left(\prod_{j=0}^{N-1} (g^{\alpha^j})^{-b_i b_{0,j}} \prod_{j=0}^{N-2} \{(g^{\alpha^{j+1}})^{b_{i,j}} (g^{\alpha^j})^{b_{i,j} \text{ID}_i}\} \right)^{\frac{1}{\beta}} \\ &= \left(g^{-b_i \sum_{j=0}^{N-1} b_{0,j} \alpha^j} g^{\{(\alpha + \text{ID}_i) \sum_{j=0}^{N-2} b_{i,j} \alpha^j\}} \right)^{\frac{1}{\beta}} = \left(g^{-b_i P^0(\alpha) + (\alpha + \text{ID}_i) P^i(\alpha)} \right)^{\frac{1}{\beta}}, \end{aligned}$$

$$\begin{aligned} h_i^{\alpha^k} &= (g^{\alpha^k})^{b_i} \prod_{j=0}^{N-2} \{(g^{\alpha^{k+j+2}})^{b_{i,j}} (g^{\alpha^{k+j+1}})^{b_{i,j} \text{ID}_i}\} \\ &= g^{\alpha^k \left(\alpha(\alpha + \text{ID}_i) P^i(\alpha) + b_i \right)} = g^{\alpha^k Q^i(\alpha)}. \end{aligned}$$

- Sets $\text{label}_i = (h_i^{\alpha^k}, k \in [0, N])$ and $sk_i = (d_{1,i}, d_{2,i}, d_{3,i}, \text{label}_i)$.

Sends sk_i for $i \in \{i_1, \dots, i_m\}$ to the adversary \mathcal{A} .

As $b_i, Q^i(x)$ are random, $d_{2,i}, \text{label}_i$ have identical distribution to those in the original scheme. It is left to show that $d_{1,i}, d_{3,i}$ follow the original distribution.

$$d_{1,i} = g^{\frac{P^i(\alpha)}{\beta}} = g^{\frac{Q^i(\alpha) - b_i}{\alpha\beta(\alpha + \text{ID}_i)}} = (g^{Q^i(\alpha)} g^{d_{2,i}})^{\frac{1}{\alpha\beta(\alpha + \text{ID}_i)}} = (h_i g^{d_{2,i}})^{\frac{1}{\alpha\beta(\alpha + \text{ID}_i)}},$$

$$\begin{aligned}
\text{Now, } -b_i P^0(\alpha) + (\alpha + \text{ID}_i) P^i(\alpha) &= \frac{1}{\alpha} \left\{ -b_i \alpha P^0(\alpha) + Q^i(\alpha) - b_i \right\} \\
&= \frac{1}{\alpha} \left\{ -b_i (Q^0(\alpha) - 1) + Q^i(\alpha) - b_i \right\} = \frac{1}{\alpha} \left\{ -b_i Q^0(\alpha) + Q^i(\alpha) \right\} \\
\Rightarrow d_{3,i} &= \left(g^{-b_i P^0(\alpha) + (\alpha + \text{ID}_i) P^i(\alpha)} \right)^{\frac{1}{\beta}} = g^{\frac{1}{\alpha\beta} \left\{ -b_i Q^0(\alpha) + Q^i(\alpha) \right\}} = (h_i l_0^{d_{2,i}})^{\frac{1}{\alpha\beta}}.
\end{aligned}$$

Thus $d_{1,i}, d_{3,i}$ are identical to original scheme.

Challenge: The adversary \mathcal{A} sends a set of user identities G to \mathcal{C} , where identities of G have not been queried before. It also sends two equal length messages M_0, M_1 , and a revocation identity set R to the challenger \mathcal{C} where no identity in the set R lies in G . The challenger \mathcal{C} does the following:

- Computes $\prod_{i=0}^{N-1} (g^{\alpha^i})^{b_{0,i}} = g^{\sum_{i=0}^{N-1} b_{0,i} \alpha^i} = g^{P^0(\alpha)}$ by extracting g^{α^i} values from the given instance $\langle Z, K \rangle$.
- Selects $M_\mu, \mu \in_R \{0, 1\}$ and sets C_{M_μ} as, $C_{M_\mu} = M_\mu \cdot K \cdot e(\hat{g}^{\alpha^{q+2}}, g^{P^0(\alpha)})$, where K is extracted from the given instance $\langle Z, K \rangle$. Here K is either $e(\hat{g}, g)^{\alpha^{q+1}}$ or a random element of \mathbb{G}_1 . If $K = e(\hat{g}, g)^{\alpha^{q+1}}$ then the simulated $C_{M_\mu} (= c_{M_\mu})$ has the same distribution as in the original scheme as

$$\begin{aligned}
C_{M_\mu} &= M_\mu \cdot K \cdot e(\hat{g}^{\alpha^{q+2}}, g^{P^0(\alpha)}) = M_\mu \cdot e(\hat{g}, g)^{\alpha^{q+1}} e(\hat{g}^{\alpha^{q+2}}, g^{P^0(\alpha)}) \\
&= M_\mu \cdot e(\hat{g}^{\alpha^{q+1}}, g^{\alpha^{P^0(\alpha)+1}}) = M_\mu \cdot e(g^s, l_0) = M_\mu \cdot e(g, l_0)^s
\end{aligned}$$

where s is implicitly set as $s = \alpha^{q+1} \log_g \hat{g}$.

- Sets $\Gamma(x) = \prod_{\text{ID}_j \in G} (x + \text{ID}_j) = \sum_{i=0}^{|G|} \Gamma_i x^i$, where Γ_i are function of $\text{ID}_j \in G$.
- Computes $\prod_{i=0}^{|G|} (\hat{g}^{\alpha^{q+2+i\beta}})^{\Gamma_i} = (\hat{g}^{\alpha^{q+2\beta}})^{\sum_{i=0}^{|G|} \Gamma_i \alpha^i} = (\hat{g}^{\alpha^{q+2\beta}})^{\prod_{\text{ID}_i \in G} (\alpha + \text{ID}_i)}$. Note that $\hat{g}^{\alpha^i}, i \in [q+2, 2q], q \geq 2N$ are available to \mathcal{C} through $\langle Z, K \rangle$.
- If $R \neq \phi$, sets the challenge ciphertext CT^* as,

$$\text{CT}^* = \left((\hat{g}^{\alpha^{q+2\beta}})^{\prod_{\text{ID}_i \in G} (\alpha + \text{ID}_i)}, K^{-1}, \hat{g}^{-\alpha^{q+2}}, C_{M_\mu} \right) = (C_1, C_2, \hat{C}_1, C_{M_\mu}).$$

else if $R = \phi$ (i.e. $G = S$), sets CT^* as

$$\begin{aligned}
\text{CT}^* &= \left((\hat{g}^{\alpha^{q+2\beta}})^{\prod_{\text{ID}_i \in G} (\alpha + \text{ID}_i)}, K^{-1}, \hat{g}^{-\alpha^{q+2}}, \hat{g}^{\alpha^{q+3}}, \dots, \hat{g}^{\alpha^{q+k+1}}, C_{M_\mu} \right) \\
&= (c_1, c_2, \hat{c}_1, \hat{c}_2, \dots, \hat{c}_{k+1}, c_{M_\mu}).
\end{aligned}$$

If $K = e(\hat{g}, g)^{\alpha^{q+1}}$, then as s is implicitly set to be $s = \alpha^{q+1} \log_g \hat{g}$, we have

$$\begin{aligned} C_1 &= c_1 = (\hat{g}^{\alpha^{q+2}\beta})_{\text{ID}_i \in G}^{\prod (\alpha + \text{ID}_i)} = (g^{\beta \log_g \hat{g}} \hat{g}^{\alpha^{q+2}})_{\text{ID}_i \in G}^{\prod (\alpha + \text{ID}_i)} \\ &= (g^{\beta \alpha \alpha^{q+1} \log_g \hat{g}})_{\text{ID}_i \in G}^{\prod (\alpha + \text{ID}_i)} = (g^{\alpha \beta})_s^{\prod (\alpha + \text{ID}_i)}, \\ C_2 &= c_2 = K^{-1} = e(g^{\log_g \hat{g}}, g)^{-\alpha^{q+1}} = e(g, g)^{-\alpha^{q+1} \log_g \hat{g}} = e(g, g)^{-s}, \\ \widehat{C}_1 &= \hat{c}_1 = \hat{g}^{-\alpha^{q+2}} = g^{-(\log_g \hat{g}) \alpha^{q+2}} = g^{-\alpha \alpha^{q+1} \log_g \hat{g}} = g^{-\alpha s}, \\ \hat{c}_i &= \hat{g}^{\alpha^{q+1+i}} = g^{\alpha^i \alpha^{q+1} \log_g \hat{g}} = g^{\alpha^i s}, 2 \leq i \leq k. \end{aligned}$$

Consequently, distribution of CT^* is similar to our real construction from \mathcal{A} 's point of view.

– Returns CT^* to \mathcal{A} .

Note that in our RRBE-I (see Sect. 3.1), components c_1, c_2, \hat{c}_1, c_M generated for message M in **Encrypt** are identical to $C_1, C_2, \widehat{C}_1, C_M$ of **Revoke** respectively except randomness. Therefore in this **Challenge** phase, from adversary \mathcal{A} 's point of view there is no difference between $(c_1, c_2, \hat{c}_1, c_{M_\mu})$ and $(C_1, C_2, \widehat{C}_1, C_{M_\mu})$. Consequently we can take $C_1 = c_1, C_2 = c_2, \widehat{C}_1 = \hat{c}_1, C_{M_\mu} = c_{M_\mu}$ as challenge ciphertext in $R \neq \phi$ case.

Phase 2: This is similar to Phase 1 key generation queries. The adversary \mathcal{A} sends key generation queries for $\{\text{ID}_{i_{m+1}}, \dots, \text{ID}_{i_t}\}$ with a restriction that $\text{ID}_{i_j} \notin G$ and receives back secret keys $\{sk_{i_j}\}_{j=m+1}^t$ simulated in the same manner by \mathcal{C} as in Phase 1.

Guess: Finally, \mathcal{A} outputs a guess $\mu' \in \{0, 1\}$ of μ to \mathcal{C} and wins if $\mu' = \mu$.

If $\mu' = \mu$, \mathcal{C} outputs 0, indicating that $K = e(\hat{g}, g)^{\alpha^{q+1}}$; otherwise, it outputs 1, indicating that K is a random element of \mathbb{G}_1 .

The simulation of \mathcal{C} is perfect when $K = e(\hat{g}, g)^{\alpha^{q+1}}$. Therefore, we have $Pr[\mathcal{C}(Z, K = e(\hat{g}, g)^{\alpha^{q+1}}) = 0] = \frac{1}{2} + Adv_{\mathcal{A}, \text{RRBE-I}}^{\text{IND-aCPA}}$, where $Adv_{\mathcal{A}, \text{RRBE-I}}^{\text{IND-aCPA}}$ is the advantage of the adversary \mathcal{A} in the above indistinguishability game. On the other hand, M_μ is completely hidden from the adversary \mathcal{A} when $K = X$ is random, thereby $Pr[\mathcal{C}(Z, K = X) = 0] = \frac{1}{2}$. Hence, the advantage of the challenger \mathcal{C} in solving q -wDABDHE is

$$\begin{aligned} Adv_{\mathcal{C}}^{q\text{-wDABDHE}} &= |Pr[\mathcal{C}(Z, K = e(\hat{g}, g)^{\alpha^{q+1}}) = 0] - Pr[\mathcal{C}(Z, K = X) = 0]| \\ &= \frac{1}{2} + Adv_{\mathcal{A}, \text{RRBE-I}}^{\text{IND-aCPA}} - \frac{1}{2} = Adv_{\mathcal{A}, \text{RRBE-I}}^{\text{IND-aCPA}}. \end{aligned}$$

Therefore, if \mathcal{A} has non-negligible advantage in correctly guessing μ' , then \mathcal{C} predicts $K = e(\hat{g}, g)^{\alpha^{q+1}}$ or random element of \mathbb{G}_1 (i.e., solves q -wDABDHE ($q \geq 2N$) instance given to \mathcal{C}) with non-negligible advantage. Hence the theorem.

B Correctness and Security of RRBE-II

B.1 Correctness:

The correctness of RRBE-II follows from the argument below:

If $ID_u \in G$, then

$$\begin{aligned}
& \frac{e(C_0, d_0)}{e(d_1, \prod_{i \in I_R} C_{i,1}^{\left\{ \frac{1}{ID_u - ID_i} \right\}}) \cdot e(d_2, \prod_{i \in I_R} C_{i,2}^{\left\{ \frac{1}{ID_u - ID_i} \right\}})} \\
&= \frac{e(g^\rho, g^\alpha g^{b^2 t})}{\prod_{i \in I_R} \left\{ e((g^{bID_u} h)^t, g^{bs_i}) \cdot e(g^{-t}, (g^{b^2 ID_i} h^b)^{s_i}) \right\}^{\left\{ \frac{1}{ID_u - ID_i} \right\}}} \\
&= \frac{e(g, g)^{\alpha \rho} e(g, g)^{\rho b^2 t}}{\prod_{i \in I_R} \left\{ e(g, g)^{b^2 ID_u s_i t} \cdot e(g, g)^{-b^2 ID_i s_i t} \right\}^{\left\{ \frac{1}{ID_u - ID_i} \right\}}} \\
&= \frac{e(g, g)^{\alpha \rho} e(g, g)^{\rho b^2 t}}{\prod_{i \in I_R} e(g, g)^{s_i b^2 t}} = e(g, g)^{\alpha \rho},
\end{aligned}$$

and consequently, $\frac{C_M}{e(g, g)^{\alpha \rho}} = \frac{M \cdot e(g, g)^{\alpha \rho}}{e(g, g)^{\alpha \rho}} = M$.

B.2 Security

Theorem 2. *Our proposed scheme RRBE-II described in Sect. 4.1 achieves selective semantic (indistinguishability against CPA) security as per the message indistinguishability security game of Sect. 2.2 under the q -DMEBDH ($q \geq r$) assumption where r is the number of revoked users.*

Proof. Assume that there is a PPT adversary \mathcal{A} that breaks the selective semantic security of our proposed RRBE-II scheme with a non-negligible advantage. We construct a PPT distinguisher \mathcal{C} that attempts to solve the q -DMEBDH problem using \mathcal{A} as a subroutine. Let \mathcal{C} be given a q -DMEBDH ($q \geq r$) instance $\langle Z, K \rangle$

$$\text{with } Z = \left\{ \begin{array}{ll} \mathbb{S}, g, g^s, e(g, g)^\alpha, & \\ g^{a_i}, g^{a_i s}, g^{a_i a_j}, g^{\alpha/a_i^2}, & \forall 1 \leq i, j \leq q \\ g^{a_i a_j s}, g^{\alpha a_j/a_i^2}, g^{\alpha a_i a_j/a_k^2}, g^{\alpha a_i^2/a_j^2}, & \forall 1 \leq i, j, k \leq q, i \neq j \end{array} \right\}$$

where $\mathbb{S} = (p, \mathbb{G}, \mathbb{G}_1, e)$ is a prime order bilinear group system, g is a generator of the group \mathbb{G} , $\alpha, a_i \in_R \mathbb{Z}_p$, and K is either $e(g, g)^{\alpha s}$ or a random element X of \mathbb{G}_1 . We describe below the interaction of \mathcal{A} with the distinguisher \mathcal{C} who

attempts to output 0 if $K = e(g, g)^{\alpha s}$ and 1 otherwise. Both adversary \mathcal{A} and challenger \mathcal{C} knows the universal set of users \mathcal{U} .

Initialization: The adversary \mathcal{A} selects a target identity set G of subscribed users. Let $R = \mathcal{U} \setminus G = \{\text{ID}_{i_1}, \dots, \text{ID}_{i_r}\}$ and $I_R = \{i_1, \dots, i_r\}$ be index set for R .

Setup: The challenger \mathcal{C} generates the public parameter PP using $\langle Z, K \rangle$ as:

- Selects $y \in_R \mathbb{Z}_p$ and implicitly sets $b = a_{i_1} + a_{i_2} + \dots + a_{i_r}$.
- Computes $g^b = \prod_{i \in I_R} g^{a_i}$, $g^{b^2} = \prod_{i, j \in I_R} g^{a_i a_j}$.
- Sets h, h^b as $h = \prod_{i \in I_R} (g^{a_i})^{-\text{ID}_i} g^y$, $h^b = \prod_{i, j \in I_R} (g^{a_i a_j})^{-\text{ID}_i} g^{a_j y}$. Since α, b are not known to \mathcal{C} , master key MK is explicitly implicitly set as $\text{MK} = (\alpha, b)$.
- Sends $\text{PP} = (\mathbb{S}, g, g^b, g^{b^2}, h^b, e(g, g)^\alpha)$ to the adversary \mathcal{A} .

Note that as b, α are random and therefore, PP has the same distribution as in the original RRBE-II.

Phase 1: The adversary \mathcal{A} issues m key generation queries on $\{\text{ID}_{i_j}\}_{j=1}^m$, where $\text{ID}_j \notin G$. Receiving key generation query for ID_i , the challenger \mathcal{C} does the followings:

- Selects $z_i \in_R \mathbb{Z}_p$ and using y chosen randomly from \mathbb{Z}_p in Setup phase, it sets

$$\begin{aligned}
 d_0 &= \left(\prod_{\substack{j, k \in I_R \text{ such that} \\ \text{if } j=k \text{ then } j, k \neq i}} g^{-\alpha a_j a_k / a_i^2} \right) \prod_{j, k \in I_R} (g^{a_j a_k})^{z_i} \\
 &= g^\alpha \prod_{j, k \in I_R} g^{-\alpha a_j a_k / a_i^2} \prod_{j, k \in I_R} (g^{a_j a_k})^{z_i} = g^\alpha g^{b^2 t_i}, \text{ where } t_i = z_i - \alpha / a_i^2, \\
 d_1 &= \left(\prod_{\substack{j \in I_R \\ j \neq i}} (g^{-\alpha a_j / a_i^2})^{(\text{ID}_i - \text{ID}_j)} (g^{(\text{ID}_i - \text{ID}_j) a_j})^{z_i} \right) (g^{-\alpha / a_i^2})^y g^{y z_i} \\
 &= \left(\prod_{\substack{j \in I_R \\ j \neq i}} (g^{(\text{ID}_i - \text{ID}_j) a_j t_i}) \right) g^{y t_i} = g^{b \text{ID}_i t_i} \left(\prod_{j \in I_R} (g^{-\text{ID}_j a_j t_i}) g^{y t_i} \right) = g^{b \text{ID}_i t_i} h^{t_i}, \\
 d_2 &= g^{\alpha / a_i^2} g^{-z_i} = g^{-t_i}.
 \end{aligned}$$

- Sets $sk_i = (d_0, d_1, d_2)$ and sends sk_i to the adversary \mathcal{A} .

Note that d_0, d_1, d_2 have similar distributions as in the original scheme RRBE-II.

Challenge: The adversary \mathcal{A} sends two equal length messages M_0, M_1 to the challenger \mathcal{C} together with a set of revoked user identities $R_2 (\subseteq R)$ which will be revoked in the revocation phase. The challenger \mathcal{C} does the following:

- Selects $s' \in_R \mathbb{Z}_p$ and split it into $r = |R|$ components $s'_{i_1}, s'_{i_2}, \dots, s'_{i_r} \in \mathbb{Z}_p$ such that $s' = s'_{i_1} + s'_{i_2} + \dots + s'_{i_r}$.

- Sets $C_0 = g^s g^{s'}$ and $u_i = g^{b^2 \text{ID}_i} h^b$ for each $i \in I_R$.
- Selects $\mu \in_R \{0, 1\}$ and computes $C_{M_\mu} = M_\mu \cdot K \cdot e(g, g)^{\alpha s'}$. Also for each $i \in I_R$, computes the followings

$$C_{i,1} = g^{s a_i} (\prod_{j \in I_R} g^{a_j})^{s'_i}, C_{i,2} = \prod_{\substack{j \in I_R \\ j \neq i}} (g^{s a_i a_j})^{(\text{ID}_i - \text{ID}_j)} (g^{a_i s})^y u_i^{s'_i},$$
- If $R_2 = \phi$ (i.e., $G = S$), then sends the challenge ciphertext $\text{CT}^* = \text{CT}' = (C_0, \{C_{i,1}\}_{i \in I_{R_1}}, \{C_{i,2}\}_{i \in I_{R_1}}, C_{M_\mu})$ to \mathcal{A} .
- If $R_2 \neq \phi$, then sends the challenge ciphertext $\text{CT}^* = \text{CT} = (C_0, \{C_{i,1}\}_{i \in I_R}, \{C_{i,2}\}_{i \in I_R}, C_{M_\mu})$ to \mathcal{A} .

Let us set implicitly $\rho = s + s'$ and split ρ as $\rho = \sum_{i \in I_R} \rho_i$ with $\rho_i = \frac{a_i s}{b} + s'_i$ (implicitly). If $K = e(g, g)^{\alpha s}$, then these ciphertexts have distribution identical to those in the original protocol as follows:

$$\begin{aligned} C_0 &= g^s g^{s'} = g^\rho, \quad C_{i,1} = g^{s a_i} \left(\prod_{j \in I_R} g^{a_j} \right)^{s'_i} = g^{s a_i} g^{b s'_i} = g^{b \rho_i}, \\ C_{i,2} &= \prod_{\substack{j \in I_R \\ j \neq i}} (g^{s a_i a_j})^{(\text{ID}_i - \text{ID}_j)} (g^{a_i s})^y u_i^{s'_i} = \prod_{j \in I_R} \frac{(g^{s a_i a_j})^{(\text{ID}_i - \text{ID}_j)}}{g^{s a_i a_i (\text{ID}_i - \text{ID}_i)}} g^{y s a_i} u_i^{s'_i} \\ &= g^{b \text{ID}_i s a_i} \prod_{j \in I_R} g^{-s a_i a_j \text{ID}_j} g^{s a_i y} u_i^{s'_i} = (g^{b \text{ID}_i} h)^{s a_i + b s'_i} = (g^{b^2 \text{ID}_i} h^b)^{\rho_i}. \\ C_M &= M_\mu \cdot K \cdot e(g, g)^{\alpha s'} = M_\mu \cdot e(g, g)^{\alpha s} \cdot e(g, g)^{\alpha s'} = M_\mu \cdot e(g, g)^{\alpha \rho}. \end{aligned}$$

Note that in our RRBE-II, the ciphertext components $(c_0, \{c_{i,1}\}_{i \in I_{R_1}}, \{c_{i,2}\}_{i \in I_{R_1}}, c_M)$ generated for a message M by working `Encrypt` are identical to $(C_0, \{C_{i,1}\}_{i \in I_R}, \{C_{i,2}\}_{i \in I_R}, C_M)$ generated by executing `Encrypt` followed by `Revoke` except from number of components. Therefore, if $R_2 = \phi$, the adversary will consider the challenge ciphertext CT^* as an output of `Encrypt`(G, PP, M_μ) and if $R_2 \neq \phi$ it will be considered as an output of `Encrypt`(S, PP, M_μ), `Revoke`($\text{PP}, \text{CT}', S, R_2$).

Phase 2: This is similar to Phase 1 key generation queries. The adversary \mathcal{A} sends key generation queries for $\{\text{ID}_{i_{m+1}}, \dots, \text{ID}_{i_t}\}$ with a restriction that $i_j \notin G$ and receives back secret keys $\{sk_{i_j}\}_{j=m+1}^t$ simulated in the same manner by \mathcal{C} as in Phase 1.

Guess: Finally, \mathcal{A} outputs a guess $\mu' \in \{0, 1\}$ of μ to \mathcal{C} and wins if $\mu' = \mu$. If $\mu' = \mu$, \mathcal{C} outputs 0, indicating that $K = e(g, g)^{\alpha s}$; otherwise, it outputs 1, indicating that K is a random element of \mathbb{G}_1 .

The simulation of \mathcal{C} is perfect when $K = e(g, g)^{\alpha s}$. Therefore, we have $Pr[\mathcal{C}(Z, K = e(g, g)^{\alpha s}) = 0] = \frac{1}{2} + Adv_{\mathcal{A}, \text{RRBE-II}}^{\text{IND-sCPA}}$, where $Adv_{\mathcal{A}, \text{RRBE-II}}^{\text{IND-sCPA}}$ is the advantage of the adversary \mathcal{A} in the above indistinguishability game. On the other hand, M_μ is completely hidden from the adversary \mathcal{A} when $K = X$ is random, thereby $Pr[\mathcal{C}(Z, K = X) = 0] = \frac{1}{2}$. Hence, the advantage of the challenger \mathcal{C} in solving q -DMEBDH is $Adv_{\mathcal{C}}^{q\text{-DMEBDH}} = |Pr[\mathcal{C}(Z, K = e(g, g)^{\alpha s}) = 0] - Pr[\mathcal{C}(Z, K = X) = 0]| = \frac{1}{2} + Adv_{\mathcal{A}, \text{RRBE-II}}^{\text{IND-sCPA}} - \frac{1}{2} = Adv_{\mathcal{A}, \text{RRBE-II}}^{\text{IND-sCPA}}$.

Therefore, if \mathcal{A} has non-negligible advantage in correctly guessing μ' , then \mathcal{C} predicts $K = e(g, g)^{\alpha s}$ or random element of \mathbb{G}_1 (i.e., solves q -DMEBDH ($q \geq r$) instance given to \mathcal{C}) with non-negligible advantage. Hence the theorem follows.

References

1. Acharya, K., Dutta, R.: Secure and efficient construction of broadcast encryption with dealership. In: Chen, L., Han, J. (eds.) *ProvSec 2016*. LNCS, vol. 10005, pp. 277–295. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47422-9_16
2. Acharya, K., Dutta, R.: Adaptively secure broadcast encryption with dealership. In: Hong, S., Park, J.H. (eds.) *ICISC 2016*. LNCS, vol. 10157, pp. 161–177. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-53177-9_8
3. Acharya, K., Dutta, R.: Enhanced outsider-anonymous broadcast encryption with subset difference revocation. *IACR Cryptol. ePrint Arch.* **2017**, 265 (2017)
4. Acharya, K., Dutta, R.: Provable secure constructions for broadcast encryption with personalized messages. In: Okamoto, T., Yu, Y., Au, M.H., Li, Y. (eds.) *ProvSec 2017*. LNCS, vol. 10592, pp. 329–348. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68637-0_20
5. Attrapadung, N., Libert, B., de Panafieu, E.: Expressive key-policy attribute-based encryption with constant-size ciphertexts. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) *PKC 2011*. LNCS, vol. 6571, pp. 90–108. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19379-8_6
6. 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). https://doi.org/10.1007/11426639_26
7. Boneh, D., Gentry, C., Waters, B.: Collusion resistant broadcast encryption with short ciphertexts and private keys. In: Shoup, V. (ed.) *CRYPTO 2005*. LNCS, vol. 3621, pp. 258–275. Springer, Heidelberg (2005). https://doi.org/10.1007/11535218_16
8. Boneh, D., Waters, B., Zhandry, M.: Low overhead broadcast encryption from multilinear maps. In: Garay, J.A., Gennaro, R. (eds.) *CRYPTO 2014*. LNCS, vol. 8616, pp. 206–223. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_12
9. Boneh, D., Zhandry, M.: Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In: Garay, J.A., Gennaro, R. (eds.) *CRYPTO 2014*. LNCS, vol. 8616, pp. 480–499. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_27
10. Delerablée, C.: Identity-based broadcast encryption with constant size ciphertexts and private keys. In: Kurosawa, K. (ed.) *ASIACRYPT 2007*. LNCS, vol. 4833, pp. 200–215. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-76900-2_12
11. Fiat, A., Naor, M.: Broadcast encryption. In: Stinson, D.R. (ed.) *CRYPTO 1993*. LNCS, vol. 773, pp. 480–491. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48329-2_40
12. Lai, J., Mu, Y., Guo, F., Chen, R.: Fully privacy-preserving ID-based broadcast encryption with authorization. *Comput. J.* **60**(12), 1809–1821 (2017)
13. Lai, J., Mu, Y., Guo, F., Susilo, W., Chen, R.: Anonymous identity-based broadcast encryption with revocation for file sharing. In: Liu, J.K., Steinfeld, R. (eds.) *ACISP 2016*. LNCS, vol. 9723, pp. 223–239. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-40367-0_14

14. Lai, J., Mu, Y., Guo, F., Susilo, W., Chen, R.: Fully privacy-preserving and revocable ID-based broadcast encryption for data access control in smart city. *Pers. Ubiquitous Comput.* **2017**, 855–868 (2017)
15. Lewko, A., Sahai, A., Waters, B.: Revocation systems with very small private keys. In: *IEEE Symposium on Security and Privacy*, pp. 273–285 (2010)
16. Ren, Y., Wang, S., Zhang, X.: Non-interactive dynamic identity-based broadcast encryption without random oracles. In: Chim, T.W., Yuen, T.H. (eds.) *ICICS 2012*. LNCS, vol. 7618, pp. 479–487. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34129-8_47
17. Susilo, W., Chen, R., Guo, F., Yang, G., Mu, Y., Chow, Y.-W.: Recipient revocable identity-based broadcast encryption: how to revoke some recipients in IBBE without knowledge of the plaintext. In: *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security, ASIA CCS 2016*, pp. 201–210 (2016)



Recipient Revocable Broadcast Encryption with Dealership

Joon Sik Kim^(✉), Youngkyung Lee, Jieun Eom, and Dong Hoon Lee

Graduate School of Information Security, Korea University, Seoul, Korea
{ha29re,dudrudve,donghlee}@korea.ac.kr, jieunn.eom@gmail.com

Abstract. The broadcast encryption with dealership (BED) scheme allows a dealer, instead of a broadcaster, to manage a recipient. Unlike prior broadcast encryption schemes, BED reduces the burden placed on the broadcaster to manage recipient, which makes it suitable for a broadcasting service targeting a large number of recipients. Subscribing and unsubscribing from the broadcast service occur frequently at the request of the user, however, early versions of BED schemes do not support recipient revocation. In this paper, we propose a recipient revocable broadcast encryption with dealership and show that it is secure in the adaptive security model without random oracles.

Keywords: Broadcast encryption with dealership
Adaptive security · Revocation · Chosen plaintext attack

1 Introduction

Broadcast encryption (BE) is a useful tool that enables a single ciphertext to be decrypted by multiple recipients. A broadcaster encrypts a message intended for a group of recipients instead of for a single recipient, and anyone belonging to the recipient group can decrypt the ciphertext with its secret key. Since a ciphertext is broadcast to multiple recipients, BE is suitable for a service transmitting digital content such as Pay-TV [5]. As the demand for digital content has increased, researchers have studied a variety of ways to set a recipients group [1, 5, 8, 13]. However, the increasing number of recipients has made it important to manage all recipients as well as to specify groups of recipients.

To solve this issue, Gritti et al. proposed a broadcast encryption with dealership (BED) scheme in 2015 that focuses on the separation of roles, i.e. selecting a set of recipients and providing content to them [11]. In BED, there is a dealer who sets the recipient group, unlike in BE. The dealer directly manages recipients instead of the broadcaster without knowing the content, and obtains a

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2015-0-00320, A study of a public-key authentication framework for internet entities with hierarchical identities).

commission from the broadcaster for this service. The recipient can purchase content from the dealer at a lower price than purchasing from the broadcaster. Specifically, the dealer creates a group token without exceeding its own authority, which is defined as the maximum number of recipients whom he is allowed to manage by contract, and sends it to the broadcaster. The broadcaster verifies if the group token is valid and uses it to encrypt the content. The recipient can decrypt the encrypted content using a secret key if he or she belongs to the group.

Although BED helps recipient management, requests for subscription or unsubscription of recipients changes dynamically in real online multimedia services. The recipient may receive the wrong content owing to a mistake by the dealer or the broadcaster and may request a refund for receiving incorrect services. In such a scenario, the recipient may ask the dealer to revoke his or her access of channel. To revoke the recipient's access using prior BED schemes, the dealer must generate another group token excluding the revoked user for a new ciphertext because only the dealer knows the recipient group he has set. After completing verification of the new group token, the broadcaster encrypts content again. The process is inefficient for both the dealer and the broadcaster: [(1) group token generation - (2) group token verification - (3) content encryption] whenever recipients request the dealer to revoke their access. If the dealer can revoke recipients without the broadcaster's help, the revocation process mentioned above will become much more efficient. In this paper, we first propose a recipient revocable broadcast encryption with dealership (RR-BED) that can revoke recipients more simply than prior BED schemes.

1.1 Our Contribution

In this paper, we first construct a BED scheme that supports recipient revocation. There are two types of revocation methods in BE that revoke a secret key from the system and a recipient from the ciphertext. Since the secret key revocation system excludes a recipient from the whole system, the recipient revocation method, which revokes a recipient only for a specific content, is more suitable for BED applications. There are some studies on BE that support recipient revocation [4, 15] and we use the similar technique to construct our scheme. In specific, there is a *Revoke* algorithm that allows a dealer to revoke recipients efficiently without the complex process mentioned above. The dealer removes a set of recipients from the original group defined in the old ciphertext using the *Revoke* algorithm. To generate a ciphertext for a new group, the *Revoke* algorithm takes as inputs a group R of recipients to be revoked and the ciphertext CT for the old group G and outputs an updated ciphertext CT' for a new group $S = G - R$. The *Revoke* algorithm should be executed by the dealer because only the dealer knows which clients are in the recipient group. At this point, the dealer performs revocation without assistance, and the revocation interactions between the dealer and the broadcaster are no longer necessary.

In our RR-BED, there are three security requirements. First, the broadcaster must not be able to obtain any recipients' information from the group token. If

recipient information in the group token is exposed to the broadcaster, then the broadcaster can directly sell the access of channel to a recipient at the same price that the dealer sells, without owing a commission to the dealer. The second is that the dealer cannot generate a group token exceeding his own authority. If a group token forged by the dealer passes verification, even though it exceeds the dealer's authority, the dealer can get an unfair commission. The final requirement is that an unauthorized user or the dealer cannot get any information about content from the ciphertext. We prove its security under the discrete logarithm problem, the $(q+i)$ -Diffie-Hellman exponent problem, and the q -weak decisional augmented bilinear Diffie-Hellman exponent problem.

1.2 Related Work

The first BED scheme for a new business model was proposed by Gritti et al. in 2015 [11]. The authors designed the BED scheme by leveraging the concept of group tokens used in a membership encryption and presented a new security model. The security of the scheme is proved in the semi-static security model in which an adversary specifies a target group at the beginning of the attack. In 2016, Acharya and Dutta pointed out the problem of [11] that a malicious dealer can generate a group token for a larger group beyond his own privilege [2]. They proposed a more efficient BED scheme and proved the security of their scheme in the semi-static security model with random oracles. After that, Acharya and Dutta proposed an improved BED scheme that does not use random oracles for the security proof and is proved in the adaptive security model that allows an adversary to query secret keys before a target is set [3]. However, none of the existing BED schemes support revocation.

Whereas there is no BED scheme that supports revocation, there have been many studies of BE that support revocation [4, 6, 10, 12, 15]. Some of these studies were combining BE with the trace system and revocation of the secret key [6, 10, 12]. On the other hand some studies focused on revoking the recipients rather than revoking the secret key [4, 15]. The notion of recipient revocable broadcast encryption (RR-BE) was first proposed by Susilo et al. in 2016 [15] and after that, Acharya et al. improved the security model of RR-BE from selective to adaptive security [4].

The rest of this paper is organized as follows. We describe preliminaries in Sect. 2, and we define the algorithms of RR-BED and the security model in Sect. 3. We then present our RR-BED construction in Sect. 4. The security proof of the proposed scheme is provided in Sect. 5. Finally, we summarize our results in Sect. 6.

2 Preliminaries

In this section, we describe bilinear map and give the complexity assumptions for our scheme.

2.1 Bilinear Map

Let \mathbb{G} and \mathbb{G}_T be multiplicative cyclic groups of prime order p and let g be a generator of \mathbb{G} . The bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ has the following properties:

- Bilinearity: for all $u, v \in \mathbb{G}$ and for all $a, b \in \mathbb{Z}_p$, $e(u^a, v^b) = e(u, v)^{ab}$
- Non-degeneracy: for generator $G \in \mathbb{G}$, $e(g, g) \neq 1_{\mathbb{G}_T}$, where $1_{\mathbb{G}_T}$ is an identity element in \mathbb{G}_T .

We say that $\mathbb{B} = (p, \mathbb{G}, \mathbb{G}_T, e)$ is a bilinear group if the group action in \mathbb{G} can be computed efficiently and there exists a \mathbb{G}_T for which the bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is efficiently computable.

2.2 Complexity Assumptions

We use the following complexity assumptions to prove that the scheme we propose is secure.

Definition 1 (Discrete Logarithm (DL) assumption). Let \mathbb{G} be a multiplicative cyclic group of order p . The DL Problem in \mathbb{G} is stated as follows: Given a tuple $\langle Z = (g, g^\alpha) \rangle$, an attacker \mathcal{A} outputs α where $g \in \mathbb{G}$ and random $\alpha \in \mathbb{Z}_p$. \mathcal{A} has an advantage ϵ in solving the DL problem in \mathbb{G} if

$$Adv_{\mathcal{A}}^{DL} = |Pr[A(Z)] = \alpha| \geq \epsilon.$$

We say that the DL assumption holds in \mathbb{G} if no polynomial-time algorithm has a non-negligible advantage in solving the DL problem.

Definition 2 (($N + i$)-Diffie-Hellman Exponent (DHE) ($i > 0$) assumption [7]). Let $\mathbb{B} = (p, \mathbb{G}, \mathbb{G}_T, e)$ be a bilinear group. The ($N + i$)-DHE problem in \mathbb{G} is stated as follows : Given a tuple $\langle Z = (\mathbb{B}, g, g^\alpha, \dots, g^{\alpha^N}) \rangle$, an attacker \mathcal{A} outputs $g^{\alpha^{N+i}}$ where $g \in \mathbb{G}$ and random $\alpha \in \mathbb{Z}_p$. \mathcal{A} has an advantage ϵ in solving the ($N+i$)-DHE problem in \mathbb{G} if

$$Adv_{\mathcal{A}}^{(N+i)\text{-DHE}} = |Pr[\mathcal{A}(Z) = g^{\alpha^{N+i}}]| \geq \epsilon.$$

We say that the ($N + i$)-DHE assumption holds in \mathbb{G} if no polynomial-time algorithm has a non-negligible advantage in solving the ($N+i$)-DHE problem.

Definition 3 (q -weaker Decisional Augmented Bilinear Diffie-Hellman Exponent (q -wDABDHE) assumption [14]). Let $\mathbb{B} = (p, \mathbb{G}, \mathbb{G}_T, e)$ be a bilinear group. The q -wDABDHE problem in \mathbb{G} is stated as follows: Given a tuple $\langle Z = (\mathbb{B}, u, u^{\alpha^{q+2}}, \dots, u^{\alpha^{2q}}, g, g^\alpha, \dots, g^{\alpha^q}), K \rangle$, an attacker \mathcal{A} outputs 1 if $K = e(g, u)^{\alpha^{q+1}}$ and outputs 0 if $K = R$ for random $R \in \mathbb{G}_T$ where $g \in \mathbb{G}$, random $u \in \mathbb{G}$, and random $\alpha \in \mathbb{Z}_p$. \mathcal{A} has an advantage ϵ in solving the q -wDABDHE problem in \mathbb{G} if

$$Adv_{\mathcal{A}}^{q\text{-wDABDHE}} = |Pr[\mathcal{A}(Z, K = e(g, u)^{\alpha^{q+1}}) = 1] - Pr[\mathcal{A}(Z, K = R) = 1]| \geq \epsilon.$$

We say that the q -wDABDHE assumption holds in \mathbb{G} if no polynomial-time algorithm has a non-negligible advantage in solving the q -wDABDHE problem.

3 Recipient Revocable Broadcast Encryption with Dealership

In this section, we describe the definitions of recipient revocable broadcast encryption with dealership (RR-BED) and its security models.

3.1 Definition

We give an overview of our system model. There are four entities participating in the RR-BED system that a key generation center (KGC), a dealer, a broadcaster, and a recipient. KGC sets up the system and issues secret keys to the recipients. The dealer has responsibility to manage recipients' information and decide whether the recipient can access to the content or not, which are kept secret to the other entities even to the broadcaster. It generates a group token for the set of recipients and sends it to the broadcaster. When the broadcaster obtains the group token, it first verifies the validity of it. If it is valid, the broadcaster encrypts the content with the hidden information of the set of recipients and uploads it to the server (or a cloud storage). Then, only the authorized recipients can decrypt the ciphertext. RR-BED is composed of the following seven algorithms (Setup, KeyGen, GroupGen, Verify, Encrypt, Revoke, Decrypt).

RR-BED.Setup(λ, N): This algorithm takes as inputs a security parameter λ and the maximum number of users N . It outputs public parameters PP and a master key MK .

RR-BED.KeyGen(i, MK, PP): This algorithm takes as inputs a user index i , the master key MK , and the public parameters PP . It outputs a secret key sk_i for user i .

RR-BED.GroupGen(S, v, k, PP): This algorithm takes as inputs a group of users $S = \{i_1, i_2, \dots, i_k\}$, a threshold value v , a maximum number of revocations k , and the public parameters PP . It outputs a group token $P(S)$ where $|S| \leq v$. Note that v is determined by the broadcaster and the dealer in advance as the maximum number of recipients that a dealer can service.

RR-BED.Verify($P(S), v, PP$): This algorithm takes as inputs a group token $P(S)$, the threshold value v , and the public parameters PP . If $|S| \leq v$, it outputs 1, otherwise, it outputs 0.

RR-BED.Encrypt($P(S), M, PP$): This algorithm takes as inputs a group token $P(S)$, a message M , and the public parameters PP . It outputs a ciphertext CT .

RR-BED.Revoke(CT, R, PP): This algorithm takes as inputs a ciphertext CT , a set of users R to be revoked, and the public parameters PP . It outputs an updated ciphertext CT' for a user group $G = S - R$.

RR-BED.Decrypt(sk_i, CT, G, PP): This algorithm takes as inputs a secret key sk_i , a ciphertext CT , a user group G , and the public parameters PP . It outputs message M .

Correctness. The correctness of RR-BED is as follows: If PP, MK are generated by **RR-BED.Setup**(λ, N), CT is generated by **RR-BED.Encrypt**($P(S), M, PP$), CT' is generated by **RR-BED.Revoke**(CT, R, PP), and sk_i is generated by **RR-BED.KeyGen**(i, MK, PP) for every recipient i in the recipient group G , then

$$\mathbf{RR-BED.Decrypt}(sk_i, CT', G, PP) = M.$$

3.2 Security Models

There are three security issues in RR-BED. These are privacy, maximum number of accountability, and message indistinguishability under chosen plaintext attacks. We describe each issue through the following security models.

(Privacy). RR-BED must ensure privacy for its user set as does the original BED. In more detail, information regarding the set of users in a group token should not be known to the broadcaster. The original security model for this privacy in BED was introduced by Gritti et al. [11]. We follow their security model.

Definition 4. *Privacy in RR-BED is defined using the game between a probabilistic polynomial-time (PPT) adversary \mathcal{A} and a challenger \mathcal{C} . The game proceeds as follows:*

Setup: *The challenger \mathcal{C} runs **RR-BED.Setup**(N, λ) and obtains PP, MK. It keeps MK and gives PP to the adversary \mathcal{A} .*

Challenge: *\mathcal{A} selects two user groups G_0, G_1 of the same size and submits to \mathcal{C} . \mathcal{C} picks $b \in \{0, 1\}$ and generates group token $P(G_b)$ by running **RR-BED.GroupGen**(G_b, v, k, PP), where $|G_b| \leq v$ and k is a maximum revocation number. \mathcal{C} gives $P(G_B)$ to \mathcal{A} .*

Guess: *\mathcal{A} outputs a guess $b' \in \{0, 1\}$. If $b = b'$, \mathcal{A} wins.*

The advantage of \mathcal{A} is defined as $Adv_{\mathcal{A}}^{\text{Privacy}} = |Pr[b' = b] - \frac{1}{2}|$. An RR-BED scheme is guaranteed privacy of all users in the group token if for every PPT adversary \mathcal{A} , the advantage of \mathcal{A} in the above game is negligible.

(Maximum number of accountability). In RR-BED, the dealer should not exceed the number of privileges granted by the broadcaster when the dealer generates a group token. The broadcaster must ensure that a the group size of a group token is not larger than the number of privileges granted to the dealer. We follow Gritti et al.'s security model to achieve this goal [11].

Definition 5. *Maximum number of accountability in RR-BED is defined using the following game between a PPT adversary \mathcal{A} and a challenger \mathcal{C} :*

Setup: *The challenger \mathcal{C} runs **RR-BED.Setup**(N, λ) and obtains PP, MK. It keeps MK and gives PP to the adversary \mathcal{A} .*

Challenge: *\mathcal{C} sends an integer k to \mathcal{A} .*

Guess: *\mathcal{A} computes $P(G^*)$, with $|G^*| > k$. \mathcal{A} sends $(P(S^*), S^*)$ to \mathcal{C} . If **RR-BED.Verify**($P(S^*), k, PP$)=1, \mathcal{A} wins.*

The advantage of \mathcal{A} is defined as $Adv_{\mathcal{A}}^{Max} = |Pr[RR-BED.Verify(P(S^*), k, PP) = 1]|$. An RR-BED scheme is secure regarding maximum number of accountability if for every PPT adversary \mathcal{A} , the advantage of \mathcal{A} in the above game is negligible.

(Message indistinguishability under CPA). The adaptive security model of BED was introduced by Acharya et al. [3], but it does not consider revocation. We define a chosen plaintext attack of RR-BED against an adaptive adversary with respect to the security model of RR-BE [4, 15]. In the adaptive security model, the adversary is allowed to make queries before choosing the target group that it wishes to attack.

Definition 6. The adaptive security in RR-BED is defined using the game between a PPT adversary \mathcal{A} and a challenger \mathcal{C} . The game proceeds as follows:

Setup: The challenger \mathcal{C} runs $RR-BED.Setup(N, \lambda)$ and obtains PP, MK . It keeps MK and gives PP to \mathcal{A} .

Phase 1: \mathcal{A} adaptively requests a polynomial number of queries for a user index $i \in \{i_1, \dots, i_m\}$. \mathcal{C} runs $RR-BED.KeyGen(i, MK, PP)$ on i and gives the resulting secret key sk_i to \mathcal{A} .

Challenge: \mathcal{A} submits two equal length messages M_0^*, M_1^* and a challenge identity group G^* which has not been queried. \mathcal{A} also submits a maximum revocation number $k < n$ and a revocation set R^* to \mathcal{C} , where $G^* \cap R^* = \phi$. \mathcal{C} sets $S^* = G^* + R^*$ and runs $RR-BED.GroupGen(S, v, k, PP)$. \mathcal{C} obtains $P(S^*)$ and picks $b \in \{0, 1\}$. \mathcal{C} generates $CT = RR-BED.Encrypt(P(S^*), M_b, PP)$ and $CT' = RR-BED.Revoke(CT, R^*, PP)$. If $R = \phi$, \mathcal{C} sets $CT^* = CT$ else it sets $CT^* = CT'$ and gives CT^* to \mathcal{A} .

Phase 2: \mathcal{A} may continue the key generation queries, and \mathcal{C} responds as **Phase 1**. One restriction is that \mathcal{A} cannot make queries for i in G^* .

Guess: \mathcal{A} outputs a guess $b' \in \{0, 1\}$. If $b = b'$, \mathcal{A} wins.

The advantage of \mathcal{A} is defined as $Adv_{\mathcal{A}}^{IND-CPA} = |Pr[b' = b] - \frac{1}{2}|$. An RR-BED scheme is semantically secure under a chosen plaintext attack if for every PPT adversary \mathcal{A} , the advantage of \mathcal{A} in the above game is negligible.

4 RR-BED Scheme

In this section, we present our concrete construction of the RR-BED scheme. We construct our RR-BED scheme using BED [11] and RR-BE [4] as building blocks. We extend BED using revocation technique of RR-BE so that the dealer who only knows the set of recipients is allowed to revoke the recipients. The proposed scheme is described as follows.

RR-BED.Setup(λ, N): It generates a bilinear group $\mathbb{B} = (p, \mathbb{G}, \mathbb{G}_T, e)$ and let g be a generator of \mathbb{G} . It randomly chooses $\alpha, \beta \in \mathbb{Z}_p$, and $h \in \mathbb{G}$, and computes $h^\alpha, \dots, h^{\alpha^N}, g^\alpha, \dots, g^{\alpha^N}$, and $g^{\alpha\beta}, \dots, g^{\alpha^{N+1}\beta}$. Then, it outputs $PP = (\mathbb{B}, h, h^\alpha, \dots, h^{\alpha^N}, g, g^\alpha, \dots, g^{\alpha^N}, g^{\alpha\beta}, \dots, g^{\alpha^{N+1}\beta}, \Omega = e(g, g), \Omega_1 = e(g, h), \mathbb{ID})$ and $MK = (\alpha, \beta)$, where $\mathbb{ID} = \{ID_1, ID_2, \dots, ID_N\}$ is a set of user identities.

RR-BED.KeyGen(i, MK, PP): It chooses random $r_i \in \mathbb{Z}_p, l_i \in \mathbb{G}$, and computes $d_{1,i} = (l_i g^{r_i})^{\frac{1}{\alpha\beta(\alpha+ID_i)}}$ and $d_{3,i} = (l_i h^{r_i})^{\frac{1}{\alpha\beta}}$. It sets $d_{2,i} = r_i$ and outputs $sk_i = (d_{1,i}, d_{2,i}, d_{3,i}, l_i, l_i^\alpha, \dots, l_i^{\alpha^N})$.

RR-BED.GroupGen(S, v, k, PP): It chooses a random $r \in \mathbb{Z}_p$ and computes the following values, where $F(x) = \prod_{i_j \in S} (x + ID_{i_j}) = \sum_{i=0}^{k'} F_i x^i$:

$$\begin{aligned} w_1 &= \prod_{i=0}^{k'} (g^{\alpha^{i+1}\beta})^{rF_i} = g^{\alpha\beta F(\alpha)r}, \\ w_2 &= \prod_{i=0}^{k'} (g^{\alpha^{N-v+i+1}\beta})^{rF_i} = g^{\alpha^{N-v+1}\beta F(\alpha)r}, \\ w_3 &= \Omega^{-r}, [\hat{w}_i] = [g^{-\alpha^i r}]_{1 \leq i \leq k+1}, w_M = \Omega_1^r. \end{aligned}$$

Finally, it outputs a group token $P(S) = (w_1, w_2, w_3, \hat{w}_1, \dots, \hat{w}_{k+1}, w_M)$.

Note that a dealer must set a group size k' satisfying $k' \leq v$ so as not to exceed the number of its own privileges v . The dealer sends $P(S)$ to each recipient through a secure channel.

RR-BED.Verify($P(S), v, PP$): It verifies $e(w_1, g^{\alpha^N}) = e(w_2, g^{\alpha^v})$. If it holds true, it outputs 1, otherwise it outputs 0. The following equations show the correctness of the verification:

$$\begin{aligned} e(w_1, g^{\alpha^N}) &= e(g^{\alpha\beta F(\alpha)r}, g^{\alpha^N}) = e(g, g)^{\alpha^{N+1}\beta F(\alpha)r}, \\ e(w_2, g^{\alpha^v}) &= e(g^{\alpha^{N-v+1}\beta F(\alpha)r}, g^{\alpha^v}) = e(g, g)^{\alpha^{N+1}\beta F(\alpha)r}. \end{aligned}$$

Note that if **RR-BED.Verify**($P(S), v, PP$) outputs 1, the broadcaster generates a ciphertext, otherwise it aborts.

RR-BED.Encrypt($P(S), M, PP$): It chooses random $t \in \mathbb{Z}_p$ and outputs a ciphertext $CT = (c_1, c_2, \hat{c}_1, \dots, \hat{c}_{k+1}, c_M)$ as follows:

$$\begin{aligned} CT &= (c_1, c_2, \hat{c}_1, \dots, \hat{c}_{k+1}, c_M) \\ &= (w_1^t, w_2^t, \hat{w}_1^t, \dots, \hat{w}_{k+1}^t, w_M^t) \\ &= (g^{\alpha\beta F(\alpha)rt}, \Omega^{-rt}, g^{-\alpha rt}, \dots, g^{-\alpha^{k+1}rt}, M\Omega_1^{rt}) \\ &= (g^{\alpha\beta F(\alpha)s}, \Omega^{-s}, g^{-\alpha s}, \dots, g^{-\alpha^{k+1}s}, M\Omega_1^s), \text{ where } rt = s. \end{aligned}$$

RR-BED.Revoke(CT, R, PP): Let $CT = (c_1, c_2, \hat{c}_1, \dots, \hat{c}_{k+1}, c_M)$ and $R = \{i_1, \dots, i_l\} \subseteq S$ where $l \leq k$. It generates $CT' = (C_1, C_2, \hat{C}_1, C_M)$ as follows:

1. If $R = \phi$, $CT' = (C_1, C_2, \hat{C}_1, C_M) = (c_1, c_2, \hat{c}_1, c_M)$
2. If $R \neq \phi$, it computes $\frac{\prod_{j \in R} (x + ID_j)}{\prod_{j \in R} (ID_j)} = \sum_{i=0}^l f_i x^i$, where $f_0 = 1$, and $H = \prod_{i=2}^l \hat{c}_i^{f_i} = g^{-r \sum_{i=2}^l f_i \alpha^i}$. It sets $y = s \sum_{i=0}^l f_i \alpha^i$ and computes

C_1, C_2, \hat{C}_1, C_M as follows:

$$\begin{aligned}
C_1 &= c_1^{\frac{1}{\prod_{j \in R} ID_j}} = g^{\alpha\beta s \frac{\prod_{j \in R} (\alpha + ID_j) \prod_{j \in G} (\alpha + ID_j)}{\prod_{j \in R} (ID_j)}} = g^{\alpha\beta y \{\prod_{j \in G} (\alpha + ID_j)\}}, \\
C_2 &= c_2 e(g, \hat{c}_1^{f_1} H) = \Omega^{-s} e(g, g)^{-s(f_1\alpha + f_2\alpha^2 \cdots f_l\alpha^l)} \\
&= \Omega^{-s(1+f_1\alpha+f_2\alpha^2 \cdots f_l\alpha^l)} = \Omega^{-y}, \\
\hat{C}_1 &= \prod_{i=1}^{l+1} \hat{c}_i^{f_i-1} = (g^{-\alpha s})(g^{-\alpha^2 s}) f_1 (g^{-\alpha^3 s}) f_2 \dots (g^{-\alpha^{l+1} s}) f_l \\
&= g^{-\alpha s(1+f_1\alpha+\cdots+f_l\alpha^l)} = g^{-\alpha y}, \\
C_M &= c_M e(\hat{c}_1^{-f_1} H^{-1}, h) = M \Omega_1^s e(g, h)^{s(f_1\alpha+f_2\alpha^2+\cdots+f_l\alpha^l)} \\
&= M \Omega_1^{s(1+f_1\alpha+f_2\alpha^2+\cdots+f_l\alpha^l)} = M \Omega_1^y.
\end{aligned}$$

3. It then outputs a new ciphertext $CT' = (C_1, C_2, \hat{C}_1, C_M) = (c_1, c_2, \hat{c}_1, c_M)$ for a user group $G = S - R$.

RR-BED.Decrypt(sk_i, CT, G, PP): Let $CT = (C_1, C_2, \hat{C}_1, C_M)$ and The message M is recovered as follows:

1. It first computes $A_{i,G,\alpha} = \frac{1}{\alpha} (\prod_{j \in G, j \neq i} (\alpha + ID_j) - \prod_{j \in G, j \neq i} ID_j)$ and $B_{i,G,\alpha} = \frac{1}{\alpha} (\prod_{j \in G} (\alpha + ID_j) - \prod_{j \in G} (ID_j))$
2. It then computes $e(g, l_i)^y$ and $e(g, l_i h^{r_i})^y$

$$\begin{aligned}
e(C_1, d_{1,i}) e(\hat{C}_1, (l_i g^{d_{2,i}})^{A_{i,G,\alpha}})]^{\frac{1}{\prod_{j \in G, j \neq i} (ID_j)}} C_2^{d_{2,i}} &= e(g, l_i)^y, \\
e(C_1, d_{3,i}) e(\hat{C}_1, (l_i h^{d_{2,i}})^{B_{i,G,\alpha}})]^{\frac{1}{\prod_{j \in G} (ID_j)}} &= e(g, l_i h^{r_i})^y.
\end{aligned}$$

Finally, it computes the message $C_M (\frac{e(g, l_i)^y}{e(g, l_i h^{r_i})^y})^{\frac{1}{d_{2,i}}} = M$.

4.1 Correctness

Let $I_i = \frac{1}{\prod_{j \in G, j \neq i} ID_j}$, $A_{i,G,\alpha} = \frac{1}{\alpha} A_{i,G}$, and $B_{i,G,\alpha} = \frac{1}{\alpha} B_{i,G}$. The decryption process for the secret key sk_i of a recipient $i \in G$ and the ciphertext CT' for group G is as follows:

$$\begin{aligned}
&[e(C_1, d_{1,i}) e(\hat{C}_1, (l_i g^{d_{2,i}})^{A_{i,G,\alpha}})]^{I_i} C_2^{d_{2,i}} \\
&= [e(g^{\alpha\beta y \prod_{j \in G} (\alpha + ID_j)}, (l_i g^{r_i})^{\frac{1}{\alpha\beta(\alpha + ID_i)}}) \cdot e(g^{-\alpha y}, (l_i g^{r_i})^{\frac{1}{\alpha} A_{i,G}})]^{I_i} C_2^{d_{2,i}} \\
&= [e(g, l_i g^{r_i})^y \prod_{j \in G, j \neq i} (\alpha + ID_j) \cdot e(g, l_i g^{r_i})^{-y(\prod_{j \in G, j \neq i} (\alpha + ID_j) - \prod_{j \in G, j \neq i} ID_j)}]^{I_i} C_2^{d_{2,i}} \\
&= [e(g, l_i g^{r_i})^y \prod_{j \in G, j \neq i} ID_j]^{\frac{1}{\prod_{j \in G, j \neq i} ID_j}} e(g, g)^{-y r_i} \\
&= e(g, l_i)^y,
\end{aligned}$$

$$\begin{aligned}
 & [e(C_1, d_{3,i})e(\hat{C}_1, (l_i h^{d_{2,i}})^{B_{i,G,\alpha}})]^{\frac{1}{\prod_{j \in G} ID_j}} \\
 &= [e(g^{\alpha\beta y \prod_{j \in G} (\alpha + ID_j)}, (l_i h_i^r)^{\frac{1}{\alpha\beta}}) \cdot e(g^{-\alpha y}, (l_i h^{r_i})^{\frac{1}{\alpha} B_{i,G}})]^{\frac{1}{\prod_{j \in G} ID_j}} \\
 &= [e(g, l_i h^{r_i})^{y \prod_{j \in G} (\alpha + ID_j)} \cdot e(g, l_i h^{r_i})^{-y(\prod_{j \in G} (\alpha + ID_j) - \prod_{j \in G} ID_j)}]^{\frac{1}{\prod_{j \in G} ID_j}} \\
 &= [e(g, l_i h^{r_i})^y \prod_{j \in G} ID_j]^{\frac{1}{\prod_{j \in G} ID_j}} \\
 &= e(g, l_i h^{r_i})^y. \\
 C_M \left(\frac{e(g, l_i)^y}{e(g, l_i h^{r_i})^y} \right)^{\frac{1}{d_{2,i}}} &= C_M \left(\frac{e(g, l_i)^y}{e(g, l_i)^y e(g, h^{r_i})^y} \right)^{\frac{1}{r_i}} = M \Omega_1^y \frac{1}{e(g, h)^y} = M.
 \end{aligned}$$

5 Security Analysis

We prove the security of our RR-BED under the cryptographic assumptions described in Sect. 2. Theorems 1 and 2 are proved in a similar way to [3], and the security proof of Theorem 3 follows that of [4].

Theorem 1. *The RR-BED scheme is guaranteed the group privacy if the DL assumption holds.*

Theorem 2. *The RR-BED scheme is secure with regard to the maximum number of accountability if the $(N + i)$ -DHE assumption holds.*

The proof of the above theorems are given in Appendix A.1 and A.2.

Theorem 3. *The RR-BED scheme is semantically secure if the $q - w$ DABDHE assumption ($q \geq 2N$) holds.*

Proof. Suppose there exists a PPT adversary \mathcal{A} , that can break our RR-BED scheme in the adaptive IND-CPA security model with an advantage ϵ . We build a challenger \mathcal{C} that solves the $q - w$ DABDHE problem by using \mathcal{A} . Let $\langle Z = (\mathbb{B}, u, u^{\alpha^{q+2}}, \dots, u^{\alpha^{2q}}, g, g^\alpha, \dots, g^{\alpha^q}), K \rangle$ be given to \mathcal{C} , where \mathbb{G}, \mathbb{G}_T is the groups of order p with the bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ and g is a generator of \mathbb{G} . If $K = e(u, g)^{\alpha^{q+1}}$, then \mathcal{C} outputs 1, otherwise, it outputs 0. The simulation proceeds as follows:

Setup: \mathcal{C} randomly chooses each $b_{0,j} \in \mathbb{Z}_p$ for $j \in [0, n - 1]$. It computes $P^0(x) = \sum_{j=0}^{n-1} b_{0,j} x^j$, $Q^0(x) = x + 1$ and $h^{\alpha^i} = g^\alpha \sum_{j=0}^{n-1} (g^{\alpha^{i+j+1}})^{b_{0,j}} = g^{\alpha^i(1+\alpha P^0(\alpha))} = g^{\alpha^i Q^0 \alpha}$ for $i \in [0, n]$. It randomly chooses $\beta \in \mathbb{Z}_p$ and the identities of n users $\mathbb{ID} = \{ID_1, ID_2, \dots, ID_n\} \in \mathbb{Z}_p$. It sets $\text{MK} = (\alpha, \beta)$ and $\text{PP} = (\mathbb{B}, h, h^\alpha, \dots, h^{\alpha^N}, g, g^\alpha, \dots, g^{\alpha^N}, g^{\alpha\beta}, \dots, g^{\alpha^{N+1}\beta}, \Omega = e(g, g), \Omega_1 = e(g, h))$. Note that \mathcal{C} doesn't know α . It gives PP and MK to \mathcal{A} . Since $b_{0,j}$ and $Q^0(x)$ are chosen randomly, a distribution of PP and MK is identical to the actual construction.

Phase 1: \mathcal{A} adaptively makes key generation queries. \mathcal{C} responds to a query on $i \in \{i_1, \dots, i_m\}$. \mathcal{C} randomly chooses each $b_{i,j} \in \mathbb{Z}_p$ for $j \in [0, N-2]$ and $b_i \in \mathbb{Z}_p$ and computes $P^i(x) = \sum_{j=0}^{N-2} b_{i,j} x^j$, $Q^i(x) = x(x + ID_i)P^i(x) + b_i$. \mathcal{C} computes $d_{1,i}, d_{2,i}, d_{3,i}, l_i, l_i^\alpha, \dots, l_i^{\alpha^N}$ as follows:

$$\begin{aligned} d_{1,i} &= \left(\prod_{j=0}^{N-2} (g^{\alpha^j})^{b_{i,j}} \right)^{\frac{1}{\beta}} = (g^{\sum_{j=0}^{N-2} b_{i,j} \alpha^j})^{\frac{1}{\beta}} = g^{\frac{P^i(\alpha)}{\beta}}, \\ d_{2,i} &= -Q^i(-ID_i) = ID_i(-ID_i + ID_i)P^i(-ID_i) - b_i = b_i, \\ d_{3,i} &= \left(\prod_{j=0}^{N-1} (g^{\alpha^j})^{-b_i b_{0,j}} \prod_{j=0}^{N-2} (g^{\alpha^{j+1}})^{b_{i,j}} (g^{\alpha^j})^{b_{i,j} ID_i} \right)^{\frac{1}{\beta}} \\ &= \left(\prod_{j=0}^{N-1} g^{-b_i b_{0,j} \alpha^j} \prod_{j=0}^{N-2} g^{b_{i,j} (\alpha + ID_i) \alpha^j} \right)^{\frac{1}{\beta}} \\ &= (g^{-b_i \sum_{j=0}^{N-1} b_{0,j} \alpha^j} (g^{(\alpha + ID_i) \sum_{j=0}^{N-2} b_{i,j} \alpha^j})^{\frac{1}{\beta}} \\ &= (g^{-b_i P^0(\alpha) + (\alpha + ID_i) P^i(\alpha)})^{\frac{1}{\beta}}, \\ l_i^{\alpha^k} &= (g^{\alpha^k})^{b_i} \prod_{j=0}^{N-2} (g^{\alpha^{k+j+2}})^{b_{i,j}} (g^{\alpha^{j+i+1}})^{b_{i,j} ID_i} \\ &= g^{\alpha^k (\alpha (\alpha + ID_i) P^i(\alpha) + b_i)} = g^{\alpha^k Q^i(\alpha)} \text{ for } k \in [0, N]. \end{aligned}$$

\mathcal{C} sets $sk_i = (d_{1,i}, d_{2,i}, d_{3,i}, l_i, l_i^\alpha, \dots, l_i^{\alpha^N})$ and gives it to \mathcal{A} . Since $b_i, b_{0,j}$ and $Q^0(x)$ are chosen randomly, a distribution of sk_i is identical to the actual construction.

$$\begin{aligned} d_{1,i} &= g^{\frac{P^i(\alpha)}{\beta}} = g^{\frac{Q^i(\alpha) - b_i}{\alpha\beta(\alpha + ID_i)}} = g^{\frac{Q^i(\alpha) + d_{2,i}}{\alpha\beta(\alpha + ID_i)}} = (l_i g_i^r)^{\frac{1}{\alpha\beta(\alpha + ID_i)}}, \\ d_{3,i} &= (g^{-b_i P^0(\alpha) + (\alpha + ID_i) P^i(\alpha)})^{\frac{1}{\beta}} \\ &= (g^{(-b_i \alpha P^0(\alpha) + Q^i(\alpha) - b_i)})^{\frac{1}{\alpha\beta}} \\ &= (g^{(-b_i (Q^0(\alpha) - 1) + Q^i(\alpha) - b_i)})^{\frac{1}{\alpha\beta}} \\ &= (g^{(-b_i Q^0(\alpha) + Q^i(\alpha))})^{\frac{1}{\alpha\beta}}. \end{aligned}$$

Challenge: \mathcal{A} submits two equal length messages M_0^*, M_1^* and a challenge identity group G^* that has not been queried. \mathcal{A} also submits a maximum revocation number $k (\leq n)$ and a revocation set R^* to \mathcal{C} , where $G^* \cap R^* = \phi$. \mathcal{C} sets $S^* = G^* + R^*$ and generates a challenge ciphertext as follows: \mathcal{C} chooses random $b \in \{0, 1\}$ and computes $C_{M_b} = M_b Ke(u^{\alpha^{q+2}}, g^{P^0(\alpha)})$, where $\prod_{i=0}^{n-1} (g^{\alpha^i})^{b_{0,i}} = g^{\sum_{i=0}^{n-1} b_{0,i} \alpha^i} = g^{P^0(\alpha)}$ and $\prod_{i=0}^{|G^*|} (u^{\alpha^{q+i+2}\beta})^{f_i^* x^i} = (u^{\alpha^{q+2}\beta})^{\sum_{i=0}^{|G^*|} f_i^* \alpha^i} = (u^{\alpha^{q+2}\beta})^{\prod_{i \in G^*} (\alpha + ID_j)}$, where f_i^* is a coefficient of the polynomial $F^*(x) = \prod_{j \in G^*} (x + ID_j) = \sum_{i=0}^{|G^*|} f_i^* x^i$. If $R^* \neq \phi$, \mathcal{C} sets a ciphertext as

$$\begin{aligned} CT^* &= ((u^{\alpha^{q+2}\beta})^{\prod_{i \in G^*} (\alpha + ID_j)}, K^{-1}, u^{-\alpha^{q+2}}, C_{M_b}) \\ &= (c_1, c_2, \hat{c}_1, C_{M_b}). \end{aligned}$$

If $R^* = \phi$, \mathcal{C} sets a ciphertext as

$$\begin{aligned} CT^* &= ((u^{\alpha^{q+2}\beta})_{\prod_{i \in G^*} (\alpha + ID_j)}, K^{-1}, u^{-\alpha^{q+2}}, u^{-\alpha^{q+3}}, \dots, u^{-\alpha^{q+k+1}}, C_{M_b}) \\ &= (c_1, c_2, \hat{c}_1, \hat{c}_2, \dots, \hat{c}_{k+1}, C_{M_b}). \end{aligned}$$

If $K = e(u^{\alpha^{q+1}}, g)$, \mathcal{C} sets r as $g^r = u^{\alpha^{q+1}}$ and then,

$$\begin{aligned} c_1 &= (u^{\alpha^{q+2}\beta})_{\prod_{i \in G^*} (\alpha + ID_j)} = \{(u^{\alpha^{q+1}})^{\alpha\beta}\}_{\prod_{i \in G^*} (\alpha + ID_j)} \\ &= \{(g^r)^{\alpha\beta}\}_{\prod_{i \in G^*} (\alpha + ID_j)} = (g^{\alpha\beta r})_{\prod_{i \in G^*} (\alpha + ID_j)}, \\ c_2 &= K^{-1} = e(u, g)^{-\alpha^{q+1}} \\ &= e(u^{-\alpha^{q+1}}, g) = e(g^s, g) = e(g, g)^s, \\ \hat{c}_i &= u^{-\alpha^{q+1+i}} = (u^{-\alpha^{q+1}})^{-\alpha^i} \\ &= g^{\alpha^i s}, 1 \leq i \leq k, \\ C_{M_b} &= M_b K e(u^{\alpha^{q+2}}, g^{P^0(\alpha)}) \\ &= M_b e(u^{\alpha^{q+1}}, g) e(u^{\alpha^{q+2}}, g^{P^0(\alpha)}) \\ &= M_b e(u^{\alpha^{q+1}}, g) e(u^{\alpha^{q+1}}, g^{\alpha P^0(\alpha)}) \\ &= M_b e(u^{\alpha^{q+1}}, g^{\alpha P^0(\alpha) + 1}) \\ &= M_b e(u^{\alpha^{q+1}}, g^{Q^0(\alpha)}) = M_b e(g^s, h) \\ &= M_b e(g, h)^s. \end{aligned}$$

hence the distribution of CT^* is identical to the actual construction.

Phase 2: \mathcal{A} may continue the key generation queries, and \mathcal{C} responds as **Phase 1**. One restriction is that \mathcal{A} cannot make a query for $ID_i \in G^*$.

Guess: Finally, \mathcal{A} outputs a guess $b' \in \{0, 1\}$ and \mathcal{C} also outputs the same guess b' .

If $K = e(u^{\alpha^{q+1}}, g)$, then the simulation is the same as in the real game. Hence, \mathcal{A} will have the probability $\frac{1}{2} + \epsilon$ to guess b correctly. If K is a random element of \mathbb{G}_T , then \mathcal{A} will have probability $\frac{1}{2}$ to guess b correctly. Therefore, \mathcal{C} can solve the $q - w$ DABDHE problem also with the advantage ϵ . \square

6 Efficiency

In this section, we compare several existing BED schemes [2, 3, 11] with our RR-BED scheme in Table 1.

Comparing the computation cost of our scheme and [3], the computation cost of Verify and Decrypt are same as those of [3], and the computation costs of GroupGen and Encrypt are higher than those of [3] by kE_G , respectively. Since existing BED schemes do not provide for revocation, it is necessary to add the computation costs of [(1) GroupGen - (2) Verify - (3) Encrypt] to compare

Table 1. Comparison of BED schemes: E_G, E_{G_T} denote the number of exponentiation in \mathbb{G}, \mathbb{G}_T , P denotes the number of pairings, k' denotes the number of users selected by the dealer, l denotes the actual number of revoked users, and k denotes the number of users that can be revoked, w/o RO denotes without random oracles, and RR denotes recipient revocable.

	GroupGen	Verify	Encrypt	Revoke	Decrypt	Security model	w/o RO	RR
[11]	$(k' + 4)E_G + 1E_{G_T}$	$2P$	$2E_G + 1E_{G_T} + 2P$	$(k' - l + 6)E_G + 2E_{G_1} + 4P$	$2P$	Semi-static	O	X
[2]	$(2k' + 3)E_G + 1E_{G_T}$	$2P$	$2E_G + 1E_{G_T}$	$(2k' - l + 5)E_G + 2E_{G_1} + 2P$	$(k' - 1)E_G + 1E_{G_T} + 2P$	Selective	X	X
[3]	$(2k' + 3)E_G + 2E_{G_T}$	$2P$	$2E_G + 2E_{G_T}$	$(2k' - l + 5)E_G + 4E_{G_1} + 2P$	$(4k' - 2)E_G + 4E_{G_T} + 4P$	Adaptive	O	X
Ours	$(2k' + k + 3)E_G + 2E_{G_T}$	$2P$	$(k + 2)E_G + 2E_{G_T}$	$(2l + 1)E_G + 2P$	$(4k' - 2)E_G + 4E_{G_T} + 4P$	Adaptive	O	O

the computation costs when generating the ciphertext for a new group. However, in our proposed scheme, the computation cost of the Revoke algorithm is $(2l + 1)E_G + 2P$, which is actually lower than the overall cost of previous BED schemes. Thus, our RR-BED is more efficient when executing the overall process. Moreover, the security proof of our scheme is achieved in the adaptive security model without random oracles as in [3].

7 Conclusion

In this paper, we proposed a recipient revocable broadcast encryption with dealership. We proved the security of our scheme applying cryptographic assumptions. Our scheme provides recipient revocation without the broadcaster's help, allowing the dealer to directly revoke recipients. In real multimedia services, adding recipients is also an important issue as well as revoking recipients. Designing a BED scheme that supports both addition and revocation of recipients would be an interesting future work.

A Security Proof

A.1 Proof of Theorem 1

Proof. Let a PPT adversary \mathcal{A} breaks the privacy of our RR-BED scheme. The security game between the challenger \mathcal{C} and the adversary \mathcal{A} is executed as follows:

Setup: The challenger \mathcal{C} randomly chooses $\alpha, \beta \in \mathbb{Z}_p$ and $h \in \mathbb{G}$ and generates $\text{PP} = (\mathbb{B}, h, h^\alpha, \dots, h^{\alpha^N}, g, g^\alpha, \dots, g^{\alpha^N}, g^{\alpha\beta}, \dots, g^{\alpha^{N+1}\beta}, \Omega = e(g, g), \Omega_1 = e(g, h))$ and $\text{MK} = (\alpha, \beta)$. It keeps MK secret and gives PP to \mathcal{A} .

Challenge: \mathcal{A} selects two user groups G_0, G_1 of the same size and submits to \mathcal{C} . \mathcal{C} picks $b \in \{0, 1\}$ and runs $\text{RR-BED.GroupGen}(G_b, v, k, \text{PP})$ to obtain a group token.

$$\begin{aligned} P(G_b) &= (w_1, w_2, w_3, \hat{w}_1, \dots, \hat{w}_{k+1}, w_M) \\ &= (g^{\alpha\beta F(\alpha)r}, \Omega^{-r}, g^{-\alpha r}, g^{-\alpha^i r}, \dots, g^{-\alpha^{k+1}r}, \Omega_1^r). \end{aligned}$$

where $v \geq |G_b|$, k is a maximum revocation number, and $F(x) = \prod_{i \in G_b} (x + ID_i)$. \mathcal{C} gives $P(G_b)$ to \mathcal{A} .

Guess: \mathcal{A} outputs a guess $b' \in \{0, 1\}$. If $b = b'$, \mathcal{A} wins.

\mathcal{A} must guess the group information from the group token. The group information is contained in $F(\alpha)$ of w_1 and w_2 . But $F(\alpha)$ is hidden by a random integer r . If \mathcal{A} can predict r from g^α and $\hat{w}_1 = g^{-\alpha r}$, then \mathcal{A} can generate $P(G_0)$, and $P(G_1)$, and compare them with $P(G_b)$ because G_0, G_1 are selected by \mathcal{A} . But predicting r from g^α and $\hat{w}_1 = g^{-\alpha r}$ is same as solving the DL problem. Therefore the group privacy is guaranteed if the DL assumption holds. \square

A.2 Proof of Theorem 2

Proof. Let a PPT adversary \mathcal{A} breaks the maximum number of accountability of our RR-BED scheme. The security game between a challenger \mathcal{C} and the adversary \mathcal{A} is executed as follows:

Setup: The challenger \mathcal{C} randomly chooses $\alpha, \beta \in \mathbb{Z}_p$ and $h \in \mathbb{G}$. It generates $\text{PP} = (\mathbb{B}, h, h^\alpha, \dots, h^{\alpha^N}, g, g^\alpha, \dots, g^{\alpha^N}, g^{\alpha\beta}, \dots, g^{\alpha^{N+1}\beta}, \Omega = e(g, g), \Omega_1 = e(g, h))$ and $\text{MK} = (\alpha, \beta)$. It keeps MK and gives PP to \mathcal{A} .

Challenge: \mathcal{C} chooses a threshold value $v \leq N$ and sends the value to \mathcal{A} .

Guess: \mathcal{A} chooses G^* , where $|G^*| = v' > v$, and generates a group token

$$\begin{aligned} P(G^*) &= (w_1, w_2, w_3, \hat{w}_1, \dots, \hat{w}_{k+1}, w_M) \\ &= (g^{\alpha\beta F(\alpha)r}, \Omega^{-r}, g^{-\alpha r}, g^{-\alpha^i r}, \dots, g^{-\alpha^{k+1}r}, \Omega_1^r). \end{aligned}$$

where k is a maximum revocation number and $F(x) = \prod_{i \in G^*} (x + ID_i)$. \mathcal{A} sends $(P(G^*), G^*)$ to \mathcal{C} . If $\text{RR-BED.Verify}(P(G^*), v, \text{PP}) = 1$, then \mathcal{A} wins.

$\text{RR-BED.Verify}(P(G^*), v, \text{PP}) = 1$ indicates that \mathcal{A} can generate a valid group token. So \mathcal{A} can compute $g^{\alpha^{N+2}}, \dots, g^{\alpha^{N+v'-v+1}}$ and the $v' (> v)$ degree polynomial $F(x) = \prod_{i \in G^*} (x + ID_i)$. Hence, breaking the maximum number of accountability is the same as solving the $(N + i)$ -DHE ($2 \leq i \leq v - v' + 1$) problem. \square





References

1. Attrapadung, N., Imai, H.: Conjunctive broadcast and attribute-based encryption. In: Shacham, H., Waters, B. (eds.) Pairing 2009. LNCS, vol. 5671, pp. 248–265. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03298-1_16
2. Acharya, K., Dutta, R.: Secure and efficient construction of broadcast encryption with dealership. In: Chen, L., Han, J. (eds.) ProvSec 2016. LNCS, vol. 10005, pp. 277–295. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47422-9_16
3. Acharya, K., Dutta, R.: Adaptively secure broadcast encryption with dealership. In: Hong, S., Park, J.H. (eds.) ICISC 2016. LNCS, vol. 10157, pp. 161–177. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-53177-9_8
4. Acharya, K., Dutta, R.: Adaptively secure recipient revocable broadcast encryption with constant size ciphertext. <https://eprint.iacr.org/2017/059.pdf>
5. Barth, A., Boneh, D., Waters, B.: Privacy in encrypted content distribution using private broadcast encryption. In: Di Crescenzo, G., Rubin, A. (eds.) FC 2006. LNCS, vol. 4107, pp. 52–64. Springer, Heidelberg (2006). https://doi.org/10.1007/11889663_4
6. Boneh, D., Waters, B.: A fully collusion resistant broadcast, trace, and revoke system. In: Proceedings of ACM Computer and Communications Security 2006, pp. 211–220 (2006)
7. Camacho, P.: Fair exchange of short signatures without trusted third party. In: Dawson, E. (ed.) CT-RSA 2013. LNCS, vol. 7779, pp. 34–49. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36095-4_3
8. 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). https://doi.org/10.1007/978-3-540-76900-2_12
9. Fiat, A., Naor, M.: Broadcast encryption. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 480–491. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48329-2_40
10. Furukawa, J., Attrapadung, N.: Fully collusion resistant black-box traitor revocable broadcast encryption with short private keys. In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 496–508. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-73420-8_44
11. Gritti, C., Susilo, W., Plantard, T., Liang, K., Wong, D.S.: Broadcast encryption with dealership. *Int. J. Inf. Secur.* **15**(3), 271–283 (2016)
12. Naor, D., Naor, M., Lotspiech, J.: Revocation and tracing schemes for stateless receivers. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 41–62. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_3
13. Phuong, T.V.X., Yang, G., Susilo, W., Chen, X.: Attribute based broadcast encryption with short ciphertext and decryption key. In: Pernul, G., Ryan, P.Y.A., Weippl, E. (eds.) ESORICS 2015. LNCS, vol. 9327, pp. 252–269. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24177-7_13
14. Ren, Y., Wang, S., Zhang, X.: Non-interactive dynamic identity-based broadcast encryption without random oracles. In: Chim, T.W., Yuen, T.H. (eds.) ICICS 2012. LNCS, vol. 7618, pp. 479–487. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34129-8_47
15. Susilo, W., Chen, R., Guo, F., Yang, G., Mu, Y., Chow, Y.: Recipient revocable identity-based broadcast encryption. In: ASIA CCS 2016, pp. 201–210 (2016)

Elliptic Curve



Solving 114-Bit ECDLP for a Barreto-Naehrig Curve

Takuya Kusaka¹(✉), Sho Joichi¹, Ken Ikuta¹, Md. Al-Amin Khandaker¹,
Yasuyuki Nogami¹, Satoshi Uehara², Nariyoshi Yamai³,
and Sylvain Duquesne⁴

¹ Graduate School of Natural Science and Technology, Okayama University,
Okayama 700-8530, Japan

{kusaka-t,yasuyuki.nogami}@okayama-u.ac.jp,
{sho.joichi,pwu03iut,khandaker}@s.okayama-u.ac.jp

² Department of Information and Media Engineering, The University of Kitakyushu,
Fukuoka 808-0135, Japan

uehara@kitakyu-u.ac.jp

³ Institute of Engineering, Tokyo University of Agriculture and Technology,
Tokyo 184-8588, Japan

nyamai@cc.tuat.ac.jp

⁴ Univ Rennes, CNRS, IRMAR - UMR 6625, 35000 Rennes, France
sylvain.duquesne@univ-rennes1.fr

Abstract. The security of cryptographic protocols which are based on elliptic curve cryptography relies on the intractability of elliptic curve discrete logarithm problem (ECDLP). In this paper, the authors describe techniques applied to solve 114-bit ECDLP in Barreto-Naehrig (BN) curve defined over the odd characteristic field. Unlike generic elliptic curves, BN curve holds an especial interest since it is well studied in pairing-based cryptography. Till the date of our knowledge, the previous record for solving ECDLP in a prime field was 112-bit by Bos et al. in Certicom curve ‘secp112r1’. This work sets a new record by solving 114-bit prime field ECDLP of BN curve using Pollard’s rho method. The authors utilized sextic twist property of the BN curve to efficiently carry out the random walk of Pollard’s rho method. The parallel implementation of the rho method by adopting a client-server model, using 2000 CPU cores took about 6 months to solve the ECDLP.

Keywords: ECDLP · Barreto-Naehrig curve · Pollard’s rho method

1 Introduction

The complexity of solving a difficult mathematical problem such as discrete logarithm problem (DLP) and elliptic curve discrete logarithm problem (ECDLP) in a practical amount of time determines the security level of many popular public key cryptosystems. The mathematical estimation of such complexity is common

in literature [10]. However, actual experiment is also bearing same importance, which is the focus of this work.

Pairing-based cryptography became popular as a form of public key cryptography (PKC) by the works of [3, 4, 17, 18]. Typically, pairing is defined as a bilinear map of two additive cyclic sub-groups \mathbb{G}_1 and \mathbb{G}_2 of prime order r to the same order multiplicative group \mathbb{G}_3 . In practice, \mathbb{G}_1 and \mathbb{G}_2 are defined over a certain pairing-friendly curve of embedding degree k and \mathbb{G}_3 is defined in extension field \mathbb{F}_{p^k} . This paper considers Barreto-Naehrig curve [1], one of the widely used curves for pairing-based cryptographic applications. Let the curve be E , defined over the finite extension field \mathbb{F}_{p^k} , where embedding degree k is the smallest positive integer such that $r \mid (p^k - 1)$. The set of rational points $E(\mathbb{F}_p)$ is defined over the prime field \mathbb{F}_p , together with the *point at infinity* \mathcal{O} forms a commutative group. The curve's order r is a large prime number such that $r \mid \#E(\mathbb{F}_p)$, where $\#E(\mathbb{F}_p)$ denotes the total number of rational points. The bilinearity property of pairing is e given as $e(aP, bQ) = e(P, Q)^{ab}$ for every rational point $P \in \mathbb{G}_1, Q \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}$. In the case of BN curve, $\mathbb{G}_1 = E(\mathbb{F}_p)$ and $\mathbb{G}_2 \subset E(\mathbb{F}_{p^{12}})$. The ECDLP in $\mathbb{G}_1 = E(\mathbb{F}_p)$ with two arbitrary rational point P, R is finding an integer s ($0 < s \approx r$) such that $[s]P = R$ in E , provided that such s exists for P, R . The security of pairing-based cryptosystems depends on the difficulty of

- solving ECDLP in \mathbb{G}_1 and \mathbb{G}_2 ,
- solving DLP of the multiplicative group \mathbb{G}_3 .
- and the difficulty of pairing inversion, i.e. $\mathbb{G}_1 \times \mathbb{G}_2 \leftarrow \mathbb{G}_3$.

This paper focuses on solving ECDLP in $\mathbb{G}_1 = E(\mathbb{F}_p)$ of BN curve.

The Pollard's rho algorithm [16] can solve the ECDLP in $\sqrt{\pi r}/2$ steps. Later Gallant et al. [8] improved the time complexity of the Pollard's rho method by $\sqrt{2}$ factors, that took $\sqrt{\pi r}/2$ steps. Van Oorschot and Wiener [20] proposed a distributed version of Pollard's rho algorithm that can be parallelized on n CPUs taking $\sqrt{\pi r}/2n$ steps giving n -fold speed up. This paper utilizes the above idea of parallelized rho method together with an efficient implementation of elliptic curve arithmetic using Montgomery reduction [13] and Montgomery trick [14].

In the experimental implementation, a client-server model consisting 2000 processor cores (each core acts as a single client for generating random rational points) is utilized. The generated random points with special feature also called the distinguished point, is sent to server to minimize collision detection cost. The server checks collision detection and uses MySQL database to store received rational points. The inversion in elliptic curve addition and doubling steps are efficiently carried out by applying Montgomery tricks while creating 95 random walks for each inversion in each thread. In addition, the sextic twist property of BN curve is utilized to apply skew Frobenius map on the twisted curve. The skew Frobenius map generates 6 associated rational points, which are treated as a single rational point in a random walk. The distinguished points technique is applied with the condition of 29 trailing zeros of x -coordinate to send the filtered point to the server. The experiment took about 6 months to solve the 114-bit ECDLP in BN curve defined over the prime field.

Related works

Several works have been done both from the algorithmic perspective and actual attack implementation. However, most of the works are focused on solving ECDLP in curves defined over binary field [2, 21]. A very few works have done on the actual attack for solving ECDLP in curves defined over \mathbb{F}_p . In 2016, Kajitani et al. [9] solved 70-bit prime field ECDLP in less than 24 min by using web-based volunteer computing. The authors of this paper have been influenced by Miyoshi et al. [12] work, where 94-bit ECDLP on was solved in 28 h by parallelizing 71 computers. In 2012, Bos et al. [5] set a record by solving 112-bit ECDLP in prime field using about 200 Sony PlayStation 3s for about 6 months. Before that in 2002, the 109-bit ECDLP of Certicom ECCp-109 curve defined over \mathbb{F}_p was solved by Chris Monico [6] in 549 days of calendar time. Therefore, this work sets a new record for solving ECDLP on BN curve defined over 114-bit prime field.

2 Preliminaries

2.1 Elliptic Curve and ECDLP

Let p be a prime number and \mathbb{F}_p be a prime field. An elliptic curve, generally represented by affine coordinates over \mathbb{F}_p is defined as,

$$E(\mathbb{F}_p) : y^2 = x^3 + ax + b, \quad a, b \in \mathbb{F}_p. \tag{1}$$

A pair of coordinates x and y that satisfy Eq. (1) are known as rational points on the curve. Let $\#E(\mathbb{F}_p)$ be the set of all rational points on the curve defined over \mathbb{F}_p including the point at infinity denoted by \mathcal{O} .

Elliptic curve addition (ECA) between rational points $Q_1(x_1, y_1)$ and $Q_2(x_2, y_2)$ is $Q_1 + Q_2 = Q_3(x_3, y_3)$, defined as follows:

$$\lambda = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1}, & \text{if } Q_1 \neq Q_2 \text{ and } x_1 \neq x_2, \\ \frac{3x_1^2 + a}{2y_1}, & \text{else if } Q_1 = Q_2 \text{ and } y_1 \neq 0, \\ \emptyset, & \text{otherwise,} \end{cases} \tag{2}$$

$$(x_3, y_3) = \begin{cases} (\lambda^2 - x_1 - x_2, (x_1 - x_3)\lambda - y_1), & \text{if } \lambda \neq \emptyset, \\ \mathcal{O}, & \text{otherwise.} \end{cases} \tag{3}$$

where Q_3 is also a rational points of elliptic curve $E(\mathbb{F}_p)$ and λ is the tangent between the points. If $Q_1 = Q_2$ then $Q_1 + Q_2 = 2Q_1$, which is known as elliptic curve doubling (ECD). ECDLP is the problem that calculates the scalar s only by using rational points P and Q in $E(\mathbb{F}_p)$ such that $Q = [s]P$, where $[s]P$ is

$$[s]P = \underbrace{P + P + \dots + P}_{s-1 \text{ times addition of } P}. \tag{4}$$

2.2 BN Curve

BN curve is a class of non super-singular pairing friendly elliptic curve of embedding degree 12, defined over extension field \mathbb{F}_q ($q = p^{12}$) is given by

$$E : y^2 = x^3 + b, \quad (b \neq 0 \in \mathbb{F}_q \text{ and } x, y \in \mathbb{F}_q). \tag{5}$$

Other parameter settings are given by

$$p = 36\chi^4 - 36\chi^3 + 24\chi^2 - 6\chi + 1, \tag{6}$$

$$r = 36\chi^4 - 36\chi^3 + 18\chi^2 - 6\chi + 1, \tag{7}$$

where χ is a certain integer and p is the characteristic of \mathbb{F}_p . Let $\#E(\mathbb{F}_q)$ be the set of all rational points on the curve defined over \mathbb{F}_q including the point at infinity denoted by \mathcal{O} .

2.3 Pollard’s Rho Method

Pollard’s rho method is known as an efficient technique for solving an ECDLP. We designed and implemented a parallelized rho method for solving a 114-bit ECDLP with thousands of CPU cores. The proposed method consists of three steps. The first step generates a set of n different random rational points denoted by L . For an integer i where $1 \leq i \leq n$ and two random scalars α_i, β_i ($\in \mathbb{F}_q$), let W_i denote i -th random walk seed to be used in random walks where

$$W_i \triangleq [\alpha_i]P + [\beta_i]Q, \tag{8}$$

$$L \triangleq \{W_i | 1 \leq i \leq n\}. \tag{9}$$

Let m be the number of branches of the parallel random walks. The second step generates a set of m different random starting rational points in the m parallel random walk branches, denoted by U . For two integers i and j where $1 \leq i \leq m$ and $0 \leq j$, let $R_{i,j}$ denote the j -th rational point which is generated with random scalars $\alpha_{i,j}$ and $\beta_{i,j}$ by the i -th random walk branch. For an integer i where $1 \leq i \leq m$, since $R_{i,0}$ denotes the starting point of the i -th random walk branch where

$$R_{i,0} = [\alpha_{i,0}]P + [\beta_{i,0}]Q, \tag{10}$$

$R_{i,0}$ is randomly generated and,

$$U \triangleq \{R_{i,0} | 1 \leq i \leq m\}. \tag{11}$$

The third step performs m random works in parallel by using L and U . For a rational point R , let $\eta(R)$ be a function which gives an unique index of a rational point in L . For two integers i and j where $1 \leq i \leq m$ and $1 \leq j$, let the rational points $R_{i,j}$ which is the j -th rational point in the i -th random walk branch, be calculated by the following,

$$R_{i,j} = R_{i,j-1} + W_{\eta(R_{i,j-1})}, \quad (12)$$

$$= [\alpha_{i,j-1} + \alpha_{\eta(R_{i,j-1})}]P + [\beta_{i,j-1} + \beta_{\eta(R_{i,j-1})}]Q, \quad (13)$$

$$= [\alpha_{i,j}]P + [\beta_{i,j}]Q. \quad (14)$$

Let H be the set of all generated rational points in the rho method. Suppose the i -th random walk branch generate a rational point $R_{i,j} \in H$ at the j -th iteration step and we have a rational point $R_{i,j} = R_{i',j'}$ where $\alpha_{i,j} \neq \alpha_{i',j'}$ and $\beta_{i,j} \neq \beta_{i',j'}$, the case is called a collision and the ECDLP is solved by a simultaneous equation.

The algorithm of parallelized rho method is given as Algorithm 1. An example of parallelized rho method with 12-bit ECDLP is shown as follow Fig. 1. In this paper, L and U are generated as preparation steps, and perform random walk at every random walk branches by using L and U . Each attacking clients run the algorithm. Therefore, the total number of random walk branches is m times the number of attacking clients. It is said that a collision would occur when $\sqrt{\pi r/2}$ points are generated on average according to the birthday paradox.

Algorithm 1. Parallelized rho Method

Input: $P, Q(= [s]P) \in E(\mathbb{F}_p)(0 \leq s < r)$

Output: s

```

1  for  $i = 1$  to  $n$  do
2  |  $\alpha_i, \beta_i$  are random elements ( $0 \leq \alpha_i, \beta_i < r$ ),
3  |  $W_i \leftarrow [\alpha_i]P + [\beta_i]Q$ .
4  |  $H \leftarrow \phi$ .
5  for  $i = 1$  to  $m$  do
6  |  $\alpha_{i,0}, \beta_{i,0}$  are random elements ( $0 \leq \alpha_{i,0}, \beta_{i,0} < r$ ),
7  |  $R_{i,0} \leftarrow [\alpha_{i,0}]P + [\beta_{i,0}]Q$ .
8  |  $H \leftarrow H \cup \{R_{i,0}\}$ .
9  for  $j = 1$  to  $r - 1$  do
10 |   for  $i = 1$  to  $m$  do
11 |     |  $l \leftarrow \eta(R_{i,j-1})$ .
12 |     |  $R_{i,j} \leftarrow R_{i,j-1} + W_l, \alpha_{i,j} \leftarrow \alpha_{i,j-1} + \alpha_l, \beta_{i,j} \leftarrow \beta_{i,j-1} + \beta_l$ .
13 |     | if  $R_{i,j} = R_{i',j'} (R_{i',j'} \in H, \alpha_{i,j} \neq \alpha_{i',j'}, \beta_{i,j} \neq \beta_{i',j'})$  then
14 |     |   | go to line 15.
14 |     | else
14 |     |   |  $H \leftarrow H \cup \{R_{i,j}\}$ .
15  $s \leftarrow -\frac{(\alpha_{i,j} - \alpha_{i',j'})}{(\beta_{i,j} - \beta_{i',j'})} \pmod{r}$ .
```

3 Techniques for Accelerating Random Walk

In the attack, Montgomery reduction [13] is used for efficient modular arithmetics over \mathbb{F}_p . Montgomery trick [14] is also used for reducing the number of inversions over \mathbb{F}_p by parallelizing many random walks on each client. Then, this paper

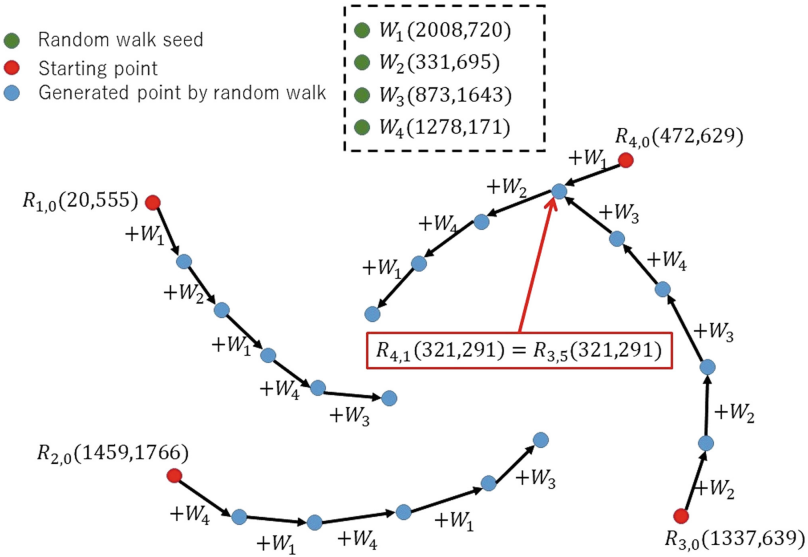


Fig. 1. Parallelized rho method with 12bit ECDLP, $n = 4$ and 4 branches

attacks 114-bit ECDLP in \mathbb{G}_1 on BN curve to which a grouping technique is applied based on the sextic twist [15]. This section concentrates on introducing the grouping technique with skew Frobenius mapping defined in \mathbb{G}_1 .

3.1 Groups of Rational Points for Ate Pairing on BN Curve

Let us remember the additive groups of rational points for Ate pairing e defined as follows, where $E(\mathbb{F}_{p^k})[r]$, Ker , and π denote the set of rational points of order r , kernel of homomorphism, and Frobenius mapping [7], respectively.

$$\mathbb{G}_1 = E(\mathbb{F}_{p^k})[r] \cap \text{Ker}(\pi - [1]), \tag{15}$$

$$\mathbb{G}_2 = E(\mathbb{F}_{p^k})[r] \cap \text{Ker}(\pi - [p]), \tag{16}$$

$$e(\cdot, \cdot) : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_3 = \mathbb{F}_{p^k}^* / (\mathbb{F}_{p^k}^*)^r. \tag{17}$$

In the case of BN curve, the embedding degree k is equal to 12 and the above \mathbb{G}_1 is equal to $E(\mathbb{F}_p)$. Based on these definitions, the next section introduces a grouping technique with sextic twist and skew Frobenius mapping.

3.2 Sextic Twist and Skew-Frobenius Mapping

Basically, in order to improve Ate or optimal-ate pairing with BN curve, the sextic-twist technique is available. However, it also contributes to attacking the

ECDLP on BN curve. Since the embedding degree of BN curve is 12 and BN curve is written as Eq. (5), sextic-twisted curve E' is given by

$$E' : y^2 = x^3 + bv^{-1}, \quad (18)$$

where v is a cubic and quadratic non residue in \mathbb{F}_{p^2} . In this case, we have the following isomorphism [19].

$$\mathbb{G}'_1 = E'(\mathbb{F}_{p^{12}})[r] \cap \text{Ker}(\pi^2 - [p^2]), \quad (19)$$

$$\begin{aligned} \psi_6 : (x, y) \in \mathbb{G}_1 \\ \longmapsto (v^{1/3}x, v^{1/2}y) \in \mathbb{G}'_1. \end{aligned} \quad (20)$$

\mathbb{G}'_1 has the following automorphism mapping $\tilde{\pi}$, where Q is a rational point in \mathbb{G}_1 . It is called skew Frobenius mapping.

$$\begin{aligned} \tilde{\pi}(Q) &= \psi_6^{-1}(\pi^2(\psi_6(Q))) \\ &= (v^{\frac{p^2-1}{3}}x, v^{\frac{p^2-1}{2}}y). \end{aligned} \quad (21)$$

In this case, $\tilde{\pi}^6(Q) = \tilde{\pi}$, $v^{(p^2-1)/3}$ becomes a primitive cubic root of unity ϵ in \mathbb{F}_p , and $v^{(p^2-1)/2}$ becomes $p-1$. Thus, in what follows, the skew Frobenius $\tilde{\pi}$ map in this case is denoted by $\tilde{\pi}_6$ because it is periodic of period 6. Then, $\tilde{\pi}_6$ enables an efficient grouping in the rho method as introduced in the next section.

3.3 A Grouping of Rational Points in \mathbb{G}_1 on BN Curve

Let us consider a rational point $T_i \in \mathbb{G}_1$ as generated in the random walk process. Then, based on the skew Frobenius mapping $\tilde{\pi}$, the following six rational points in \mathbb{G}_1 are easily obtained.

$$T_i = (x_i, y_i), \quad (22a)$$

$$\tilde{\pi}_6(T_i) = (\epsilon x_i, y_i), \quad (22b)$$

$$\tilde{\pi}_6^2(T_i) = (\epsilon^2 x_i, y_i), \quad (22c)$$

$$\tilde{\pi}_6^3(T_i) = (x_i, -y_i), \quad (22d)$$

$$\tilde{\pi}_6^4(T_i) = (\epsilon x_i, -y_i), \quad (22e)$$

$$\tilde{\pi}_6^5(T_i) = (\epsilon^2 x_i, -y_i). \quad (22f)$$

Then, a certain representative point among the six points is systematically and efficiently determined, which enables the following efficient grouping attack. For a rational point $R \in \mathbb{G}_1$, let $\text{Rep}(R)$ denote a function which uniquely gives the representative in the group of six rational points given by the skew Frobenius map $\tilde{\pi}_6$.

Let us suppose that a collision is detected as

$$\text{Rep}(T_i) = \text{Rep}(T_j), \quad (23)$$

$$\text{Rep}(T_i) = [\alpha_i]P + [\beta_i]Q, \quad (24)$$

$$\text{Rep}(T_j) = [\alpha_j]P + [\beta_j]Q, \quad (25)$$

where $(\alpha_i, \beta_i) \neq (\alpha_j, \beta_j)$. Then, since $\tilde{\pi}_6^t(T_j) = [p^{2t}]T_j$, it means that the following relation holds:

$$\begin{aligned} T_i &= \tilde{\pi}_6^t(T_j), \\ T_i &= [p^{2t}]T_j, \\ [\alpha_i]P + [\beta_i]Q &= [p^{2t}]([\alpha_j]P + [\beta_j]Q), \\ \alpha_i + \beta_i \cdot s &\equiv p^{2t} \cdot \alpha_j + p^{2t} \cdot \beta_j \cdot s \pmod{r}, \\ s &\equiv -\frac{(\alpha_i - p^{2t} \cdot \alpha_j)}{(\beta_i - p^{2t} \cdot \beta_j)} \pmod{r}. \end{aligned} \tag{26}$$

where $0 \leq t < 6$. Therefore, since the skew Frobenius mapping is efficiently carried out as previously introduced, this grouping technique enables to reduce the average number for detecting a collision from $\sqrt{\pi r/2}$ to $\sqrt{\pi r/12}$. Algorithm 2 accommodates the above procedure of obtaining representative point among the 6 associated rational points. The step 7, 12, 13 and 16 actually differs from Algorithm 1. In this paper, the last 16 generated scalars α are saved in order to avoid fruitless cycles. If a new generated scalar α is same as one of the previous scalars, $\eta(R)$ in Sect. 2.3 is incremented by 1.

4 Implementation

In this experiment, the authors employed about 2000 heterogeneous Intel64 CPU cores to attack the ECDLP by the rho method. In the parallel rho method, each random walk branches are executed on distributed computer resources in parallel. It is necessary to aggregate all the generated random rational points to a single computer in order to check the collision. Therefore, this paper employs a typical client-server model. In this context, all clients generate random rational points in parallel and only the distinguished points are sent to the server. The collision detection is done on the server.

4.1 Basic Integer Operations and Algebra

To handle the 114-bit integers efficiently, the authors employed 128-bit integer type on Intel64 CPUs. Since the target is a 114-bit ECDLP, an addition and a subtraction between two 114-bit integers never causes an overflow or an underflow. In contrast, since a result of a multiplication between two 114-bit integers easily exceeds 128-bit integer, the authors implemented big number arithmetic to handle this case.

In this attack, since the most of multiplications are multiplication modulo prime p or r ; Montgomery reduction is implemented by using the 128-bit multiplication. A calculation of a multiplicative inverse is implemented by using the well-known extended Euclid's algorithm.

Since the majority of the computational cost of an ECA or an ECD is the cost of the inversion, well-known Montgomery trick is also employed. Since several inversions must be aggregated to utilize Montgomery trick, the authors evaluated

Algorithm 2. Customized parallelized rho method with representative point technique

Input: $P, Q(= [s]P) \in E(\mathbb{F}_p)(0 \leq s < r)$

Output: s

```

1 for  $i = 1$  to  $n$  do
2    $\alpha_i, \beta_i$  are random elements ( $0 \leq \alpha_i, \beta_i < r$ ),
3    $W_i \leftarrow [\alpha_i]P + [\beta_i]Q$ .
4    $H \leftarrow \phi$ .
5 for  $i = 1$  to  $m$  do
6    $\alpha_{i,0}, \beta_{i,0}$  are random elements ( $0 \leq \alpha_{i,0}, \beta_{i,0} < r$ ),
7    $R_{i,0} \leftarrow \text{Rep}([\alpha_{i,0}]P + [\beta_{i,0}]Q)$ .
8    $H \leftarrow H \cup \{R_{i,0}\}$ .
9 for  $j = 1$  to  $r - 1$  do
10  for  $i = 1$  to  $m$  do
11     $l \leftarrow \eta(R_{i,j-1})$ .
12    if  $\alpha_{i,j-1} = \alpha_{i,c} (j - 18 < c < j - 1)$  then
13       $l \leftarrow l + +$ .
14     $R_{i,j} \leftarrow \text{Rep}(R_{i,j-1} + W_i)$ .  $\alpha_{i,j} \leftarrow \alpha_{i,j-1} + \alpha_l, \beta_{i,j} \leftarrow \beta_{i,j-1} + \beta_l$ .
15    if  $R_{i,j} = R_{i',j'} (R_{i',j'} \in H, \alpha_{i,j} \neq \alpha_{i',j'}, \beta_{i,j} \neq \beta_{i',j'})$  then
16      go to line 16.
17    else
18       $H \leftarrow H \cup \{R_{i,j}\}$ .
19  $s \leftarrow -\frac{(\alpha_{i,j} - p^{2t} \cdot \alpha_{i',j'})}{(\beta_{i,j} - p^{2t} \cdot \beta_{i',j'})} \pmod{r}$ .

```

the performance of our implementation of the inversion and choose 95 number of aggregation for Montgomery trick. Therefore, 95 random walk branches are synchronized and handled in a single thread in the implementation. The number of the threads in a client computer m is chosen as the number of real CPU cores or the number of Hyper Threading of the computer. For this case, the computational cost of a single step in rho method having Montgomery trick with a grouping (Sect. 3.3) and without a grouping are shown in Table 1. The inversion takes 305 additions and a single multiplication on average.

Table 1. The computational cost of a single step in Rho method with Montgomery trick

Operations (mod p)	With grouping	Without grouping
Addition	1073	760
Multiplication	691	472
Squaring	95	95
Inversion	1	1

4.2 Distinguished Point Method

The collision detection is a check whether a rational point is in the set of all previously stored rational points or not. If the all generated rational points are sent to the server from the all clients, the number of rational points which must be stored in the server easily exceeds the capacity of the memory space of the server. For example, when solving 114-bit ECDLP, about $\sqrt{\pi \times 2^{114}/12} \approx 7.4 \times 10^{16}$ points need to be stored to the server. If the size of the data of a rational point is 50[Byte], 3.7×10^6 [TB] storage space is required. Therefore, the authors choose distinguished point method to reduce the number of transmitted and stored rational points. In this attack, a distinguished point is a rational point where it's x -coordinate is divisible by an integer θ of the form of power of two. Here θ is called parameter of the distinguished point method. The computational cost to distinguish the point is a logical AND operation and comparison to numerical zero. The number of rational points sent to the server is reduced to $1/\theta$. The number of stored rational points are also reduced to $1/\theta$ which significantly reduce the cost of collision detection on the server. This reduction causes an overhead by generating extra rational points in the random walk branches.

If there is a pair of two rational points $(R_{i,j}, R_{i',j'})$ where $R_{i,j} = R_{i',j'}$ for positive integer h , $(\alpha_{i,j}, \beta_{i,j}) \neq (\alpha_{i',j'}, \beta_{i',j'})$, the following holds.

$$\eta(R_{i,j}) = \eta(R_{i',j'}), \quad (27)$$

$$R_{i,j+1} = R_{i',j'+1}, \quad (28)$$

$$R_{i,j+h} = R_{i',j'+h}. \quad (29)$$

In addition, if $\theta/2$ rational points are generated in a random walk branch, a distinguished point can be generated in the random walk branch on average. Therefore, the overhead is at most $\theta/2$ iterations in the pair of random walk branches, which means the distinguished method does not significantly increase the number of iteration steps before the rho method ends.

4.3 Aggregation on Generated Distinguished Points

By using the distinguished point method, the frequency to transmit the distinguished points can be significantly reduced. However, to reduce the frequency to transmit the IP datagrams, which includes the distinguished points, we employ a method to aggregate the distinguished points.

Each processor acts as a single client for generating random rational points in the experiment. The frequency to transmit the generated rational points to the server depends on the parameter of the distinguished method θ . A client stores a set of distinguished points. If the number of stored points becomes large enough, they are sent to the server at once. The number of the aggregated distinguished points is set 128 in this work. Therefore, the frequency to transmit the IP datagrams is reduced to $1/128$ in the implemented system.

4.4 Collision Detection at Server

The server stores information about coordinate of the rational point and scalars. For example, when the server receives $R(x,y) = [\alpha]P + [\beta]Q$, it stores x and α , β . When a new rational point is received, the server compares the information with stored rational points. If any corresponding point is found in the server, a scalar is calculated and the attack is ended.

By using Sect. 4.2, the number of stored rational points is reduced. However, it is difficult to store all generated points on the memory of the server. Therefore, this study stores rational points on the MySQL databases.

4.5 Computer Resources

This attack employed several types of computers for both the server and the clients. Table 2 shows the major clients and the specification of the computers. The attack consists of 3 phases. The first phase was 51 days attack with Client 1 and 2. The second phase was 47 days attack with same clients. The final phase was 81 days attack with Client 3.

Table 2. Computer resources

Client 1	The number of computers	150
	OS	Mac OSX 10.7.5 (64-bit)
	CPU	Intel Core i5 (2.50 GHz)
Client 2	The number of computers	716
	OS	Windows7 Professional (64-bit)
	CPU	Intel Core 2 Duo E7600 (3.06 GHz)
Client 3	The number of computers	162
	OS	FreeBSD 11 (64-bit)
	CPU	Dual Intel Xeon X5670 (2.90 GHz)
Server	OS	CentOS 6.8 (64-bit)
	CPU	Intel Core i5 (3.40 GHz)
	Database	MySQL ver. 5.1.73

5 Result and Analysis

The parameter settings of BN curve for this attack is as follows:

$$\begin{aligned}\chi &= 135000271, \\ b &= 3, \\ r &= 11957518425389075145185553763727233, \\ p &= 11957518425389075254535992784167879.\end{aligned}$$

The rational points for this ECDLP attack were randomly chosen by Mersenne Twister (MT) pseudorandom number generator [11] as follows:

$$\begin{aligned} P &= (2555802502070605019799774084777870, \\ &\quad 2766213229730430765452066521605006), \\ Q &= (8729198974879999392476239739358779, \\ &\quad 2797493486736137111251588290298576). \end{aligned}$$

The x coordinate of rational point P is generated randomly where x satisfies Eq. (5) for the first time. The secret scalar s was incremented by 1 after generating randomly. The random seed of MT pseudorandom number generator is 5489. The Q is obtained as $[s]P = Q$. This s is only used to obtain Q in initial step but no longer used during the attack stage. After the 81 days, the attack introduced in Sect. 4.5 with 136887663 rational points stored in the server; ended with the following two pairs of random scalars of a collision:

$$\begin{aligned} (\alpha, \beta) &= (10978171553023857380293303028367032, \\ &\quad 7667182665169261066594516279554751), \\ (\alpha', \beta') &= (49568355245740117450745472411632, \\ &\quad 1446344014944960931462013632028773), \end{aligned}$$

where $[\alpha]P + [\beta]Q = [\alpha']P + [\beta']Q$. The secret scalar s was calculated by

$$\begin{aligned} s &= 10928603197778117262842557555955400 / \\ &\quad 5736679775164775010053051116201255. \end{aligned}$$

The results proved that this attack certainly succeeded to solve the 114-bit ECDLP. Note that 5.5 GB of memory was consumed to store the information of generated rational points.

To save the memory space of the collision detection server and to reduce the network traffic, the authors disposed the information on the actual number of all the generated points, the authors give an estimation of the number. Since the authors chose $\theta = 2^{29}$ as the parameter of the distinguished points, we can estimate the total number of generated rational points is $136887663 \times 2^{29} = 73491004476358656$. Since $r = 11957518425389075145185553763727233$, the result might be one of the typical average cases. However, the result of the authors attack is little bit better than the average number of rational points where a simple collision attack stops.

6 Conclusion

In this paper, the authors implemented a parallel rho attack for a 114-bit BN curve and solved an ECDLP for the curve. The attack system employed well-known Montgomery reduction, Montgomery trick and the extended Euclid' algorithm by using specially implemented integer operations over 128-bit integer type

on Intel64 CPUs. The authors also employed a grouping of rational points in \mathbb{G}_1 on BN curve which significantly reduced the number of useless collision detections in the attack. The results of the experiment indicates that the 114-bit ECDLP for BN curves can be solved in 81 days with 2000 cores of Intel Xeon X5670 (2.90 GHz) CPU on average.

References

1. Barreto, P.S.L.M., Naehrig, M.: Pairing-friendly elliptic curves of prime order. In: Preneel, B., Tavares, S. (eds.) SAC 2005. LNCS, vol. 3897, pp. 319–331. Springer, Heidelberg (2006). https://doi.org/10.1007/11693383_22
2. Bernstein, D.J., Engels, S., Lange, T., Niederhagen, R., Paar, C., Schwabe, P., Zimmermann, R.: Faster elliptic-curve discrete logarithms on FPGAs. Technical report, Cryptology eprint Archive, Report 2016/382 (2016)
3. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_30
4. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the weil pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_30
5. Bos, J.W., Kaihara, M.E., Kleinjung, T., Lenstra, A.K., Montgomery, P.L.: Solving a 112-bit prime elliptic curve discrete logarithm problem on game consoles using sloppy reduction. IJACT **2**(3), 212–228 (2012). <https://doi.org/10.1504/IJACT.2012.045590>
6. Certicom: the Certicom ECC challenge. <https://www.certicom.com/content/dam/certicom/images/pdfs/challenge-2009.pdf>. Accessed 10 Aug 2017
7. Cohen, H., Frey, G., Avanzi, R., Doche, C., Lange, T., Nguyen, K., Vercauteren, F.: Handbook of Elliptic and Hyperelliptic Curve Cryptography. CRC Press, Boca Raton (2005)
8. Gallant, R., Lambert, R., Vanstone, S.: Improving the parallelized pollard lambda search on anomalous binary curves. Math. Comput. Am. Math. Soc. **69**(232), 1699–1705 (2000)
9. Kajitani, S., Nogami, Y., Miyoshi, S., Austin, T., Al-Amin, K.M., Begum, N., Duquesne, S.: Web-based volunteer computing for solving the elliptic curve discrete logarithm problem. Int. J. Netw. Comput. **6**(2), 181–194 (2016)
10. Kim, T., Barbulescu, R.: Extended tower number field sieve: a new complexity for the medium prime case. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9814, pp. 543–571. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53018-4_20
11. Matsumoto, M., Nishimura, T.: Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Trans. Model. Comput. Simul. **8**(1), 3–30 (1998). <https://doi.org/10.1145/272991.272995>
12. Miyoshi, S., Nogami, Y., Kusaka, T., Yamai, N.: Solving 94-bit ECDLP with 70 computers in parallel. Int. J. Comput. Electr. Autom. Control Inf. Eng. **9**(8), 1966–1969 (2015)
13. Montgomery, P.: Modular multiplication without trial division. Math. Comput. **44**, 519–521 (1985)

14. Montgomery, P.L.: Speeding the pollard and elliptic curve methods of factorization. *Math. Comput.* **48**(177), 243–264 (1987)
15. Nogami, Y., Sakemi, Y., Okimoto, T., Nekado, K., Akane, M., Morikawa, Y.: Scalar multiplication using frobenius expansion over twisted elliptic curve for ate pairing based cryptography. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **92–A**(1), 182–189 (2009)
16. Pollard, J.M.: Monte carlo methods for index computation (mod p). *Math. Comput.* **32**(143), 918–924 (1978)
17. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) *EUROCRYPT 2005*. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_27
18. Sakai, R., Kasahara, M.: Id based cryptosystems with pairing on elliptic curve. *IACR Cryptology ePrint Archive* 2003, 54 (2003)
19. Sakemi, Y., Nogami, Y., Okeya, K., Kato, H., Morikawa, Y.: Skew Frobenius map and efficient scalar multiplication for pairing-based cryptography. In: Franklin, M.K., Hui, L.C.K., Wong, D.S. (eds.) *CANS 2008*. LNCS, vol. 5339, pp. 226–239. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-89641-8_16
20. Van Oorschot, P.C., Wiener, M.J.: Parallel collision search with cryptanalytic applications. *J. Cryptol.* **12**(1), 1–28 (1999)
21. Wenger, E., Wolfger, P.: Solving the discrete logarithm of a 113-bit Koblitz curve with an FPGA cluster. In: Joux, A., Youssef, A. (eds.) *SAC 2014*. LNCS, vol. 8781, pp. 363–379. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-13051-4_22



On the Computational Complexity of ECDLP for Elliptic Curves in Various Forms Using Index Calculus

Chen-Mou Cheng^(✉), Kenta Kodera, and Atsuko Miyaji

Graduate School of Engineering, Osaka University, Suita, Japan
{ccheng, kodera}@cy2sec.comm.eng.osaka-u.ac.jp,
miyaji@comm.eng.osaka-u.ac.jp

Abstract. The security of elliptic curve cryptography is closely related to the computational complexity of the elliptic curve discrete logarithm problem (ECDLP). Today, the best practical attacks against ECDLP are exponential-time, generic discrete logarithm algorithms such as Pollard's rho method. Recently, there is a line of research on index calculus for ECDLP started by Semaev, Gaudry, and Diem. Under certain heuristic assumptions, such algorithms could lead to subexponential attacks to ECDLP in some cases. In this paper, we investigate the computational complexity of ECDLP for elliptic curves in various forms—including Hessian, Montgomery, (twisted) Edwards, and Weierstrass using index calculus. The research question we would like to answer is: Using index calculus, is there any significant difference in the computational complexity of ECDLP for elliptic curves in various forms? We will provide some empirical evidence and insights showing an affirmative answer in this paper.

Keywords: Security evaluation · ECDLP · Index calculus
Summation polynomial · Point decomposition problem

1 Introduction

In recent years, elliptic curve cryptography is gaining momentum in deployment, as it can achieve the same level of security as RSA using much shorter keys and ciphertexts. The security of elliptic curve cryptography is closely related to the computational complexity of the elliptic curve discrete logarithm problem (ECDLP). Let p be a prime number and E , a nonsingular elliptic curve over \mathbb{F}_{p^n} , the finite field of p^n elements. That is, E is a plane algebraic curve defined by the equation $y^2 = x^3 + ax + b$ for $a, b \in \mathbb{F}_{p^n}$ such that $\Delta = -16(4a^3 + 27b^2) \neq 0$. Along with a point \mathcal{O} at infinity, the set of rational points $E(\mathbb{F}_{p^n})$ forms an abelian group with \mathcal{O} as the identity. Given $P \in E(\mathbb{F}_{p^n})$ and Q in the subgroup generated by P , ECDLP is the problem of finding an integer α such that $Q = \alpha P$.

Today, the best practical attacks against ECDLP are exponential-time, generic discrete logarithm algorithms such as Pollard's rho method [14]. However, recently there is a line of research on index calculus for ECDLP started

by Semaev, Gaudry, and Diem [5, 10, 15]. Under certain heuristic assumptions, such algorithms could lead to subexponential attacks to ECDLP in some cases [7, 11, 13]. The interested reader is referred to a survey paper by Galbraith and Gaudry for a more comprehensive and in-depth account of the recent development of ECDLP algorithms along various directions [8].

In this paper, we investigate the computational complexity of ECDLP for elliptic curves in various forms—including Hessian [16], Montgomery [12], (twisted) Edwards [3, 4], and Weierstrass using index calculus. Recently, elliptic curves of various forms such as Curve25519 [2] have been drawing a lot of attention in deployment, partly because some of them allow for fast implementation and security against timing-based side channel attacks. Furthermore, we can construct these curves not only over prime fields (such as the field of $2^{255} - 19$ elements as used in Curve25519) but also extension fields. In this paper, we will focus on curves over optimal extension fields (OEFs) [1]. An OEF is an extension field from a prime field \mathbb{F}_p with p close to $2^8, 2^{16}, 2^{32}, 2^{64}$, etc. Such primes fit nicely into the processor words of 8, 16, 32, or 64-bit microprocessors and hence are particularly suitable for software implementation, allowing for efficient utilization of fast integer arithmetics on modern microprocessors [1]. As we will see, our experimental results show quite significant difference in the computational complexity of ECDLP for elliptic curves in various forms over OEFs.

The rest of this paper is organized as follows. In Sect. 2, we will review the relevant literature, giving an high-level overview of attacking ECDLP using index calculus and the state of the art in this research direction. In Sect. 3, we will present how we can attack ECDLP using index calculus for elliptic curves in Montgomery and Hessian forms. In Sect. 4, we will experimentally compare its computational complexity for elliptic curves in various forms. Finally, we will conclude this paper by analyzing why ECDLP for elliptic curves in certain forms may be “easier” to attack using index calculus in Sect. 5.

2 Previous Works

2.1 Index Calculus for ECDLP

Let E be an elliptic curve defined over a finite field \mathbb{F}_{p^n} . For cryptographic applications, we are mostly interested in a prime-order subgroup generated by a rational point $P \in E(\mathbb{F}_{p^n})$. Here we first give a high-level overview of a typical index calculus algorithm for finding an integer α such that $Q = \alpha P$ for $Q \in \langle P \rangle$.

1. Determine a *factor base* $\mathcal{F} \subset E(\mathbb{F}_{p^n})$.
2. Collect a set \mathcal{R} of *relations* by decomposing random points $a_i P + b_i Q$ into a sum of points from \mathcal{F} , i.e.,

$$\mathcal{R} = \left\{ a_i P + b_i Q = \sum_j P_{i,j} : P_{i,j} \in \mathcal{F} \right\}.$$

- When $|\mathcal{R}| \approx |\mathcal{F}|$, eliminate the righthand side using linear algebra to obtain an equation in the form $aP + bQ = \mathcal{O}$, and $\alpha = -a/b \pmod{\text{ord}(P)}$.

The last step of linear algebra is relatively well studied in the literature, so we will focus on the subproblem in the second step, namely, the point decomposition problem (PDP) on an elliptic curve in the rest of this paper.

Definition 1 (Point Decomposition Problem of m -th Order). *Given a rational point $R \in E(\mathbb{F}_{p^n})$ on an elliptic curve E and a factor base $\mathcal{F} \subset E(\mathbb{F}_{p^n})$, find, if they exist, $P_1, \dots, P_m \in \mathcal{F}$ such that*

$$R = P_1 + \dots + P_m.$$

2.2 Semaev’s Summation Polynomials

We can solve PDP by considering when a set of points sum to zero on an elliptic curve. It is straightforward that if two points sum to zero on an elliptic curve $E : y^2 = x^3 + ax + b$ in Weierstrass form, then their x -coordinates must be equal. Let us now consider the simplest yet nontrivial case where three points on E sum to zero. Let

$$Z = \left\{ \begin{array}{l} (x_1, y_1, x_2, y_2, x_3, y_3) \in \mathbb{F}_{p^n}^6 : (x_i, y_i) \in E(\mathbb{F}_{p^n}), i = 1, 2, 3; \\ (x_1, y_1) + (x_2, y_2) + (x_3, y_3) = \mathcal{O} \end{array} \right\}.$$

Clearly, Z is in the variety of the ideal $I \subset \mathbb{F}_{p^n}[X_1, Y_1, X_2, Y_2, X_3, Y_3]$ generated by

$$\left\{ \begin{array}{l} Y_i^2 - (X_i^3 + aX_i + b), i = 1, 2, 3; \\ (X_3 - X_1)(Y_2 - Y_1) - (X_2 - X_1)(Y_3 - Y_1) \end{array} \right\}.$$

Now let $J = I \cap \mathbb{F}_{p^n}[X_1, X_2, X_3]$. Using MAGMA’s `EliminationIdeal` function, we obtain that J is actually a principal ideal generated by the polynomial $(X_2 - X_3)(X_1 - X_3)(X_1 - X_2)f_3$, where

$$f_3 = X_1^2 X_2^2 - 2X_1^2 X_2 X_3 + X_1^2 X_3^2 - 2X_1 X_2^2 X_3 - 2X_1 X_2 X_3^2 - 2aX_1 X_2 - 2aX_1 X_3 - 4bX_1 + X_2^2 X_3^2 - 2aX_2 X_3 - 4bX_2 - 4bX_3 + a^2.$$

Clearly, the linear factors of this generator correspond to the degenerated case where two or more points are the same or of opposite signs, and f_3 is the 3rd summation polynomial, that is, the summation polynomial for three distinct points summing to zero.

Starting from the 3rd summation polynomial, we can recursively construct the subsequent summation polynomials f_m for $m > 3$ via taking resultants. As a result, the degree of each variable in f_m is 2^{m-2} , which grows exponentially as m . This is the observation Semaev made in his seminal work [15]. In short, his proposal is to consider factor bases of the following form:

$$\mathcal{F} = \left\{ (x, y) \in E(\mathbb{F}_{p^n}) : x \in V \subset \mathbb{F}_{p^n} \right\},$$

where V is a subset of \mathbb{F}_{p^n} . Then we solve PDP of m -th order via solving the corresponding $(m + 1)$ -th summation polynomial $f_{m+1}(X_1, \dots, X_m, \tilde{x}) = 0$, where \tilde{x} is the x -coordinate of the point to be decomposed.

Note that this factor base is naturally invariant under point negation. That is, $P_i \in \mathcal{F}$ implies $-P_i \in \mathcal{F}$. In this case, we have about $|\mathcal{F}|/2$ (trivial) relations $P_i + (-P_i) = \mathcal{O}$ for free, so we just need to find the other $|\mathcal{F}|/2$ nontrivial relations. In general, we will only discuss factor bases that are invariant under point negation, so by abuse of language, both \mathcal{F} and \mathcal{F} modulo point negation may be referred to as a factor base in the rest of this paper.

2.3 Weil Restriction

Restricting the x -coordinates of the points in a factor base to a subset of \mathbb{F}_{p^n} is important from a viewpoint of polynomial system solving. Take f_3 as an example. When decomposing a random point $aP + bQ$, we first substitute its x -coordinate into say X_3 , projecting the ideal onto $\mathbb{F}_{p^n}[X_1, X_2]$. The dimension of the variety of this ideal is nonzero. Therefore, we would like to pose some restrictions on X_1 and X_2 to reduce the dimension to zero so that the solving time can be more manageable.

When looking for solutions to a polynomial $f = \sum a_i X^i \in \mathbb{F}_{p^n}[X]$ in \mathbb{F}_{p^n} , we can view $\mathbb{F}_{p^n}[X]$ as a commutative affine algebra $\mathcal{A} = \mathbb{F}_{p^n}[X]/(X^{p^n} - X) \cong \mathbb{F}_{p^n}[X_1, \dots, X_n]/(X_1^p - X_1, \dots, X_n^p - X_n)$. This can be done by identifying the indeterminate X as $X_1\theta_1 + \dots + X_n\theta_n$, where $(\theta_1, \dots, \theta_n)$ is a basis for \mathbb{F}_{p^n} over \mathbb{F}_p . Hence, f can be identified as a polynomial $f_1\theta_1 + \dots + f_n\theta_n$, where $f_1, \dots, f_n \in \mathcal{A}' = \mathbb{F}_p[X_1, \dots, X_n]/(X_1^p - X_1, \dots, X_n^p - X_n)$, by appropriately sending each coefficient $a_i \in \mathbb{F}_{p^n}$ to $a_i^{(1)}\theta_1 + \dots + a_i^{(n)}\theta_n$ for $a_i^{(1)}, \dots, a_i^{(n)} \in \mathbb{F}_p$. Therefore, an equation $f = 0$ over \mathbb{F}_{p^n} will give rise to a system of equations $f_1 = \dots = f_n = 0$ over \mathbb{F}_p . This technique is known as the *Weil restriction* and is used in the Gaudry-Diem attack, in which the factor base is chosen to consist of points whose x -coordinates lie in a subspace V of \mathbb{F}_{p^n} over \mathbb{F}_p [5, 10].

2.4 Exploiting Symmetry

Naturally, the symmetric group S_m acts on a point decomposition $P_1 + \dots + P_m$ because elliptic curve groups are abelian. As noted by Gaudry in his seminal work [10], we can therefore rewrite the variables $x_1, \dots, x_m \in \mathbb{F}_{p^n}$ by elementary symmetric polynomials e_1, \dots, e_m , where $e_1 = \sum x_i$, $e_2 = \sum_{i \neq j} x_i x_j$, $e_3 = \sum_{i \neq j, i \neq k, j \neq k} x_i x_j x_k$, etc. Such rewriting can reduce the degree of summation polynomials and significantly speed up point decomposition [7, 11].

We might be able to exploit additional symmetry brought by actions of other groups, e.g., when the factor base is invariant under addition of small torsion points. For example, consider a decomposition of a point R under the action of addition of a 2-torsion point T_2 :

$$R = P_1 + \dots + P_n = (P_1 + u_1 T_2) + \dots + (P_{n-1} + u_{n-1} T_2) + \left(P_n + \left(\sum_{i=1}^{n-1} u_i \right) T_2 \right).$$

Clearly this holds for any $u_1, \dots, u_{n-1} \in \{0, 1\}$, so a decomposition can give rise to 2^{n-1-1} other decompositions. Similar to rewriting using the elementary symmetric polynomials for the action of S_m , we can also take advantage of this additional symmetry by appropriate rewriting [6].

Naturally, such kind of speed-up is curve-specific. Furthermore, even if the factor base is invariant under additional group actions, we may or may not be able to exploit such symmetry to speed up point decomposition depending on whether the action is “easy to handle in the polynomial system solving process” [6].

2.5 PDP on (twisted) Edwards Curves

Faugère et al. studied PDP on twisted Edwards, twisted Jacobi intersections, and Weierstrass curves [6]. For the sake of completeness, we include some of their results here. An Edwards curve over \mathbb{F}_{p^n} for $p \neq 2$ is defined by the equation $x^2 + y^2 = 1 + dx^2y^2$ for certain $d \in \mathbb{F}_{p^n}$ [4]. A twisted Edwards curve $tE_{a,d}$ over \mathbb{F}_{p^n} for $p \neq 2$ is defined by the equation $ax^2 + y^2 = 1 + dx^2y^2$ for certain $a, d \in \mathbb{F}_{p^n}$ [3]. A twisted Edwards curve is a quadratic twist of an Edwards curve by $a_0 = 1/(a - d)$. For $P = (x, y) \in tE_{a,d}$, $-P = (-x, y)$. Furthermore, the addition and doubling formulae for $(x_3, y_3) = (x_1, y_1) + (x_2, y_2)$ are given as follows.

$$\text{When } (x_1, y_1) \neq (x_2, y_2) : \begin{cases} x_3 = \frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2}, \\ y_3 = \frac{y_1y_2 - ax_1x_2}{1 - dx_1x_2y_1y_2}. \end{cases}$$

$$\text{When } (x_1, y_1) = (x_2, y_2) : \begin{cases} x_3 = \frac{2x_1y_1}{1 + dx_1^2y_1^2}, \\ y_3 = \frac{y_1^2 - ax_1^2}{1 - dx_1^2y_1^2}. \end{cases}$$

The 3rd summation polynomial for twisted Edwards curves is [6]:

$$f_{tE,3}(Y_1, Y_2, Y_3) = \left(Y_1^2Y_2^2 - Y_1^2 - Y_2^2 + \frac{a}{d} \right) Y_3^2 + 2\frac{d-a}{d}Y_1Y_2Y_3 + \frac{a}{d}(Y_1^2 + Y_2^2 - 1) - Y_1^2Y_2^2.$$

Again subsequent summation polynomials are obtained by taking resultants.

2.6 Symmetry and Decomposition Probability

Symmetry brought by group action on point decomposition will inevitably be accompanied by *decrease in decomposition probability*. For example, if a factor base \mathcal{F} is invariant under addition of a 2-torsion point, then the decomposition

probability for PDP of m -th order should decrease by a factor of 2^{m-1} . This is due to the same reason that the decomposition probability decreases by a factor of $m!$ because the symmetric group S_m acts on \mathcal{F} .

However, this simple fact seems to have been largely ignored in the literature. For example, Faugère et al. explicitly stated in Sect. 5.3 of their paper that “[the] probability to decompose a point [into a sum of n points from the factor base] is $\frac{1}{n!}$ ” for twisted Edwards or twisted Jacobi intersections curves, despite the fact that the factor base is invariant under addition of 2-torsion points [6]. At a first glance, this may not seem a problem, as we would expect to obtain 2^{n-1} solutions if we can successfully solve a PDP instance. (Unfortunately this is also *not true* in general. We will come back to it in more detail in Sect. 5.3.) However, when estimating the cost of a complete ECDLP attack, they proposed to *collapse* these 2^{n-1} relations into one in order to reduce the size of the factor base and thus the cost of the linear algebra, cf. Remark 5 of the paper. In this case, the decrease in decomposition probability *does* have an adverse effect, and their estimation for the overall ECDLP cost ended up being overoptimistic by a factor of at least 2^{n-1} .

3 Montgomery and Hessian Curves

3.1 Montgomery Curves

A Montgomery curve $M_{A,B}$ over \mathbb{F}_{p^n} for $p \neq 2$ is defined by the equation

$$By^2 = x^3 + Ax^2 + x \tag{1}$$

for $A, B \in \mathbb{F}_{p^n}$ such that $A \neq \pm 2, B \neq 0$, and $B(A^2 - 4) \neq 0$ [12]. For $P = (x, y) \in M_{A,B}, -P = (x, -y)$. Furthermore, the addition and doubling formulae for $(x_3, y_3) = (x_1, y_1) + (x_2, y_2)$ are given as follows. When $(x_1, y_1) \neq (x_2, y_2)$:

$$\begin{cases} x_3 = B \left(\frac{y_2 - y_1}{x_2 - x_1} \right)^2 - A - x_1 - x_2 = \frac{B(x_2y_1 - x_1y_2)^2}{x_1x_2(x_2 - x_1)^2}, \\ y_3 = \frac{(2x_1 + x_2 + A)(y_2 - y_1)}{x_2 - x_1} - \frac{B(y_2 - y_1)^3}{(x_2 - x_1)^3} - y_1. \end{cases}$$

When $(x_1, y_1) = (x_2, y_2)$:

$$\begin{cases} x_3 = \frac{(x_1^2 - 1)^2}{4x_1(x_1^2 + Ax_1 + 1)}, \\ y_3 = \frac{(2x_1 + x_1 + A)(3x_1^2 + 2Ax_1 + 1)}{2By_1} - \frac{B(3x_1^2 + 2Ax_1 + 1)^3}{(2By_1)^3} - y_1. \end{cases}$$

It was noted by Montgomery himself in his original paper that such curves can give rise to efficient scalar multiplication algorithms [12]. That is, consider a

random point $P \in M_{A,B}(\mathbb{F}_{p^n})$ and $nP = (X_n : Y_n : Z_n)$ in projective coordinates for some integer n . Then:

$$\begin{cases} X_{m+n} = Z_{m-n}[(X_m - Z_m)(X_n + Z_n) + (X_m + Z_m)(X_n - Z_n)]^2, \\ Z_{m+n} = X_{m-n}[(X_m - Z_m)(X_n + Z_n) - (X_m + Z_m)(X_n - Z_n)]^2. \end{cases}$$

In particular, when $m = n$:

$$\begin{cases} X_{2n} = (X_n + Z_n)^2(X_n - Z_n)^2, \\ Z_{2n} = (4X_n Z_n) ((X_n - Z_n)^2 + ((A + 2)/4)(4X_n Z_n)), \\ 4X_n Z_n = (X_n + Z_n)^2 - (X_n - Z_n)^2. \end{cases}$$

In this way, scalar multiplication on Montgomery curve can be performed without using y -coordinates, leading to fast implementation.

3.2 Summation Polynomials for Montgomery Curves

Following Semaev’s approach [15], we can construct summation polynomials for Montgomery curves. Like Weierstrass curves, the 2nd summation polynomial for Montgomery curves is simply $f_{M,2} = X_1 - X_2$. Now consider $P, Q \in M_{A,B}$ for $P = (x_1, y_1)$ and $Q = (x_2, y_2)$. Let $P + Q = (x_3, y_3)$ and $P - Q = (x_4, y_4)$. By the addition formula, we have

$$x_3 = \frac{B(x_2 y_1 - x_1 y_2)^2}{x_1 x_2 (x_2 - x_1)^2}, x_4 = \frac{B(x_2 y_1 - x_1 y_2)^2}{x_1 x_2 (x_2 + x_1)^2}.$$

It follows that

$$\begin{cases} x_3 + x_4 = \frac{2((x_1 + x_2)(x_1 x_2 + 1) + 2Ax_1 x_2)}{(x_1 - x_2)^2}, \\ x_3 x_4 = \frac{(1 - x_1 x_2)^2}{(x_1 - x_2)^2}. \end{cases}$$

Using the relationship between the roots of a quadratic polynomial and its coefficients, we obtain

$$(x_1 - x_2)^2 x^2 - 2((x_1 + x_2)(x_1 x_2 + 1) + 2Ax_1 x_2)x + (1 - x_1 x_2)^2.$$

From here, we can obtain for Montgomery curve the 3rd summation polynomial:

$$f_{M,3}(X_1, X_2, X_3) = (X_1 - X_2)^2 X_3^2 - 2((X_1 + X_2)(X_1 X_2 + 1) + 2AX_1 X_2) X_3 + (1 - X_1 X_2)^2,$$

as well as the subsequent summation polynomials via taking resultants:

$$f_{M,m}(X_1, \dots, X_m) = \text{Res}_X (f_{M,m-k}(X_1, \dots, X_{m-k-1}, X), f_{M,k+2}(X_{m-k}, \dots, X_m, X)).$$

3.3 Small Torsion Points on Montgomery Curves

A Montgomery curve always contains an affine 2-torsion point T_2 . Since $T_2 + T_2 = 2T_2 = \mathcal{O}$, it follows that $-T_2 = T_2$. If we write $T_2 = (x, y)$, then we can see that $y = 0$ in order for $-T_2 = T_2$, as $p \neq 2$. Substituting $y = 0$ into Eq. (1), we get an equation $x^3 + Ax^2 + x = 0$. The lefthand side factors into $x(x^2 + Ax + 1) = 0$, so we get

$$x = 0, \frac{-A \pm \sqrt{A^2 - 4}}{2}.$$

Therefore, the set of rational points over the definition field F_{p^n} of a Montgomery curve includes at least two 2-torsion points, namely, \mathcal{O} and $(0, 0)$. The other 2-torsion points may or may not be rational, so we will focus on $(0, 0)$ in this paper. Substituting $(x_2, y_2) = (0, 0)$ into the addition formula for Montgomery curves, we get that for any point $P = (x, y) \in M_{A,B}$, $P + (0, 0) = (1/x, -y/x^2)$.

To be able to exploit the symmetry of addition of $T_2 = (0, 0)$, we need to choose the factor base $\mathcal{F} = \{(x, y) \in E(F_{p^n}) : x \in V \subset F_{p^n}\}$ invariant under addition of T_2 . This means that V needs to be closed under taking multiplicative inverses. In other words, V needs to be a *subfield* of F_{p^n} , i.e., $V = F_{p^\ell}$ for some integer ℓ that divides n . In this case, f_m is invariant under the action of $x_i \mapsto 1/x_i$. Unfortunately, such an action is not linear and hence not easy to handle in polynomial system solving. How to take advantage of such kind of symmetry in PDP is still an open research problem.

3.4 Hessian Curves

A Hessian curve H_d over F_{p^n} for $p^n = 2 \bmod 3$ is defined by the equation

$$x^3 + y^3 + 1 = 3dxy \tag{2}$$

for $d \in F_{p^n}$ such that $27d^3 \neq 1$ [16]. For $P = (x, y) \in H_d$, $-P = (y, x)$. Furthermore, the addition and doubling formulae for $(x_3, y_3) = (x_1, y_1) + (x_2, y_2)$ are given as follows.

$$\text{When } (x_1, y_1) \neq (x_2, y_2) : \begin{cases} x_3 = \frac{y_1^2 x_2 - y_2^2 x_1}{x_2 y_2 - x_1 y_1}, \\ y_3 = \frac{x_1^2 y_2 - x_2^2 y_1}{x_2 y_2 - x_1 y_1}. \end{cases}$$

$$\text{When } (x_1, y_1) = (x_2, y_2) : \begin{cases} x_3 = \frac{y_1(1 - x_1^3)}{x_1^3 - y_1^3}, \\ y_3 = \frac{x_1(y_1^3 - 1)}{x_1^3 - y_1^3}. \end{cases}$$

3.5 Summation Polynomials for Hessian Curves

Following a similar approach outlined by Galbraith and Gebregiyorgis [9], we can construct summation polynomials for Hessian curves. First, we introduce a new variable $T = X + Y$, which is invariant under point negation. The 2nd summation polynomial for Hessian curves is simply $f_{H,2} = T_1 - T_2$. Now let

$$Z = \left\{ \begin{array}{l} (x_1, y_1, t_1, x_2, y_2, t_2, x_3, y_3, t_3) \in \mathbb{F}_{p^n}^9 : (x_i, y_i) \in H_d(\mathbb{F}_{p^n}), i = 1, 2, 3; \\ (x_1, y_1) + (x_2, y_2) + (x_3, y_3) = \mathcal{O}; x_i + y_i = t_i, i = 1, 2, 3 \end{array} \right\}.$$

Clearly, Z is in the variety of the ideal $I \subset \mathbb{F}_{p^n}[X_1, Y_1, T_1, X_2, Y_2, T_2, X_3, Y_3, T_3]$ generated by

$$\left\{ \begin{array}{l} X_i^3 + Y_i^3 + 1 - 3dX_iY_i, i = 1, 2, 3; \\ (X_3 - X_1)(Y_2 - Y_1) - (X_2 - X_1)(Y_3 - Y_1); \\ X_i + Y_i - T_i, i = 1, 2, 3 \end{array} \right\}.$$

Again we compute the elimination ideal $I \cap \mathbb{F}_{p^n}[T_1, T_2, T_3]$ and obtain a principal ideal generated by some polynomial. After removing the degenerate factors, we can obtain for Hessian curve the 3rd summation polynomial:

$$\begin{aligned} f_{H,3}(T_1, T_2, T_3) = & T_1^2 T_2^2 T_3 + dT_1^2 T_2^2 + T_1^2 T_2 T_3^2 + dT_1^2 T_2 T_3 + dT_1^2 T_3^2 - T_1^2 + \\ & T_1 T_2^2 T_3^2 + dT_1 T_2^2 T_3 + dT_1 T_2 T_3^2 + 3d^2 T_1 T_2 T_3 + 2T_1 T_2 + 2T_1 T_3 + \\ & 2dT_1 + dT_2^2 T_3^2 - T_2^2 + 2T_2 T_3 + 2dT_2 - T_3^2 + 2dT_3 + 3d^2, \end{aligned}$$

as well as the subsequent summation polynomials via taking resultants:

$$f_{H,m}(T_1, \dots, T_m) = \text{Res}_T (f_{H,m-k}(T_1, \dots, T_{m-k-1}, T), f_{H,k+2}(T_{m-k}, \dots, T_m, T)).$$

3.6 Small Torsion Points on Hessian Curves

As we shall see in Sect. 4.1, we will compare elliptic curves in various forms that are isomorphism to one another over the same definition field. As a result, we will only experiment with those Hessian curves that include 2-torsion points like Montgomery or (twisted) Edwards curves. Since $T_2 + T_2 = 2T_2 = \mathcal{O}$, it follows that $-T_2 = T_2$. If we write $T_2 = (x, y)$, then we can see that $x = y$ in order for $-T_2 = T_2$, as $-T_2 = (y, x)$. Substituting $x = y$ into Eq. (2), we get an equation $2x^3 - 3dx^2 + 1 = 0$. Therefore, a Hessian curve $H_d(\mathbb{F}_{p^n})$ has a 2-torsion point (ζ, ζ) if the polynomial $2X^3 - 3dX^2 + 1$ has a root ζ in \mathbb{F}_{p^n} . In this case, the addition of this 2-torsion point to a point (x, y) would give a point (x', y') , where

$$\begin{cases} x' = \frac{\zeta y^2 - \zeta^2 x}{\zeta^2 - xy}, \\ y' = \frac{\zeta x^2 - \zeta^2 y}{\zeta^2 - xy}. \end{cases}$$

Obviously, typical factor bases are not invariant under addition of this 2-torsion point in general.

A Hessian curve always contains a 3-torsion point T_3 such that $3T_3 = \mathcal{O}$ [16]. If we let $T_3 = (x, y)$, then we see that $2(x, y) = -(x, y) = (y, x)$, substituting which into the doubling formula, we get:

$$\begin{cases} \frac{y(1-x^3)}{x^3-y^3} = y, \\ \frac{x(y^3-1)}{x^3-y^3} = x. \end{cases}$$

Since x and y cannot be zero at the same time, we have $x^3 - y^3 = 1 - x^3 = y^3 - 1$, or $x^3 = y^3 = 1$. Now since $p^n \equiv 2 \pmod 3$, \mathbb{F}_{p^n} does not have any primitive cubic roots of unity, so $x = y = 1$, and $T_3 = (1, 1)$. By the addition formula, if $P = (x, y)$, then

$$P + T_3 = (x, y) + (1, 1) = \left(\frac{y^2 - x}{1 - xy}, \frac{x^2 - y}{1 - xy} \right).$$

However, for $P \in \mathcal{F}$, we only know that $t = x + y \in V \subset \mathbb{F}_{p^n}$, but we know nothing about $1 - xy$, which can lie outside of V . So again, typical factor bases are not invariant under addition of this 3-torsion point in general. Therefore, it is not clearly how to exploit such symmetry brought by addition of small torsion points for Hessian curves.

4 Experiments on PDP Solving

This section shows our experimental results, which we have conducted to compare the computational complexity of PDP on four different curves: Hessian (H), Weierstrass (W), Montgomery (M), and twisted Edwards (tE).

4.1 Experimental Setup

To make a fair comparison, we use curves in different forms but are nonetheless isomorphic to one another over the same definition field \mathbb{F}_{p^n} . That is, $H(\mathbb{F}_{p^n}) \cong W(\mathbb{F}_{p^n}) \cong M(\mathbb{F}_{p^n}) \cong tE(\mathbb{F}_{p^n})$ as groups, and we consider ECDLP in the same largest prime-order subgroup. We will also explicitly state whether the factor base is invariant under addition of 2-torsion points for each of the four forms under investigation.

We start from a Hessian curve H_d satisfying $x^3 + y^3 + 1 = 3dxy$ for $d \in \mathbb{F}_{p^n}$ such that the number of its rational points $\#H_d(\mathbb{F}_{p^n})$ is divisible by 12. As we have seen in Sect. 3.5, the factor base of H_d is in general not invariant under addition of 2-torsion points. From H_d , we can obtain an isomorphic Weierstrass curve $W_{a,b}$ satisfying $y^2 = x^3 + ax + b$ for $a = -27d(d^3 + 8)$ and $b = 54(d^6 -$

$20d^3 - 8$) [16]. The isomorphism $\phi_{W,H}$ from $W_{a,b}(\mathbb{F}_{p^n})$ to $H_d(\mathbb{F}_{p^n})$ is defined over \mathbb{F}_{p^n} and is given by sending $(u, v) \in W_{a,b}$ to $(x, y) \in H_d$, where

$$\begin{cases} x = \frac{36(d^3 - 1) - v}{6(u + 9d^2)} - \frac{d}{2}, \\ y = \frac{36(d^3 - 1) + v}{6(u + 9d^2)} - \frac{d}{2}. \end{cases}$$

The inverse $\phi_{H,W}$ is given by
$$\begin{cases} u = \frac{12(d^3 - 1)}{d + x + y} - 9d^2, \\ v = \frac{36(d^3 - 1)(y - x)}{d + x + y}. \end{cases}$$

The factor base of $W_{a,b}$ is in general not invariant under addition of 2-torsion points [6].

With a high probability, we can obtain a Montgomery curve $M_{A,B}$ satisfying $By^2 = x^3 + Ax^2 + x$ from $W_{a,b}$ by solving the following equations

$$\begin{cases} a = \frac{3 - A^2}{3B^2}, \\ b = \frac{2A^3 - 9A}{27B^3}. \end{cases}$$

The isomorphism $\phi_{W,M}$ is defined over \mathbb{F}_{p^n} and is given by sending $(u, v) \in W_{a,b}$ to $(x, y) \in M_{A,B}$ for $x = Bu - 1/3A$ and $y = Bv$. The inverse $\phi_{M,W}$ can be obtained by equation solving. As we have seen in Sect. 3.1, the factor base is invariant under addition of a particular 2-torsion point $(0, 0)$, though we are not able to exploit this symmetry in general.

Finally, we can obtain a twisted Edwards curve $tE_{a',d'}$ satisfying $a'x^2 + y^2 = 1 + d'x^2y^2$ from $M_{A,B}$ by taking $a' = (A + 2)/B$ and $d' = (A - 2)/B$. Again we let $a_0 = 1/(a' - d')$ be the amount of quadratic twist. The isomorphism $\phi_{W,tE}$ is defined over \mathbb{F}_{p^n} and given by sending $(u, v) \in W_{a,b}$ to $(x, y) \in tE_{a',d'}$, where

$$\begin{cases} x = \frac{2a_0u}{v}, \\ y = \frac{u - a_0}{u + a_0}. \end{cases}$$

The inverse $\phi_{tE,W}$ is given by
$$\begin{cases} u = \frac{a_0(1 + y)}{1 - y}, \\ v = \frac{2a_0^2(1 + y)}{x(1 - y)}. \end{cases}$$

As shown by Faugère et al. [6], the factor base is invariant under addition of the 2-torsion point $(0, -1)$.

As explained in Sect. 2.1, we focus on PDP in these experiments, as the linear algebra step is already well understood. Furthermore, we focus on the bottleneck computation in PDP, namely, the cost of the F4 algorithm for computing Gröbner bases of the polynomial systems obtained after rewriting using the elementary symmetric polynomials and applying the Weil restriction technique to summation polynomials. This way we will be taking advantage of the symmetry of S_m acting on point decompositions. However, we *did not* exploit symmetry of any other group actions. This is because we want to compare the *intrinsic* computational complexity of PDP and hence only consider the symmetry that is present in *all* curves. Exploiting further curve-specific symmetry whenever possible will result in further speed-up, but it would be independent of our findings here.

4.2 Experimental Results

Table 1 presents our experimental results for the case of $n = 5$. Here we choose our factor base by taking V as the base field \mathbb{F}_p of \mathbb{F}_{p^n} . All our experiment are done using the MAGMA computation algebra system (version 2.23-1) on a single core of an Intel Xeon CPU E7-4830 v4 running at 2 GHz. Comparisons to solve each PDP are done by running time (in second), Dreg, Matcost, and Rank. the “Dreg” is the maximum step degree reached during the execution of the F4 algorithm, which referred to as the “degree of regularity” in the literature [9], and provides an upper bound for the sizes of the Macaulay submatrices involved in the computation, the “Matcost” is a number output by the MAGMA implementation of the F4 algorithm and provides an estimate of the linear algebra cost during the execution of the F4 algorithm, and finally the “Rank” is the number of linearly independent relations we obtain once successfully solving a PDP instance. It is an important factor to consider, as it determines how many PDP instances we need to successfully solve in order to have enough relations for a complete ECDLP attack using index calculus.

We can clearly see that the PDP solving time and Matcost for twisted Edwards curves are much smaller than those for the other curves. In contrast, the degree of regularity for Montgomery and twisted Edwards curves is smaller than that of the other curves in the case of $m = 4$. Also, we can see that the rank for Hessian and Weierstrass curves is 1 in all cases, whereas for Montgomery and twisted Edwards curves, it is 4 and 5 in the case of $m = 3$ and $m = 4$, respectively. Last but not least, although we only present the results for small p (around 8-bit long) here, we have some preliminary results for larger p (around 16-bit and 32-bit long). Aside from slight difference in absolute running time, all other results such as Dreg, Matcost, and Rank are similar, so we do not repeat them here.

Table 1. Experimental results on PDP solving for the case of $n = 5$

m	p	Curve	Time	Dreg	Matcost	Rank
3	239	Hessian	0	6	42336.8	1
		Weierstrass	0	6	41259.0	1
		Montgomery	0	6	61239.0	4
		tEdwards	0	6	6308.4	4
	251	Hessian	0	6	41420.4	1
		Weierstrass	0	6	42132.0	1
		Montgomery	0	6	61127.9	4
		tEdwards	0	6	6308.4	4
4	239	Hessian	3.990	19	12066100000	1
		Weierstrass	3.680	19	12064700000	1
		Montgomery	3.489	18	11399100000	5
		tEdwards	0.150	18	54093000	5
	251	Hessian	3.459	19	12069800000	1
		Weierstrass	3.659	19	12066400000	1
		Montgomery	3.280	18	11401700000	5
		tEdwards	0.119	18	54102900	5

5 Analysis

5.1 Revisit Summation Polynomial in Each Forms

As we have seen in Sect. 4.2, PDP on (twisted) Edwards curves seems easier to solve than on other curves. The explanation offered by Faugère et al. is “due to the smaller degree appearing in the computation of Gröbner basis of \mathcal{S}_{D_n} in comparison with the Weierstrass case,” cf. Sect. 4.1.1 of their paper [6]. Unfortunately, this *cannot* explain the difference between (twisted) Edwards and Montgomery curves, as the highest degrees appearing in the computation of Gröbner bases are *the same* for these two curves. Therefore, there must be other reasons. Table 2 shows the sparsity of the polynomials before and after Weil restriction for each of the four curves under investigation in the case of $m = 2, 3, 4$. As there are n polynomials after Weil restriction, we only show the average numbers here. We also include the maximum number of terms for a polynomial of degree $2^{(m+1)-2}$ in m variables.

We can see that total number of terms for twisted Edwards curves is significantly fewer than that for the other curves in all cases. Naturally, this could lead to faster solving time with the F4 algorithm. We also note that, except for twisted Edwards curves, the summation polynomials before Weil restriction for the other curves are all 100% dense without any missing terms.

Table 2. Number of terms (experimental/maximum) in polynomial systems before and after Weil restrictions

m	Curve	Before Weil restriction			After Weil restriction		
		Total	Odd	Even	Total	Odd	Even
2	Hessian	6/6	2/2	4/4	5.2/6	2.0/2	3.2/4
	Weierstrass	6/6	2/2	4/4	5.2/6	2.0/2	3.2/4
	Montgomery	6/6	2/2	4/4	5.2/6	2.0/2	3.2/4
	tEdwards	4/6	0/2	4/4	3.2/6	0.0/2	3.2/4
3	Hessian	35/35	16/16	19/19	34.2/35	16.0/16	18.2/19
	Weierstrass	35/35	16/16	19/19	34.0/35	16.0/16	18.0/19
	Montgomery	35/35	16/16	19/19	33.4/35	16.0/16	17.4/19
	tEdwards	25/35	6/16	19/19	23.4/35	6.0/16	17.4/19
4	Hessian	495/495	240/240	255/255	493.2/495	239.4/240	253.8/255
	Weierstrass	495/495	240/240	255/255	492.0/495	238.4/240	253.6/255
	Montgomery	495/495	240/240	255/255	492.2/495	239.2/240	253.0/255
	tEdwards	255/495	0/240	255/255	253.0/495	0.0/240	253.0/255

5.2 Missing Terms of Summation Polynomials in (twisted) Edwards Curves

In this section, we will show that the summation polynomials for (twisted) Edwards curves *mainly* have terms of *even* degrees. The set of terms of even degrees is closed under multiplication, so intuitively this kind of polynomials are easier to solve, which can be the main reason for the efficiency gain observed in the case of (twisted) Edwards curves.

We shall make this intuition precise in Theorem 1, but before we state the main result, we need to clarify our terminology for ease of exposition. When a multivariate polynomial is regarded as a univariate polynomial in one of its variables T , we say that the coefficient a_i of a term $a_i T^i$ is an *even or odd-degree coefficient* depending on whether i is even or odd, respectively. Note that these coefficients are themselves multivariate polynomials in one fewer variables.

We say that a monomial $m = \prod_{i=1}^n x_i^{e_i}, e_i \geq 0$ in a multivariate polynomial in n variables is *of even degree* or simply an *even-degree monomial* if $\sum_i e_i$ is even; that it is *of odd degree* or simply an *odd-degree monomial* otherwise. In contrast, a monomial is *of (homogeneous) even parity* if all e_i are even; it is *of (homogeneous) odd parity* if all e_i are odd. A monomial is *of homogeneous parity* if it is either of homogeneous even or odd parity. Note that the definition of monomials of odd parity depends on the total number of variables in the polynomial, which is not the case for monomials of even parity because we regard 0 as even. For example, the monomial $x_1 x_2$ is a monomial of odd parity in a polynomial in x_1 and x_2 but not so in another polynomial in x_1, \dots, x_n for $n > 2$.

By abuse of language, we say that a polynomial is *of even or odd parity* if it is a linear combination of monomials of even or odd parity, respectively; that

We denote as s_{ij} the entry at the i -th row and j -th column of S for $1 \leq i, j \leq m + n$. Since both m and n are even, an even-degree coefficient a_{2k} or b_{2k} will appear in s_{ij} for which the sum of indices $i + j$ is even. Similarly, an odd-degree coefficient a_{2k+1} or b_{2k+1} will appear in s_{ij} for which the sum of indices $i + j$ is odd. Now recall that the determinant of S is defined as

$$\sum_{\sigma \in S_{n+m}} \operatorname{sgn}(\sigma) s_{1,\sigma(1)} \cdot s_{2,\sigma(2)} \cdots s_{m+n,\sigma(m+n)}.$$

We note that the sum of the indices of any summand is

$$\sum_i^{m+n} i + \sigma(i) = (m + n)(m + n + 1),$$

which is always even. Therefore, the odd-degree coefficients must appear an even number of times, thus completing the proof. \square

Lemma 2. *Let \mathcal{E} be a family of elliptic curves such that its 3rd summation polynomial $f_{\mathcal{E},3}(X_1, X_2, X_3)$ is of degree 2 in each variable X_i and of homogeneous parity. Then any subsequent summation polynomial $f_{\mathcal{E},m}(X_1, \dots, X_m)$ for $m > 3$ is of homogeneous parity.*

Proof. As the summation polynomial $f_{\mathcal{E},m+1}$ for $m \geq 3$ is defined recursively from $f_{\mathcal{E},m}$ and $f_{\mathcal{E},3}$ via taking resultants

$$f_{\mathcal{E},m+1}(X_1, \dots, X_{m+1}) = \operatorname{Res}_X (f_{\mathcal{E},m}(X_1, \dots, X_{m-1}, X), f_{\mathcal{E},3}(X_m, X_{m+1}, X)),$$

we shall prove this lemma by induction on m . Let $f_{\mathcal{E},m}(X_1, \dots, X_{m-1}, X) = a_{2m-2}X^{2m-2} + \dots + a_1X + a_0$ and $f_{\mathcal{E},3}(X_m, X_{m+1}, X) = b_2X^2 + b_1X + b_0$. By the premise that $f_{\mathcal{E},3}$ is of homogeneous parity, b_0 and b_2 must consist only of monomials (in X_m and X_{m+1}) of even parity. Furthermore, $b_1 = cX_mX_{m+1}$ for some constant c . This is because $f_{\mathcal{E},3}$ is of degree 2 in each variable, for which the only monomial of odd parity is $X_mX_{m+1}X$.

Now consider a term $c_kX_{m+1}^k$ of

$$f_{\mathcal{E},m+1}(X_1, \dots, X_m, X_{m+1}) = c_{2m-1}X_{m+1}^{2m-1} + \dots + c_1X_{m+1} + c_0$$

as a univariate polynomial in X_{m+1} . Again as $f_{\mathcal{E},3}$ is of degree 2 in X , we have the case of $n = 2$ in Eq. 3. Now X_{m+1} must come from b_1 , so we can conclude that

$$c_kX_{m+1}^k = \sum_i \alpha_i a_{\beta_i} a_{\gamma_i} b_0^{\delta_i} b_2^{\epsilon_i} X_m^k X_{m+1}^k,$$

where α_i a constant, $\beta_i, \gamma_i \in \{0, \dots, 2^{m-2}\}$, and δ_i, ϵ_i nonnegative integers such that $\delta_i + \epsilon_i + k = 2^{m-2}$. We will complete the proof by showing as follows that $c_kX_{m+1}^k$ is a polynomial in X_1, \dots, X_{m+1} of homogeneous parity for all k .

1. If k is even, then by Lemma 1, β_i and γ_i are both even or both odd in each summand. In either case, the product $a_{\beta_i} a_{\gamma_i}$ is a polynomial in X_1, \dots, X_{m-1} of even parity. It follows that each summand is a polynomial of even parity because it is a product of polynomials of even parity. Hence $c_k X_{m+1}^k$ is a polynomial of even parity.
2. If k is odd, the situation is similar but slightly more complicated. By Lemma 1, exactly one of β_i and γ_i is odd in each summand, say β_i . By induction hypothesis, a_{β_i} is a polynomial in X_1, \dots, X_{m-1} of odd parity because it comes from $a_{\beta_i} X^{\beta_i}$ in $f_{\mathcal{E},m}$. It follows that each summand is a polynomial of odd parity because it is a product of a polynomial of even parity $a_{\gamma_i} b_0^{\delta_i} b_2^{\epsilon_i}$ and a polynomial of odd parity $a_{\beta_i} X_m^k X_{m+1}^k$. Hence $c_k X_{m+1}^k$ is a polynomial of odd parity.

□

By Lemma 2, $g_{\mathcal{E},m}(X_1, \dots, X_m) = f_{\mathcal{E},m+1}(X_1, \dots, X_m, x)$ is of homogeneous parity. Obviously, the monomials of even parity will remain of even degree after x is substituted. If m is even, then the monomials of odd parity in $f_{\mathcal{E},m+1}$ will become of even degree after x is substituted because an even number of odd numbers sum to an even number. Similarly if m is odd, then the monomials of odd parity in $f_{\mathcal{E},m+1}$ will become of odd degree after x is substituted. However, those odd-degree monomials that are *not* of homogeneous parity, e.g., $X_1^2 X_2$, cannot appear in $g_{\mathcal{E},m}$ by Lemma 2. This completes the proof of Theorem 1.

5.3 What Price for a Highly Symmetric Factor Base?

Last but not least, we discuss the price needed to pay to have a highly symmetric factor base \mathcal{F} that is invariant under more group actions in addition to that of the symmetric group S_m . As previewed in Sect. 2.6, we would expect that the effect of the decrease in decomposition probability due to additional symmetry in \mathcal{F} could be offset by that of the increase in number of solutions. For example, let us reconsider the group action of addition of T_2 in Sect. 2.4. If we could get 2^{m-1} solutions, then the loss of the factor of 2^{m-1} in decomposition probability would be compensated. This way everything would be the same as if there were no such symmetry, and we could exploit the additional symmetry at no cost.

Unfortunately, this proposition is *false* in general. Consider an example of $m = 4$. Let $Q_i = P_i + T_2$ for $i = 1, 2, 3, 4$. We can write down all $2^{m-1} = 8$ possible ways of a point decomposition under this group action:

$$\begin{aligned}
 P_1 + P_2 + P_3 + P_4 &= Q_1 + Q_2 + P_3 + P_4 \\
 &= Q_1 + P_2 + Q_3 + P_4 = Q_1 + P_2 + P_3 + Q_4 \\
 &= P_1 + Q_2 + Q_3 + P_4 = P_1 + Q_2 + P_3 + Q_4 \\
 &= P_1 + P_2 + Q_3 + Q_4 = Q_1 + Q_2 + Q_3 + Q_4.
 \end{aligned}$$

It is easy to find that we have only 5 linearly independent relations from these 8 relations, as there are nontrivial linear combinations summing to zero, e.g.:

$$(P_1 + P_2 + P_3 + P_4) - (Q_1 + Q_2 + P_3 + P_4) - (P_1 + P_2 + Q_3 + Q_4) + (Q_1 + Q_2 + Q_3 + Q_4) = \mathcal{O}.$$

As explained in Sect. 4.1, the factor bases for Montgomery and twisted Edwards curves are invariant under addition of 2-torsion points. For $m = 3$, we achieve maximum rank of $2^{m-1} = 4$. For $m = 4$, as we have explained above, we can only have rank 5, which is strictly less than the maximum possible rank $2^{m-1} = 8$.

Finally, we note that we have not exploited any symmetry for Hessian curves in our experiments. However, the rank for Hessian curves is always 1 in all our experiments. This shows that the factor base we have chosen for Hessian curves is *not* invariant under addition of small torsion points, as the rank would be >1 otherwise.

6 Conclusion

In this paper, we have experimentally explored index-calculus attack on ECDLP over different forms such as twisted Edwards curves, Montgomery, Hessian and Weierstrass curves under the totally fair conditions as they are isomorphic to each other over the same definition field \mathbb{F}_{p^n} and shown that twisted Edwards curves are clearly faster than others. We have investigated summation polynomials of all forms in detailed; found that big differences are exist in the number of terms; and proved that monomials of odd degrees in summation polynomial on twisted Edwards curves does not exist. We have seen that this difference causes less solving time of index-calculus attack on ECDLP over twisted Edwards than others.

Acknowledgments. This work is partially supported by JSPS KAKENHI Grant (C)(JP15K00183) and (JP15K00189) and Japan Science and Technology Agency, CREST and Infrastructure Development for Promoting International S&T Cooperation and Project for Establishing a Nationwide Practical Education Network for IT Human Resources Development, Education Network for Practical Information Technologies.

References

1. Bailey, D.V., Paar, C.: Optimal extension fields for fast arithmetic in public-key algorithms. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 472–485. Springer, Heidelberg (1998). <https://doi.org/10.1007/BFb0055748>
2. Bernstein, D.J.: Curve25519: new Diffie-Hellman speed records. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 207–228. Springer, Heidelberg (2006). https://doi.org/10.1007/11745853_14
3. Bernstein, D.J., Birkner, P., Joye, M., Lange, T., Peters, C.: Twisted Edwards curves. IACR Cryptology ePrint Archive, 2008:13 (2008)
4. Bernstein, D.J., Lange, T.: Faster addition and doubling on elliptic curves. IACR Cryptology ePrint Archive, 2007:286 (2007)
5. Diem, C.: On the discrete logarithm problem in class groups of curves. Math. Comput. **80**(273), 443–475 (2011)
6. Faugère, J., Gaudry, P., Huot, L., Renault, G.: Using symmetries in the index calculus for elliptic curves discrete logarithm. J. Cryptol. **27**(4), 595–635 (2014)

7. Faugère, J.-C., Perret, L., Petit, C., Renault, G.: Improving the complexity of index calculus algorithms in elliptic curves over binary fields. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 27–44. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_4
8. Galbraith, S.D., Gaudry, P.: Recent progress on the elliptic curve discrete logarithm problem. *Des. Codes Cryptogr.* **78**(1), 51–72 (2016)
9. Galbraith, S.D., Gebregiyorgis, S.W.: Summation polynomial algorithms for elliptic curves in characteristic two. In: Meier, W., Mukhopadhyay, D. (eds.) INDOCRYPT 2014. LNCS, vol. 8885, pp. 409–427. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-13039-2_24
10. Gaudry, P.: Index calculus for abelian varieties of small dimension and the elliptic curve discrete logarithm problem. *J. Symb. Comput.* **44**(12), 1690–1702 (2009)
11. Huang, Y.-J., Petit, C., Shinohara, N., Takagi, T.: Improvement of Faugère *et al.*'s method to solve ECDLP. In: Sakiyama, K., Terada, M. (eds.) IWSEC 2013. LNCS, vol. 8231, pp. 115–132. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-41383-4_8
12. Montgomery, P.L.: Speeding the Pollard and elliptic curve methods of factorization. *Math. Comput.* **48**, 243–264 (1987). [http://links.jstor.org/sici?sici=0025-5718\(198701\)48:177<243:STPAEC>2.0.CO;2-3](http://links.jstor.org/sici?sici=0025-5718(198701)48:177<243:STPAEC>2.0.CO;2-3)
13. Petit, C., Quisquater, J.: On polynomial systems arising from a Weil descent. *IACR Cryptology ePrint Archive 2012:146* (2012)
14. Pollard, J.M.: Monte Carlo methods for index computation mod p . *Math. Comput.* **32**, 918–924 (1978)
15. Semaev, I.A.: Summation polynomials and the discrete logarithm problem on elliptic curves. *IACR Cryptology ePrint Archive 2004:31* (2004)
16. Smart, N.P.: The Hessian form of an elliptic curve. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 118–125. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44709-1_11

Signature and Protocol



Two Mutual Authentication Protocols Based on Zero-Knowledge Proofs for RFID Systems

Hafsa Assidi^(✉), Edoukou Berenger Ayebie, and El Mamoun Souidi

Laboratory of Mathematics, Computer Science and Applications, Faculty of Sciences,
Mohammed V University in Rabat, BP 1014 RP, Rabat, Morocco
assidihafsa@gmail.com, berenger.ayebie@gmail.com, emsouidi@gmail.com

Abstract. RFID technology is becoming more useful and it is applied in various domains such as inventory management, supply chain, logistics, control access, e-passport, e-health and many other applications. Proposing an authentication protocol for RFID systems must find a compromise between the limitation of resources and the security requirements. In this paper, we propose two Zero-Knowledge mutual authentication protocols for the first time based on error correcting codes which are supposed to be resistant to quantum computer. Our proposed schemes have many advantages in terms of the approach used in the definition of the protocols and in terms of security properties and performances. Besides all the standard security requirements that our schemes fulfill, an adversary who can compromise the Reader cannot get the identifier of the Tag. We achieve this by using a Zero-Knowledge identification protocol.

Keywords: RFID · Authentication · Code-based cryptography
Zero-Knowledge · Security

1 Introduction

Radio Frequency Identification (RFID) is a contactless technology used to identify objects using radio frequency. Recently RFID systems have got much popularity and are now progressively substituting the bar-codes for identification. This technology is more and more drawing attention of researchers and industrials and it has numerous applications in different real life scenarios. For example, RFID systems are used in inventory management, supply chain, transportation, ticketing and control access systems, e-passport, e-health and many other domains. An RFID system is generally composed by two or three entities: Tags or tagged objects, Readers and Server as a backend database. In this case the Server has more resources than Tags and Readers and consequently it is more powerful in computing ability and storage. The Reader communicates with the Server through a secure network channel and with the Tag using a radio frequency interface which is supposed to be insecure. For this case we can give for

examples e-passport, access control and e-health *etc.* RFID system with only Reader and Tag is very useful in some applications like vehicle tags. It is in this category that our work is located. RFID Tags can be divided into three categories: passive Tags, which are powered by the radio wave from an RFID Reader and communicates with the Reader through backscattering. Active Tags, which are powered by their own energy sources (battery) and semi active Tags, which use internal energy sources to power their circuits. In this paper, the RFID system is formed by only one Reader and one Tag. This case can be illustrated by the following real life scenario: Vehicle authentication system when the reader is embedded in the vehicle and the vehicle key contains an RFID Tag. The constraint of resources and the security requirements are the main issues in the RFID systems and proposing an authentication protocol must verify the following security and privacy properties as cited in [13]: Anonymity, Secrecy, Mutual Authentication, Untraceability, Desynchronization Resilience, Forward Secrecy and Resist Replay Attack. We also introduce the notion of Inside Secrecy adapted from [20] that means that even if an Adversary can corrupt a Reader he cannot impersonate the Tag. The RFID authentication protocols can be divided into four types: the first type concerns the protocols that support cryptographic functions whether symmetric or asymmetric while in the second type, the protocols use pseudo random generator number PRNG and one way hash functions. The third category called lightweight protocols and generally based on some error correcting codes such as Cyclic Redundancy Check (CRC) and checksum. The last type (the ultra-lightweight) concerns those protocols that support only basic binary operations like XOR, AND *etc.*

Security and privacy are the two essential properties that must verify an authentication protocol for RFID systems. In the literature, many works have been proposed as authentication schemes and most of them are based on classical cryptography such as number theory assumptions or elliptic curve cryptography as it is shown these publications [1–4]. In 2006, Chien [5] presented an RFID authentication protocol for control access systems, his scheme provides the anonymity beside other properties. Thereafter Cao and Shen [6] proposed attacks on two RFID authentication protocols. Naveed *et al.* [3] designed a reliable and low cost RFID authentication protocol and later in [7], Chikouche *et al.* reported different attacks on two RFID authentication protocols based on coding theory assumptions. The authors of [8] proposed an authentication protocol based on randomized version of Niederreiter, however their protocol does not achieve forward secrecy and Reader's authentication. In [9] Chien and Lai suggested an RFID authentication protocol based on error correcting codes, basically, they used a confusion scheme to ensure the untraceability property. In [10], Malek and Miri presented an authentication protocol for RFID systems where they used the randomized McEliece cryptosystem but their protocol does not achieve desynchronization resilience. In 2014, Li *et al.* [11] designed a mutual authentication protocol based on error correcting codes. However, their proposal remains vulnerable to traceability and forward secrecy attacks. Thereafter their paper was improved by Chikouche *et al.* [12]. In [13], the authors improved an

existing mutual authentication protocol based on randomized McEliece and they proved that their scheme does not suffer from any problem in terms of security and privacy. In a survey of authentication protocols for RFID systems based on Zero-Knowledge proofs, we cite [14] where the authors present a private storage-security based on public key scheme for low-cost RFID. Independently Deng *et al.* in [15] proposed a formal RFID security and privacy framework based on a Zero-Knowledge (ZK) formulation.

On the other hand, code based cryptography is a promising alternative that resists to quantum computers. Therefore it is used in the construction of many important cryptographic primitives such as group signatures and ring signatures [21–23].

Our contribution. This paper consists in proposing two Zero-Knowledge authentication protocols based on coding theory assumptions for the RFID systems. In the literature, we notice that such authentication protocols for RFID systems are not well studied. Proposing an authentication protocol is a challenging task insofar as we should do a compromise between the required privacy and the low resources. In our construction, the security requirements are verified and even if we have a dishonest Reader or this one is compromised, an adversary cannot have any more Tag’s identifier since it is never revealed by the Tag in the authentication process.

Organization. The remainder of this paper is organized as follows: Sect. 2 is devoted to the Stern-Based RFID Zero-Knowledge authentication protocol. Then we propose the AGS-Based RFID authentication protocol in Sect. 3. Section 4 is devoted to analyze the performance of our proposal in terms of performances and security. We conclude in Sect. 5.

Preliminaries. In what follows, we recall some necessary notions in code-based cryptography to make the understanding of this paper easier.

Let \mathbb{F}_q be the finite field of cardinality q and let \mathcal{C} be a $[n, k, d]$ linear code over \mathbb{F}_q of length n , dimension k and minimum distance d , i.e. a linear subspace of dimension k of the vector space \mathbb{F}_q^n . Let $\mathcal{M}_{m \times n}(\mathbb{F}_q)$ denotes matrices over \mathbb{F}_q of m rows and n columns. We denote by $a \stackrel{\$}{\leftarrow} S$ when a is chosen uniformly random from a finite set S . Let $G \in \mathcal{M}_{k \times n}(\mathbb{F}_q)$ be a generator matrix of a $[n, k, d]$ code \mathcal{C} and $H \in \mathcal{M}_{(n-k) \times n}(\mathbb{F}_q)$ be its parity check matrix. Let \oplus denote the xor operation. We use x^\top to denote the transpose of x and $wt(x)$ to denote the Hamming weight of x . We define $Pr[A = a]$ as being the probability to have the event $A = a$. We denote by T and R the Tag and Reader respectively and by h a one way hash function. Let \mathcal{S}_n be the group of permutations over the set $\{1, \dots, n\}$. We define the following elements such as: σ is a permutation in \mathcal{S}_n , $u_i \in \mathbb{F}_2^k$ and Tag’s identifier $ID \in \mathbb{F}_2^n$ in Stern case and $ID = (e, m) \in \mathbb{F}_2^n \times \mathbb{F}_2^k$ in AGS case. We denote by $r = (r_1, \dots, r_N) \in \mathbb{F}_2^N$ a random vector of rotations such as $a_{r_i} = Rot_{r_i}(a)$ where Rot_{r_i} is a left shift rotation of a r_i positions. Let

$C = h(C_{1,1}, C_{2,1}, C_{3,1}, \dots, C_{1,N}, C_{2,N}, C_{3,N})$ be the vector obtained by hashing all the commitments in AGS case and $C = \{C_i\}_{1 \leq i \leq N}$ in Stern case. We define $b = (b_1, \dots, b_N) \in \mathbb{F}_2^N$ in AGS case and $b = (b_1, \dots, b_N) \in \mathbb{F}_3^N$ in Stern protocol. For protocols, we denote by \mathcal{P} , \mathcal{V} and \mathcal{A} the prover, the verifier and the adversary respectively. Our protocols are based on the Syndrome and General Decoding Problems that are based on coding theory and proved to be NP-complete in [18].

2 The Proposed Stern-Based RFID Zero-Knowledge Authentication Protocol

In this section, we propose a RFID Zero-Knowledge authentication protocols based on Stern identification scheme [19].

2.1 Mutual Authentication Stern's Protocol

We modify the original Stern identification scheme [19] that we describe in Algorithm 1, to add the mutual authentication and we call it Mutual Authentication Stern's Protocol (\mathcal{MASP}). Let H be an $(n - k) \times n$ random matrix, the Prover \mathcal{P} chooses $ID \in \mathbb{F}_2^n$ of weight w and computes $x = H \cdot ID^\top$. In Stern Identification scheme H and x are public and ID is the Prover \mathcal{P} secret Identity. In this version of Stern x is considered as a shared secret between \mathcal{P} and \mathcal{V} .

Algorithm 1. Mutual Authentication Stern's Protocol (\mathcal{MASP})

1. Let $H \in \mathcal{M}_{(n-k) \times n}(\mathbb{F}_2)$, the prover \mathcal{P} randomly chooses a binary string u of length n and a permutation $\sigma \in \mathcal{S}_n$. He then sends c_1 , c_2 and c_3 to \mathcal{V} defined by:

$$c_1 = h(\sigma|Hu^\top); c_2 = h(\sigma(u)); c_3 = h(\sigma(u \oplus ID))$$
 (where h is a hash function).
 2. \mathcal{V} sends a random challenge $g \in \{0, 1, 2\}$ to \mathcal{P} .
 3. three possibilities occur:
 - (a) if $g = 0$: \mathcal{P} reveals u and σ .
 - (b) if $g = 1$: \mathcal{P} reveals $(u \oplus ID)$ and σ .
 - (c) if $g = 2$: \mathcal{P} reveals $\sigma(u)$ and $\sigma(ID)$.
 4. Verification step by the verifier:
 - (a) if $g = 0$: \mathcal{V} checks that c_1 and c_2 received at Step 1 were correctly computed and sends $L = h(\sigma(u) \parallel x)$ to \mathcal{P} .
 - (b) if $g = 1$: \mathcal{V} checks that c_1 and c_3 received at Step 1 were correctly computed.
 - for c_1 one remarks that $Hu^\top = H(u \oplus ID)^\top \oplus x$
 - \mathcal{V} sends $L = h(Hu^\top)$ to \mathcal{P} .
 - (c) if $g = 2$: \mathcal{V} checks that c_3 and c_2 received at Step 1 were correctly computed and that the weight of $\sigma(ID)$ is exactly w and sends $L = h(\sigma(ID) \parallel x)$ to \mathcal{P} .
 5. Verification step by the prover:
 - (a) if $g = 0$: \mathcal{P} checks that L received at Step 4 is equal to $h(\sigma(u) \parallel x)$
 - (b) if $g = 1$: \mathcal{P} checks that L received at Step 4 is equal to $h(Hu^\top)$
 - (c) if $g = 2$: \mathcal{P} checks that L received at Step 4 is equal to $h(\sigma(ID) \parallel x)$
-

2.2 Security of Mutual Authentication Stern's Protocol

We now study the completeness and soundness of \mathcal{MASP} presented in Algorithm 1. We also show that this protocol, as the original one [19], is Zero-Knowledge with cheating probability of $\frac{2}{3}$.

Completeness. To ensure the completeness of our scheme, we show that the \mathcal{MASP} is always succeed when it is executed by honest prover and verifier.

Lemma 1. *If \mathcal{P} and \mathcal{V} honestly execute \mathcal{MASP} , we have for any round: $Pr[\mathcal{MASP}_{\mathcal{P},\mathcal{V}} = \text{Accept}] = 1$.*

Proof. The completeness is natural, we draw inspiration from Stern's completeness proof. We just notify that \mathcal{V} is authenticated until he has a valid y . Furthermore in the case of $g = 1$, \mathcal{V} can compute:

$$h(H \cdot (u \oplus ID)^\top \oplus x) = h(H \cdot u^\top \oplus H \cdot ID^\top \oplus H \cdot ID^\top) = h(H \cdot u^\top) \quad (1)$$

■

Soundness. To ensure the soundness properties, we show that a cheater cannot deceive the verifier that he knows the secret ID when it is not true.

Lemma 2. *If a cheater \mathcal{P} and an honest verifier \mathcal{V} execute \mathcal{MASP} , we have for any round: $Pr[\mathcal{MASP}_{\mathcal{P},\mathcal{V}} = \text{Accept}] = 2/3$.*

Proof. The proof of the soundness is the same as the Stern's one, where \mathcal{P} can cheat in the cases $g = 0$ and $g = 2$. We notify that in \mathcal{MASP} the additional requirements witch allow \mathcal{P} to authenticate \mathcal{V} are used after \mathcal{P} authentication. Then these additional requirements does not impact the result of \mathcal{P} authentication. Therefore, like in Stern's soundness proof, a cheater \mathcal{P} cannot anticipate the 3 challenges and the protocol clearly outputs "Accept" with a maximum probability of $2/3$. ■

Zero-Knowledge. To ensure the Zero-Knowledge, we show that during a normal execution the verifier learns nothing but only what he needs to confirm that the data he possesses is linked or not to the prover's secret.

Lemma 3. *The mutual Authentication Stern's Protocol \mathcal{MASP} is a prover-verifier Zero-Knowledge in ROM assuming the hardness of the syndrome decoding problem.*

Proof. In [19], Stern proves that his scheme is Zero-Knowledge in ROM assuming the hardness of SD problem which means that in normal execution, \mathcal{V} learns nothing but only what he needs to confirm that the data he possesses is linked or not to the prover's secret. From the Step 1 to 4 \mathcal{MASP} is constituted by fully Stern scheme, furthermore, in the last steps, \mathcal{P} sends nothing to \mathcal{P} . Therefore, based on Zero-Knowledge Stern's scheme proof, \mathcal{MASP} is a prover-verifier Zero-Knowledge in ROM assuming the hardness of SD problem. ■

Theorem 1. *The mutual Authentication Stern’s Protocol \mathcal{MASP} is a prover-verifier Zero-Knowledge with a cheating probability of $\frac{2}{3}$ verifying properties of completeness, soundness and Zero-Knowledge in ROM (Random Oracle Model) assuming the hardness of SD problem.*

Proof. We prove this theorem using Lemma 2 for completeness, Lemma 1 for soundness and Lemma 3 to prove the Zero-Knowledge. ■

2.3 \mathcal{MASP} -Based Authentication Scheme

We describe our \mathcal{MASP} -based RFID authentication scheme in two phases:

Initialization. During this step, the Customer generates randomly a $(n - k) \times n$ random matrix H and $ID \in \mathbb{F}_2^n$ of weight w and computes $x = H \cdot ID^\top$. The tuple (H, x, w) is stored in the Reader and the couple (H, ID) in the Tag.

Authentication. In the authentication step we repeat Algorithm 1 many times (N rounds) to reach a good security level. We describe the mutual authentication in the following steps:

1. The Tag generates all N commitments $C = \{C_i\}_{1 \leq i \leq N}$ as follow: for each $1 \leq i \leq N$ the Tag gets a vector $u_i \xleftarrow{\$} \mathbb{F}_2^n$ and a permutation $\sigma_i \xleftarrow{\$} \mathcal{S}_n$ and he computes $C_{i,1} = h(\sigma_i | Hu_i^\top)$; $C_{i,2} = h(\sigma_i(u_i))$; $C_{i,3} = h(\sigma_i(u_i \oplus ID))$ and sets $C_i = \{C_{i,1}; C_{i,2}; C_{i,3}\}$. The Tag sends $C = \{C_i\}_{1 \leq i \leq N}$ to the Reader.
2. The Reader sends back to the Tag the vector challenge $b \xleftarrow{\$} \mathbb{F}_3^N$.
3. After receiving the vector b , the Tag generates the vector RSP as the response. For each $1 \leq i \leq N$:
 - if $b_i = 0$: $RSP(i) = \{u_i; \sigma_i\}$
 - if $b_i = 1$: $RSP(i) = \{(u_i \oplus ID); \sigma_i\}$
 - if $b_i = 2$: $RSP(i) = \{\sigma_i(u_i); \sigma_i(ID)\}$
 the Tag sends $RSP = (RSP(1), \dots, RSP(N))$ to the Reader.
4. At this step the Reader verifies the Tag identity. For each $1 \leq i \leq N$:
 - if $b_i = 0$: Reader checks that $C_{i,1}$ and $C_{i,2}$ received at Step 2 are correctly computed and sets $L_i = h(\sigma_i(u_i) \parallel x)$.
 - if $b_i = 1$: Reader checks that $C_{i,1}$ and $C_{i,3}$ received at Step 2 are correctly computed.
 - For $C_{i,1}$ we notice that $Hu_i^\top = H(u_i \oplus ID)^\top \oplus x$.
 - The Reader sets $L(i) = h(Hu_i^\top)$.
 - if $b_i = 2$: Reader checks if $C_{i,3}$ and $C_{i,2}$ received at Step 2 are correctly computed and that the weight of $\sigma_i(ID)$ is exactly w and sets $L(i) = h(\sigma_i(ID) \parallel x)$.

To authenticate himself, the Reader sends $L = (L(1), \dots, L(N))$ to the Tag if no error occurs during this verification step else the Reader sets L to random data.

5. At the end, the **Tag** tries to authenticate the **Reader**. For each $1 \leq i \leq N$:
 - if $b_i = 0$: the **Tag** checks if $L(i)$ is exactly $h(\sigma_i(u_i) \parallel x)$.
 - if $b_i = 1$: the **Tag** checks if $L(i)$ is exactly $h(Hu_i^\top)$.
 - if $b_i = 2$: the **Tag** checks if $L(i)$ is exactly $h(\sigma_i(ID) \parallel x)$.

If all of these steps are passed, the **Tag** and the **Reader** are successfully authenticated.

2.4 Security of *MASP*-Based Authentication Scheme

In this part we give a security analysis of our *MASP*-based authentication scheme. Our RFID system is just composed by one legitimate couple of **Tag** and **Reader** consequently it's not necessary to study the anonymity and untraceability. These two properties have a meaning in the case when we have many **Tags**.

Mutual authentication. Let n_i where $1 \leq i \leq 5$ be five integers polynomial bounded in the security parameter. To study this property we use the following phases:

- Learning phase: during this phase \mathcal{A} observes the protocol running n_1 times between **Reader** **R** and **Tag** **T**, interacts n_2 times with **R** and interacts n_3 times with **T**.
- Challenge phase: during this phase \mathcal{A} tries to impersonate **T** n_4 times and \mathcal{A} tries to impersonate **R** n_5 times.

Assume that \mathcal{A} can successfully impersonate **T**, which means that \mathcal{A} can cheat all of the N rounds of our *MASP* protocol or he can guess the **Tag**'s *ID* in any one of n_4 sessions during the challenge phase. This event can occur with a probability of

$$\frac{n_4 w!(n-w)!}{n!} + \left(\frac{2}{3}\right)^{Nn_4}$$

which is negligible (where n , w and N denotes respectively the length of the code, the small weight of the secret and the number of rounds). Now we assume that \mathcal{A} can successfully impersonate **R** n_5 times, we distinguish three cases:

- $b_i = 0$: \mathcal{A} receives $C_{i,1} = h(\sigma_i | Hu_i^\top)$; $C_{i,2} = h(\sigma_i(u_i))$; $C_{i,3} = h(\sigma_i(u_i \oplus ID))$ and $RSP(i) = \{u_i; \sigma_i\}$ from **T** and must computes $L(i) = h(\sigma_i(u_i) \parallel x)$.
- $b_i = 1$: \mathcal{A} receives $C_{i,1} = h(\sigma_i | Hu_i^\top)$; $C_{i,2} = h(\sigma_i(u_i))$; $C_{i,3} = h(\sigma_i(u_i \oplus ID))$ and $RSP(i) = \{(u_i \oplus ID); \sigma_i\}$ from **T** and must computes $L(i) = h(Hu_i^\top)$.
- $b_i = 2$: \mathcal{A} receives $C_{i,1} = h(\sigma_i | Hu_i^\top)$; $C_{i,2} = h(\sigma_i(u_i))$; $C_{i,3} = h(\sigma_i(u_i \oplus ID))$ and $RSP(i) = \{\sigma_i(u_i); \sigma_i(ID)\}$ from **T** and computes $L(i) = h(\sigma_i(ID) \parallel x)$.

Depending on the cases presented above, without the knowledge of x and *ID*, impersonating **R** means that \mathcal{A} can find a pre-image under h , guess a good x , guess **T**'s *ID* or the value of σ_i or u_i are repeated in any one of n_5 sessions during the challenge phase. This event may occur with a probability of

$$\frac{1}{2^{Nn_5}} \left(\frac{n_1 + n_3}{2^{n-1}} + \frac{n_1 + n_3}{2^{n!}} \right)^{Nn_5} + \frac{n_5}{2^{n-k}} + \frac{n_5 w!(n-w)!}{n!} \tag{2}$$

which is negligible.

Desynchronizing attack. If an adversary blocks or modifies the information exchanged between the **Tag** and the **Reader** for a current session, the process of authentication will not succeed. However, in the next session the authentication will be achieved because we do not need to update the internal state in our scheme.

Replay attack. To achieve a replay attack, the adversary \mathcal{A} have to impersonate **Reader** or **Tag** by replaying the same messages of previous communications between legitimate **Tag** and **Reader** in order to forge the verification of **Tag**'s or **Reader**'s identity.

We suppose that the adversary chooses to impersonate the **Reader**, he chooses vector $b = (b_1, \dots, b_N)$ (Step 2) of a previous legitimate communication between the **Tag** and the target **Reader**, he receives C and RSP from the **Tag** and he has to send a valid vector L that fulfill verifications of Step 4 by the **Tag**. Achieving this goal means that the adversary has knowledge of x (which is supposed as a shared secret between the **Tag** and the **Reader**) or the **Tag** uses a repeated σ_i and u_i for all $1 \leq i \leq N$ as used in a legitimate conversation. However this can occur with a probability of $(\frac{1}{2^{n_i n_i}})^N$ which is negligible. Now, we suppose that \mathcal{A} tries to impersonate a specific **Tag**. Firstly, \mathcal{A} sends the vectors C and RSP used in previous communications between a **Reader** and a target **Tag**. The adversary succeeds this attack if the value of $b = (b_1, \dots, b_N)$ is the same as used in the previous communication. However this can occur with a probability of $(\frac{2}{3})^N$ which is negligible. Consequently, our protocol is not vulnerable to replay attack.

Secrecy. It is obvious that the proposed scheme achieves secrecy because our RFID authentication protocol is Zero-Knowledge and consequently, the sensitive data (secret or shared secret) are not transmitted using the radio frequency interface witch is supposed to be insecure.

Forward secrecy. Our protocol provides forward secrecy: we assume that there exists an adversary \mathcal{A} that can compromise the secret stored in **Tag**'s memory. We have to prove that this adversary cannot compromise the previous communications.

We suppose that an attacker has access to **Tag**'s memory then he gets it's ID .

Let (C', b', RSP', L') be the last conversation between the **Tag** and the **Reader**. Since b' is generated randomly and C' is computed using random elements (u_i, σ_i) , they are independent of ID . We use the same arguments to justify that even if the adversary needs ID or x to compute RSP' and L' he needs also σ_i and u_i which are chosen randomly. Thus if an intruder have access to **Tag**'s memory, he cannot compromise the previous communications.

Inside Secrecy. Let n_1, n_2 and n_3 be three integers polynomial bounded in the security parameter. We assume that \mathcal{A} can have access to the **Reader** memory it means that \mathcal{A} can have access to x . Now we study this property using the following phases:

- Learning phase: during this phase, \mathcal{A} observes the protocol running n_1 times between R and T and interacts n_2 times with R.
- Challenge phase: during this phase, \mathcal{A} tries n_3 times to impersonate T.

Assume that \mathcal{A} can successfully impersonate T which means that \mathcal{A} can recover the Tag's ID from $x = HID^\top$ or cheat all of the N rounds of our $MASP$ protocol in any one of n_3 sessions during the challenge phase. This event can occur with a probability of

$$p = \left(\left(\frac{1}{2} \right)^{Nn_3} + \frac{n_1 + n_2}{C_n^w} \right) \tag{3}$$

which is negligible. Where $C_n^w = \frac{n!}{w!(n-w)!}$.

3 The Proposed AGS-Based RFID Zero-Knowledge Authentication Protocol

In this section, we propose a RFID Zero-Knowledge authentication protocol based on AGS identification scheme [16].

3.1 The AGS Variant

Our proposed scheme is based first on a modification of AGS authentication protocol [16], which is a five pass interactive Zero-Knowledge proof, when the prover \mathcal{P} proves in a Zero-Knowledge way that he knows the secret vectors $(e, m) \in \mathbb{F}_2^n \times \mathbb{F}_2^k$ that verifies $x = mG + e$. Where the matrix G is double circulant and $wt(e) = w$, (G, w) are public parameters. We notice the difference between this variant and the original AGS authentication protocol that in our protocol we consider x as a shared secret unlike the original version where x is a public parameter. In our case even the verifier \mathcal{V} must prove to the prover \mathcal{P} that he knows the shared secret of a particular prover. We achieve this by verifying the commitments where the verifier has necessary to use the shared secret x . Consequently, we reach a mutual authentication between the prover and the verifier.

3.2 Security of the AGS Variant

In this subsection we study the completeness and soundness of the AGS variant presented in Algorithm 2. We also show that this protocol, as the original one [16], is Zero-Knowledge with cheating probability around $\frac{1}{2}$.

Completeness. A scheme is considered to be complete if the identification of a legitimate prover and verifier fail only with negligible probability. In prover authentication, the completeness is deduced from the proof in [16] and c_3 is computed as follows:

Algorithm 2. The AGS variant protocol

1. Let $(e, m) \in \mathbb{F}_2^n \times \mathbb{F}_2^k$, \mathcal{P} randomly chooses $u \in \mathbb{F}_2^k$ and a permutation σ of \mathcal{S}_n . Then \mathcal{P} sends to \mathcal{V} the commitments c_1 and c_2 such that: $c_1 = h(\sigma)$ and $c_2 = h(\sigma(uG))$.
 2. \mathcal{V} sends $0 \leq r \leq k - 1$ (the number of shifted positions) to \mathcal{P} .
 3. \mathcal{P} builds $e_r = Rot_r(e)$, $m_r = Rot_r(m)$ (e_r and m_r are a left shift rotation of r position applied to vectors e and m separately) and sends the last part of the commitment: $c_3 = h(\sigma(uG + e_r))$.
 4. \mathcal{V} sends $g \in \{0, 1\}$ to \mathcal{P} :
 5. Two possibilities:
 - if $g = 0$: \mathcal{P} reveals $(u + m_r)$ and σ .
 - if $g = 1$: \mathcal{P} reveals $\sigma(uG)$ and $\sigma(e_r)$.
 6. Verification Step, two possibilities:
 - if $g = 0$: \mathcal{V} verifies that c_1, c_3 have been honestly computed and sends $L = h(uG + e_r)$ to \mathcal{P} .
 - if $g = 1$: \mathcal{V} verifies that c_2, c_3 have been honestly computed and that $wt(\sigma(e_r)) = w$ and sends $L = h(\sigma(uG + e_r) + x_r)$ to \mathcal{P} . (x_r is a left shift rotation of r position applied to vectors x)
 7. Verification of the identity of \mathcal{V}
 - if $g = 0$, \mathcal{P} verifies if L is equal to $h((u + m_r)G + x_r) = h(uG + e_r)$.
 - if $g = 1$, \mathcal{P} verifies that L is equal to $h(\sigma(uG) + \sigma(e_r) + x_r) = h(\sigma(uG + e_r) + x_r)$.
-

- if $g = 0$,

$$c_3 = h(\sigma(u + m_r)G + \sigma(x_r)) = h(\sigma(uG + m_rG + m_rG + e_r)) = h(\sigma(uG + e_r)) \quad (4)$$

- if $g = 1$, $c_3 = h(\sigma(uG) + \sigma(e_r))$.

In verifier authentication we have:

- if $g = 0$,

$$L_1 = h((u + m_r)G + x_r) = h(uG + e_r) \quad (5)$$

- if $g = 1$,

$$L_2 = h(\sigma(uG) + \sigma(e_r) + x_r) = h(\sigma(uG + e_r) + x_r) \quad (6)$$

Soundness. The soundness property means that a dishonest prover or verifier cannot be authenticated with non negligible probability. In [16], the authors proves that a malicious prover cannot be authenticated successfully with probability higher than 1/2. In the verifier authentication, if this one has not the shared secret x he is not able to compute correctly the values L in both cases $g = 0$ or $g = 1$.

Zero-Knowledge. A scheme is considered as Zero-Knowledge if their is no sensitive data (secrets) that can be deduced from an execution of the protocol. It's proved in [16] that AGS protocol is Zero-Knowledge. Observing our scheme,

particularly where we add verifier authentication, a prover checks the prover's identity by computing L and this doesn't give any advantage to an eavesdropper to have a knowledge of x but the prover can compute it since he possess the secrets (e, m) .

3.3 Mutual Authentication Scheme Based on the AGS Variation

In the authentication process, we consider two entities: the Reader and the RFID Tag then the proposed scheme ensure a mutual authentication between the tow entities in Zero-Knowledge way.

Initialization phase. During this step of each Tag: the Customer stocks in Tag's memory G, e, m such that $e \in \mathbb{F}_2^n$ and $m \in \mathbb{F}_2^k$ are tow secrets constituting Tag identifier $ID = (e, m)$ with $wt(e) = w$ which is constant for all Tags. The public parameters are: $G \in \mathcal{M}_{k \times n}(\mathbb{F}_2)$ and ω . We consider $x = mG + e$ as a shared secret between the Tag and the Reader.

Authentication phase. The authentication protocol uses the AGS variant repeating each step N times rather than executing the protocol N rounds where N is the number of necessary iterations that we need to achieve the wanted security level as depicted in Fig. 1.

3.4 Security Analysis of Our AGS-Based RFID Authentication Scheme

We don't treat the anonymity and traceability property because we are interested in the case where we have only one Reader and one Tag.

Mutual authentication. In the challenge phase, we assume that the adversary impersonate n_1 times a Tag T and for the i^{th} session, \mathcal{A} generates a couple (C, RSP) where $1 \leq i \leq n_1$. On the other hand the adversary \mathcal{A} can impersonate the Reader n_2 times and in each session i , \mathcal{A} can produce a tuple (r, b, L) .

We assume that \mathcal{A} can impersonate the target T successfully. In that case, \mathcal{A} sends a valid couple (C, RSP) to the Reader R in one of the n_1 sessions during the challenge phase without knowing Tag's ID . Observing our protocol, \mathcal{A} can easily generates $C_{1,1}, C_{1,2}, \dots, C_{1,N}, C_{2,N}$ by choosing random $\sigma_i \in \mathcal{S}_n$ and $u_i \in \mathbb{F}_2^k$. However, the adversary cannot compute a valid $C_{3,i} = h(\sigma_i(u_i G + Rot_r(e_i)))$ without knowing the secret e which is a part of Tag's ID . Consequently, the probability to compute successfully a valid $C_{3,i}$ is $\frac{n_1}{C_n^w}$ (the probability to guess $e \in \mathbb{F}_2^k$ of weight w , where $C_n^w = \frac{n!}{w!(n-w)!}$). On the other hand, the adversary \mathcal{A} cannot compute a valid RSP . In the case where $b_i = 1$, the adversary cannot send a valid $RSP(i) = (\sigma_i(u_i G), \sigma_i(e_{r_i}), C_{2-b_i})$ without knowledge of e and he can guess e with probability $\frac{n_1}{C_n^w}$. When $b_i = 0$ he can send a valid RSP where $RSP(i) = (u_i + m_{r_i}, \sigma_i, C_{2-b_i})$ with a probability equal to $\frac{n_1}{2^k}$. Consequently

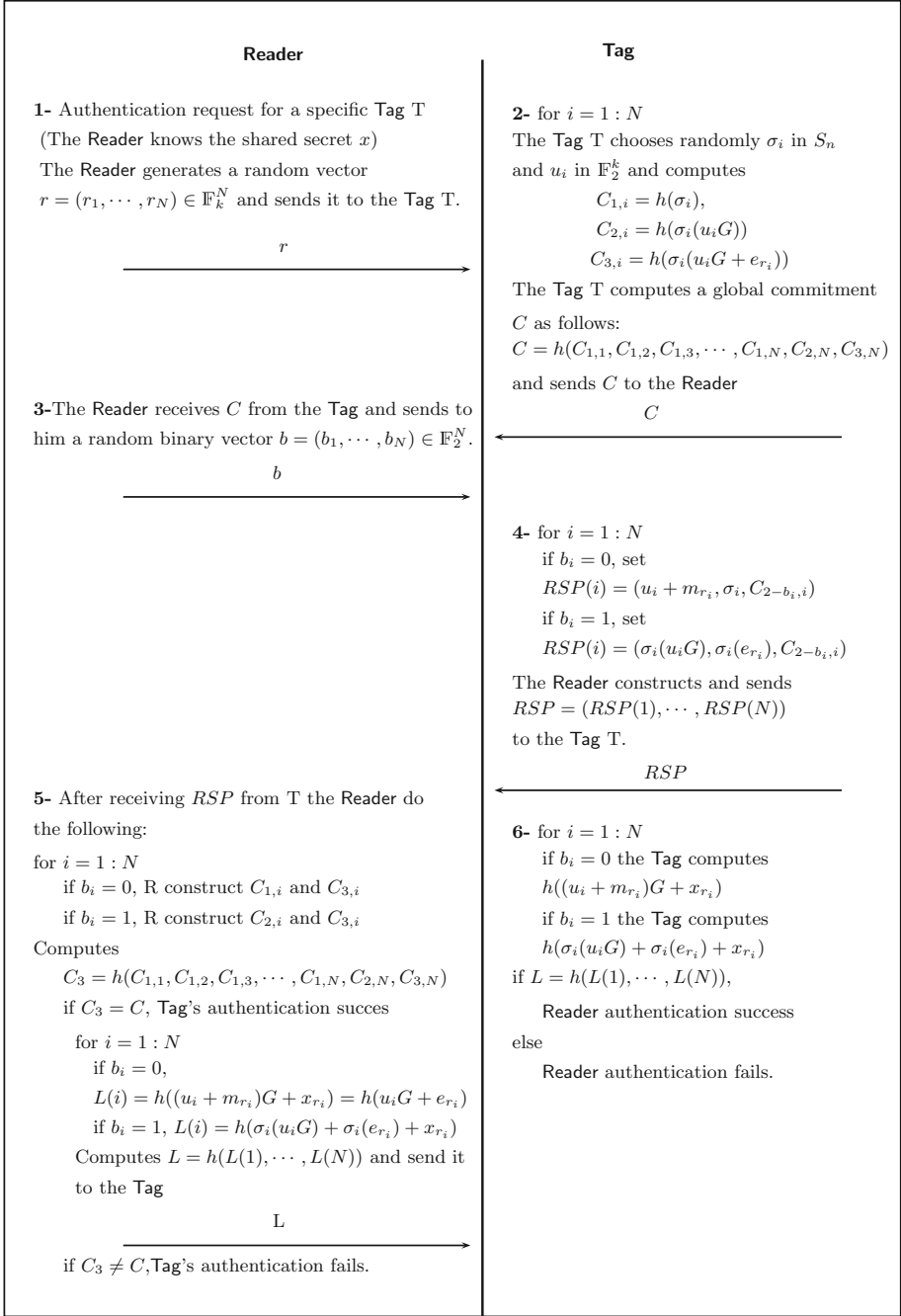


Fig. 1. Our AGS-based RFID authentication scheme

the adversary \mathcal{A} can generate a valid couple (C, RSP) with probability of $\frac{n_1}{C_n^w} + \frac{1}{2} \left(\frac{n_1}{C_n^w} + \frac{n_1}{2^k} \right)$ which is negligible.

Now, we assume that \mathcal{A} can impersonate the target Reader successfully. In that case, \mathcal{A} sends a valid tuple (r, b, L) : he can send r and b because they are chosen randomly. However it's infeasible for the adversary \mathcal{A} to compute a valid L because in both cases $b_i = 0$ and $b_i = 1$, \mathcal{A} has to use the shared secret x in order to compute a valid $L(i)$, guessing x occurs with probability of $\frac{1}{2} \left(\frac{2n_2}{2^n} + \frac{n_2}{C_n^w} \right)$ which is negligible.

Desynchronizing attack. If an adversary blocks or modifies the information exchanged between the Tag and the Reader for a current session, the process of authentication will not succeed. However, in the next session the authentication will be achieved because we do not need to update the internal state in our scheme.

Replay attack. To achieve a replay attack, the adversary \mathcal{A} has to impersonate Reader or Tag by replaying the same messages of previous communications between legitimate Tag and Reader in order to forge the verification of Tag's or Reader's identity.

We suppose that the adversary chooses to impersonate the Reader, he chooses vectors $r = (r_1, \dots, r_N)$ (Step 1 of Fig. 1) and $b = (b_1, \dots, b_N)$ (Step 3 of Fig. 1) of a previous legitimate communication between the Tag and the target Reader, he receives C and RSP from the Tag and he has to send a valid vector L that fulfill verifications of Step 6 of Fig. 1 by the Tag. Achieving this goal, means that the adversary has knowledge of x (which is supposed as a shared secret between the Tag and the Reader) or the Tag uses a repeated σ_i and u_i for all $1 \leq i \leq N$ as used in the legitimate conversation. However this can occur with a probability of $\left(\frac{1}{n!2^k}\right)^N$ which is negligible.

Now, we suppose that \mathcal{A} tries to impersonate a specific Tag. Firstly, \mathcal{A} sends a vector C and RSP used in previous communications between a Reader and a target Tag. The adversary succeeds this attack if the value of $b = (b_1, \dots, b_N)$ and $r = (r_1, \dots, r_N)$ are the same as used in the previous communication. However this can occur with a probability of $\left(\frac{1}{2^k}\right)^N$ which is negligible. Consequently, our protocol is not vulnerable to replay attack.

Secrecy. It is evident that the proposed scheme achieves the secrecy property because our RFID authentication protocol is Zero-Knowledge and consequently, the sensitive data (secret or shared secret) are not transmitted using the radio frequency interface which is supposed to be insecure.

Forward secrecy. Our protocol provides forward secrecy: we assume that there exists an adversary \mathcal{A} that can compromise the secret stored in Tag's memory. We have to prove that this adversary cannot compromise the previous communications.

We suppose that the attacker has access to Tag's memory then he gets it's $ID = (e, m)$.

Let (r', C', b', RSP', L') be the last conversation between the **Tag** and the **Reader**. Since r' and b' are generated randomly and C' is computed using random elements (u_i, σ_i) , they are independent of $ID = (e, m)$. We use the same arguments to justify that even if the adversary needs e or m to compute RSP' and L' he needs also σ_i and u_i which are chosen randomly. Thus if an intruder has access to **Tag**'s memory, he cannot compromise the previous communications.

Inside Secrecy. Let n_1, n_2 and n_3 three integers polynomial bounded in the security parameter. We assume that \mathcal{A} can have access to the **Reader** memory, it means that \mathcal{A} can have access to x . Now we define the following phases:

- Learning phase: during this phase, \mathcal{A} observes the protocol running n_1 times between **R** and **T**, interacts n_2 times with **R**.
- Challenge phase: during this phase \mathcal{A} tries to impersonate n_3 times **T**.

Assume that \mathcal{A} can successfully impersonate **T** which means that \mathcal{A} can recover the **Tag**'s $ID = (e, m)$ from $x = mG + e$ or cheat all of the N rounds of our AGS variant protocol in any one of n_3 sessions during the challenge phase. This event can occur with a probability of $(\frac{1}{2})^{Nn_3} + (n_1 + n_2) \left(\frac{1}{2^k} + \frac{1}{C_n^w} \right)$ witch is negligible.

4 Performance and Practical Results

In Table 1, we present a comparison between our RFID protocols (AGS based protocol and $MAS\mathcal{P}$ based one) and others presented in the literature based on coding theory namely: [10–12, 17].

The comparison shows that our scheme fulfill the security requirements: mutual authentication, resilience to desynchronization attacks, replay attack, secrecy and forward secrecy. Moreover, unlike the others schemes based on coding theory, our schemes achieve Inside Secrecy.

Table 1. Comparison of privacy and security requirements

	[10]	[11]	[12]	[17]	Our protocols
Mutual authentication	Yes	Yes	Yes	Yes	Yes
Desynchronizing attack	No	Yes	Yes	Yes	Yes
Replay attack	Yes	Yes	Yes	Yes	Yes
Secrecy	Yes	Yes	Yes	Yes	Yes
Forward secrecy	Yes	No	Yes	Yes	Yes
Inside secrecy	No	No	No	No	Yes

4.1 Performance Analysis

The performance of any authentication protocol for RFID systems are summarized in the storage space in **Tag**'s memory, computation cost both for the **Tag** and for the **Reader** and the communication cost between the **Tag** and the **Reader**.

Storage space in Tag’s memory. We stock in Tag’s memory the $ID = (e, m)$ which is of length $n + k$ and the first row of the circulant matrix to generate G which is of size k bits. In AGS protocol, the authors present the following parameters for 80 bits security level and cheating probability equal to 2^{-16} : $n = 698$, $k = 349$, $w = 70$. The number of required rounds is $N = 18$. Finally, the Tag’s storage is $n + 2k = 1386$ bits.

Communication cost. In general, the communication cost is measured by the number of bits transmitted during the exchange of messages between the Tag and the Reader. Now, we compute the number of bits transmitted from the Tag to the Reader. Observing our protocol, during one round the Reader sends (C, RSP) . We note by l_h the size of a hash function and l_σ the number of bits required to encode a permutation.

$$size(RSP) = \frac{N}{2} (k + l_\sigma + l_h + n + n + l_h) = N \left(\frac{k}{2} + \frac{l_\sigma}{2} + l_h + n \right)$$

We note $Comm_{T \rightarrow R}$ the communication cost of messages sends from T to R.

$$Comm_{T \rightarrow R} = size(C) + size(RSP) = l_h + N \left(\frac{k}{2} + \frac{l_\sigma}{2} + l_h + n \right) \quad (7)$$

On the other hand, we also note l_r the number of bits required to encode $1 < r_i < k - 1$ the number of shifts to apply rotations. Let (R, b, L) be the data transmitted from R to T, we have:

$$size(r) = Nl_r, \quad size(b) = N, \quad size(L) = l_h$$

We note $Comm_{R \rightarrow T}$ the communication cost of messages sends from R to T.

$$Comm_{R \rightarrow T} = N(l_r + 1) + l_h \quad (8)$$

Thus, by (7) and (8) we conclude that the communication cost of our proposed authentication protocol is:

$$Comm = Comm_{T \rightarrow R} + Comm_{R \rightarrow T} = N \left(\frac{k}{2} + \frac{l_\sigma}{2} + l_h + n + l_r + 1 \right) + 2l_h \quad (9)$$

Let l_p be the length of generating random number or hash, m the number of bits required to encode the integer r_i where $1 < r_i < k - 1$ and $|key|$ the length of key or ID .

Table 2. Comparison of performances

	Tag’s storage	$Comm_{T \rightarrow R}$	$Comm_{R \rightarrow T}$
Chikouche et al. [13]	$n = 2048$	$n + l_p$	$n + l_p$
Malek and Miri [10]	$(n + k + key)$	n	$2n + key + l_p$
Our protocol	$n + 2k = 1386$	$N(3l_p + k + n)$	$N(l_p + 1 + m)$

5 Conclusion

In this paper, we have proposed two mutual authentication protocols for RFID systems based on Zero Knowledge proofs and coding theory. Namely the *MASP*-based and the AGS-based authentication protocols. Our protocols verify all the important security and privacy requirements including secrecy, forward secrecy and inside secrecy. In addition, our proposal provides stronger resistance to desynchronizing and replay attacks. Compared to other protocols, if we suppose that the Reader is compromised or it is considered as dishonest, an adversary cannot have Tag's identifier since it has not revealed during the authentication and even the Reader ignore it. This security property is ensured by the Zero-Knowledge of our protocol. Moreover, the scheme is designed with pseudo random generators, hash functions, permutations and other simple operations. The number of bits required for storage in Tag's memory makes the protocol applicable and can be implemented for a large scale of low-cost RFID Tags and can also be extended to other compact hardware design.

References

1. Chen, C.L., Lai, Y.L., Chen, C.C., Deng, Y.Y., Hwang, Y.C.: RFID ownership transfer authorization systems conforming EPCglobal class-1 generation-2 standards. *Int. J. Netw. Secur.* **13**, 41–48 (2011)
2. Khedr, W.: On the security of moessner's and khan's authentication scheme for passive EPCglobal C1G2 RFID Zero-Knowledges. *Int. J. Netw. Secur.* **16**(5), 369–375 (2014)
3. Naveed, M., Habib, W., Masud, U., Ullah, U., Ahmad, G.: Reliable and low cost RFID based authentication system for large scale deployment. *Int. J. Netw. Secur.* **14**, 173–179 (2012)
4. Wei, C.H., Hwang, M.S., Chin, A.Y.H.: An authentication protocol for low-cost RFID Tags. *Int. J. Mob. Commun.* **9**(2), 208–223 (2011)
5. Chien, H.Y.: Secure access control schemes for RFID systems with anonymity. In: *Proceedings of the 7th International Conference on Mobile Data Management (MDM 2006)*, Nara, Japan, pp. 96–100 (2006)
6. Cao, T., Shen, P.: Cryptanalysis of two RFID authentication protocols. *Int. J. Netw. Secur.* **9**, 95–100 (2009)
7. Chikouche, N., Cherif, F., Cayrel, P.-L., Benmohammed, M.: Weaknesses in two RFID authentication protocols. In: El Hajji, S., Nitaj, A., Carlet, C., Souidi, E.M. (eds.) *C2SI 2015. LNCS*, vol. 9084, pp. 162–172. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18681-8_13
8. Cui, Y., Kobara, K., Matsuura, K., Imai, H.: Lightweight asymmetric privacy-preserving authentication protocols secure against active attack. In: *Proceedings of the Fifth Annual IEEE International Conference (PerComW 2007)*, White Plains, NY, USA, pp. 223–228 (2007)
9. Chien, H.Y., Lai, C.S.: ECC-based lightweight authentication protocol with untraceability for lowcost RFID. *J. Parallel Distrib. Comput.* **69**, 848–853 (2009)
10. Malek, B., Miri, A.: Lightweight mutual RFID authentication. In: *Proceedings of IEEE International Conference on Communications, Ottawa, ON, Canada*, pp. 868–872 (2012)

11. Li, Z., Zhang, R., Yang, Y., Li, Z.: A provable secure mutual RFID authentication protocol based on error-correct code. In: Proceedings of International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, Shanghai, China, pp. 73–78. IEEE (2014)
12. Chikouche, N., Foudil, C., Cayrel, P.L., Benmohammed, M.: A secure code-based authentication scheme for RFID systems. *Int. J. Comput. Netw. Inf. Secur.* **7**, 1–9 (2015)
13. Chikouche, N., Foudil, C., Cayrel, P.L., Benmohammed, M.: Improved RFID authentication protocol based on randomized McEliece cryptosystem. *Int. J. Netw. Secur.* **17**, 413–422 (2015)
14. El Moustaine, E., Laurent, M.: GPS+: a back-end coupons identification for low-cost RFID. In: Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks, WISEC 2013, Budapest, Hungary, pp. 73–78 (2013)
15. Deng, R.H., Li, Y., Yung, M., Zhao, Y.: A new framework for RFID privacy. In: Gritzalis, D., Preneel, B., Theoharidou, M. (eds.) ESORICS 2010. LNCS, vol. 6345, pp. 1–18. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15497-3_1
16. Aguilar Melchor, C., Gaborit, P., Schrek, J.: A new Zero-Knowledge code based identification scheme with reduced communication. In: 2011 IEEE Information Theory Workshop, pp. 648–652. IEEE Press, Paraty (2011)
17. Liu, Z., Zhang, W., Wu, C.: A lightweight code-based authentication protocol for RFID systems. In: Niu, W., Li, G., Liu, J., Tan, J., Guo, L., Han, Z., Batten, L. (eds.) ATIS 2015. CCIS, vol. 557, pp. 114–128. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48683-2_11
18. Berlekamp, E.R., McEliece, R.J., Van Tilborg, H.C.: On the inherent intractability of certain coding problems. *IEEE Trans. Inf. Theory* **24**(3), 384–386 (1978)
19. Stern, J.: A method for finding codewords of small weight. In: Cohen, G., Wolfmann, J. (eds.) Coding Theory 1988. LNCS, vol. 388, pp. 106–113. Springer, Heidelberg (1989). <https://doi.org/10.1007/BFb0019850>
20. Peeters, R., Hermans, J.: Wide strong private RFID identification based on Zero-Knowledge. *IACR Cryptol. ePrint Arch.* **2012**, 389 (2012)
21. Assidi, H., Ayebie, E.B., Souidi, E.M.: A code-based group signature scheme with shorter public key length. In: ICETE 2016, SECRIPT, vol. 4, pp. 432–439. SciTePress, Lisbon, Portugal, July 2016
22. Ayebie, B.E., Assidi, H., Souidi, E.M.: A new dynamic code-based group signature scheme. In: El Hajji, S., Nitaj, A., Souidi, E.M. (eds.) C2SI 2017. LNCS, vol. 10194, pp. 346–364. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-55589-8_23
23. Aguilar Melchor, C., Cayrel, P.-L., Gaborit, P.: A new efficient threshold ring signature scheme based on coding theory. In: Buchmann, J., Ding, J. (eds.) PQCrypto 2008. LNCS, vol. 5299, pp. 1–16. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88403-3_1



On New Zero-Knowledge Arguments for Attribute-Based Group Signatures from Lattices

Veronika Kuchta^(✉), Rajeev Anand Sahu, Gaurav Sharma,
and Olivier Markowitch

Université libre de Bruxelles, Brussels, Belgium
veronika.kuchta@ulb.ac.be

Abstract. Due to its emerging security and computational properties, lattice-based constructions are of prime concerns in recent research. Zero-knowledge evidences serve strongest security guarantees to cryptographic primitives. In this paper we formalize a new zero-knowledge argument (ZKA) suitable for lattice-based construction and employ it to security assurance of the proposed structure of attribute-based group signature on lattice assumption. To the best of our knowledge this paper proposes the first such construction.

1 Introduction

In recent years, lattice-based instantiations and concrete constructions of many well known cryptographic primitives have been formalized. Zero-knowledge arguments have been very standard and strong security evidences for such constructions. Both, the design of lattice-based protocol and their analysis by zero knowledge arguments are topics of interest in current research. In this paper, we formalize a new zero-knowledge argument (ZKA) for lattice-based primitives and employ it for security assurance of the proposed construction of attribute-based group signature from lattices. As we use the important ingredients of public key cryptography namely attribute-based construction, group signature and lattice-based setup, with broadly accessible public parameters; we systematically discuss these topics with their state of art below:

Attribute-based signature (ABS) is a variant of identity-based signature (IBS) where identities of the users are replaced by certain attributes which are supposed to be possessed by the candidate users. The required set of attributes, constrained to be satisfied by the users, is called predicate. A user satisfying the predicate can anonymously authenticate (sign) messages while maintaining fine-grained control over the attributes. In this view, ABS are fine-grained alternate to widely-used anonymous signature like group signature [8] and ring signatures [37]. The primary schemes of ABS were introduced by Maji et al. [28, 29]. Their schemes are designed over the bilinear group, but the better scheme is only proven secure in the generic group model. In recent years various useful primitives have been designed combining ABS with the standard signature

algorithms like group signature [19], ring signature [23] etc. to achieve more functionalities. Okamoto and Takashima [33] formalised the standard model of security for ABS, they follow the functional encryption proof technique of [22]. In a threshold ABS the candidate users are required to satisfy at least a threshold number of attributes. The ABS in [38] can be viewed as threshold variant of [33]. Another ABS is proposed in [11] with the property of full revocability. The constant sized ABS was first devised in [16].

Since the last decade, lattice-based cryptography has received significant attention due to its various appealing advantages including efficient computability, support to worst-case to average-case security guarantees and most importantly conjectured resistance against quantum attacks. Hence it has become the most promising candidate for post-quantum cryptography. Also in the practical platform lattice-based schemes are relevant by their asymptotic efficiency and simplicity. The first ABS from lattices (ABSL) was introduced in [30] following the ‘bonsai tree’ techniques. Security of the scheme is related to hardness of the small integer solution (SIS) problem. Nevertheless, their model suffers from long key size. Further, Zhang et al. [39] have suggested a comparative efficient scheme of ABSL using the framework of Boyen [6], their scheme also relies on worst-case hardness of the SIS problem. Over the signature size $(2k + 1) \cdot m \cdot \log(q)$ of [30], [39] offers signature with reduced size of $3m \cdot \log(q)$, where k is the number of attributes, m is dimension of lattice and q is order of the group. In [17] Jia et al. have presented a construction of ABSL in random oracle model. Their scheme is unforgeable against adaptively chosen message and selective access structure attacks under the similar hardness assumption- the hardness of SIS problem. They have improved efficiency of the scheme significantly with compared to [30, 39]. They have also extended their construction on NTRU lattice to further improve the efficiency in large amplitude. Very recently, El Bansarkhani and El Kaafarani [9] have proposed ABSL for expressive policies. For the purpose, they first construct a threshold ABS and then further an ABSL, for expressive (\wedge, \vee) -policies, based on their ABS. The drawback of scheme is the size of signature which grows linearly with the number of attributes. The ABSL is comparatively very recent topic of research and there is very good possibility of new frameworks and concrete constructions of the primitive.

The idea of group signature is introduced by Chaum and Van Heyst [8]. Most useful application of group signature is *anonymity* of the signer. The other security properties of group signature are *traceability* and *non-frameability*. The former ensures tracing of the actual signer in case of dispute and/or conflict, and the later ensures protection of uninvolved honest member. Group signatures are devised in two different plots- *static*, where no new members are allowed to join the group once the setup is done, and *dynamic*, where members are able to join dynamically any time, and the setup is updated accordingly. Standard generic framework of group signature are formalised by Bellare et al. [3, 4] with well-defined security properties. In [3] they have formalised a BMW model of security. Boneh and Boyen [5], Boyen and Waters [7] and Liang et al. [7, 24] have proposed useful structures of group signature scheme secured in this restricted

BMW model. Security of [4] is strengthened by formalising a dynamic setup of the group and the analysis is realised by the means of a non-interactive zero knowledge (NIZK) protocol.

The first realisation of lattice-based group signature (LBGS) was suggested in [14] by Gordon et al. at Asiacrypt 2010 on the learning with errors (LWE) assumption in the random oracle mode. Construction of their membership certificate is motivated by [12], encryption is motivated by [36] and a zero-knowledge proof technique is motivated by [31]. A major shortcoming of their scheme is size of public key and signature which are linear in the number of group members. To overcome the linear-size barrier, the first LBGS with logarithmic signature size was introduced by Laguillaumie et al. [20]. They also rely on LWE-based encryption like [14]. Also, their scheme does not support membership revocation. To address this issue, Langlois et al. [21] provided a group signature construction from lattice with verifier-local revocation. Unlike the previous schemes, their construction is free of encryption, but asymptotically, their scheme is as efficient as [20]. A more efficient and simpler construction of LBGS is presented in [32]. Unlike [20, 21], where encoding function is motivated by Boyen's technique [6], the group member's identities are encoded in [32] following more efficient building block of [1]. One of the main building blocks of groups signatures is a zero-knowledge proof between signer and verifier to prove the validity of signer's certificate and her membership in the group. Boyen [6] introduced an efficient group signature scheme from lattices using the technique of *mixing lattices and vanishing trapdoors*. Libert et al. [26] have suggested a Merkle-Tree Accumulator which is useful in design of LBGS and its zero-knowledge proof. Their scheme offers significant improvements compared to the existing constructions.

The Attribute Based Group Signature (ABGS) was introduced in [19]. This scheme provides only the *anonymity* of the signer and reveals the attributes of the signer satisfying the predicate. In a further version [18] they added the revocation property. To achieve full anonymity an ABGS scheme based on Oblivious Signature-Based Envelope (OSBE) protocol is proposed in [34]. The work in [10] presents a dynamic ABGS scheme with a possible application in anonymous survey for collection of attribute statistics. Another variety of ABGS scheme is Verifier-Local Revocation (VLR) scheme where only verifiers are involved in the revocation process. In [2] a constant signature sized VLR ABGS scheme has been presented which supports backward unlinkability and attribute anonymity. The scheme is proven to be secure in standard model.

1.1 Our Contribution

In this paper we formalize a new zero-knowledge argument (ZKA) for lattice-based construction and employ it to security assurance of the proposed structure of attribute-based group signature on lattice assumption. Particularly, our construction relies on the hardness of SIS and LWE problems. To support our zero-knowledge argument, we formalize a Merkle-*type* tree, motivated yet different construction by [26]. To the best of our knowledge this paper proposes the

first construction of attribute-based group signature from lattices, which is, due to the conjecture, quantum immune.

2 Preliminaries

Definition 1 (Small Integer Solution (SIS) Problem). *The $\text{SIS}_{n,m,q,\beta}^\infty$ problem is as follows: Given uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, find a non-zero vector $\mathbf{x} \in \mathbb{Z}^m$ s.t. $\|\mathbf{x}\|_\infty$ and $\mathbf{A} \cdot \mathbf{x} = \mathbf{0} \pmod q$.*

If $m, \beta = \text{poly}(n)$ and $q > \beta \tilde{O}(\sqrt{n})$, then $\text{SIS}_{n,m,q,\beta}^\infty$ problem is at least as hard as the worst-case lattice problem SIVP_γ for $\gamma = \beta \tilde{O}(\sqrt{nm})$. If $\beta = 1$, $q = \tilde{O}(n)$, $m = 2n \lceil \log q \rceil$, the $\text{SIS}_{n,m,q,1}^\infty$ problem is at least as hard as $\text{SIVP}_{\tilde{O}(n)}$.

Learning With Errors (LWE). The LWE problem, first introduced by Regev [35], relies on the Gaussian error distribution χ , which is given as $\chi = \mathcal{D}_{\mathbb{Z},s}$ over the integers. The LWE problem assumes of access to a challenge oracle \mathcal{O} , which is either a purely random sampler \mathcal{O}_r or a noisy pseudo-random sampler \mathcal{O}_s , with some random secret key $\mathbf{s} \in \mathbb{Z}_q^n$. For positive integers n and $q \geq 2$, a vector $\mathbf{s} \in \mathbb{Z}_q^n$ and error term $\mathbf{e} \leftarrow \chi$, the LWE distribution $\mathbf{A}_{s,\chi}$ is sampled over $\mathbb{Z}_q^n \times \mathbb{Z}_q$. Chosen a vector $\mathbf{a} \in \mathbb{Z}_q^n$ uniformly at random it outputs the pair $(\mathbf{a}, \mathbf{t} = \langle \mathbf{a}, \mathbf{s} \rangle + \mathbf{e} \pmod q) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$. A more detailed description of χ can be found in [35]. The sampling oracles work in the following way:

\mathcal{O}_s : outputs samples $(\mathbf{a}, \mathbf{t}) = (\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + \mathbf{e}) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, where $\mathbf{s} \in \mathbb{Z}_q^n$ is uniformly distributed value across all invocations and $\mathbf{e} \in \mathbb{Z}_q$ is a fresh sample from χ .

\mathcal{O}_r : outputs truly random samples from $\mathbb{Z}_q^n \times \mathbb{Z}_q$.

Definition 2 (LWE Problem). *For an integer q and error distribution χ , the goal of $\text{LWE}_{q,\chi}$ in n dimensions is to find $\mathbf{s} \in \mathbb{Z}_q^n$ with overwhelming probability, given access to any arbitrary $\text{poly}(n)$ number of samples from $\mathbf{A}_{s,\chi}$ for some random \mathbf{s} .*

In matrix form this problem looks as follows: collecting the vectors $\mathbf{a}_i \in \mathbb{Z}_q^n$ into a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and the error terms $\mathbf{e}_i \in \mathbb{Z}$ which form \mathbf{e} and values $\mathbf{t}_i \in \mathbb{Z}_q$ as the entries of the m -dimensional vector $\mathbf{t} \in \mathbb{Z}_q^m$, we obtain the input $\mathbf{A}, \mathbf{t} = \mathbf{A}^t \mathbf{s} + \mathbf{e} \pmod q$.

2.1 A Lattice-Based Tree-Generator with Supporting Zero-Knowledge Argument of Knowledge

In this section, we first recall a lattice-based accumulator which motivates us to construct a new cryptographic tool, called tree generator, where the leaves of tree are generated via a Merkle-Damgård hash function with a root node in the beginning of the tree.

Definition 3 (Cryptographic Tree-Accumulator). *A tree accumulator scheme consists of the following four algorithms:*

$\underline{\text{TSetup}}(1^\lambda)$: On input security parameter λ , outputs public parameters pp .
 $\underline{\text{Taccum}}(\text{pp})$: On input a set $R = \{d_0, \dots, d_{N-1}\}$ of N public keys, output an accumulated value \mathbf{u} .
 $\underline{\text{TWitness}}(\text{pp}, \mathbf{d})$: On input a data set R and a value \mathbf{d} , output \perp , if $\mathbf{d} \notin R$. Otherwise output a witness \mathbf{w} , for the fact that \mathbf{d} is accumulated value in $\text{Taccum}(\text{pp})$.
 $\underline{\text{TVerify}}(\text{pp}, \mathbf{u})$: On input accumulated value \mathbf{u} and a pair (\mathbf{d}, \mathbf{w}) , output 1 or 0.
 A generator scheme is correct if for all public parameters pp holds:
 $\text{TVerify}(\text{pp}, \text{Taccum}(\text{pp}), \mathbf{d}, \text{TWitness}(\text{pp}, \mathbf{d})) = 1$ for all $\mathbf{d} \in R$.

In [26] the authors use a hash function which is based on the Small Integer Solution (SIS) problem. The idea for computing a hashed value of a vector \mathbf{u} is to compute first its syndrome $\mathbf{A}\mathbf{u} \in \mathbb{Z}_q^\nu$ and to output $\text{bin}(\mathbf{A}\mathbf{u} \bmod q) \in \{0, 1\}^{\mu/2}$. In the next definition we recall a family of lattice-based collision-resistant hash functions (LBCRHF)

Definition 4 (LBCRHF). Let \mathcal{H} be a function family with the map $\{0, 1\}^{\nu\kappa} \times \{0, 1\}^{\nu\kappa} \rightarrow \{0, 1\}^{\nu\kappa}$, which is defined as $\mathcal{H} = \{h_{\mathbf{A}} | \mathbf{A} \in \mathbb{Z}_q^{\nu \times \mu}\}$, where $\mathbf{A} = [\mathbf{A}_0 | \mathbf{A}_1]$, and $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{\nu \times \mu\kappa}$. For any vectors $(\mathbf{u}_0, \mathbf{u}_1) \in \{0, 1\}^{\nu\kappa} \times \{0, 1\}^{\nu\kappa}$ we have

$$h_{\mathbf{A}}(\mathbf{u}_0, \mathbf{u}_1) = \text{bin}(\mathbf{A}_0 \cdot \mathbf{u}_0 + \mathbf{A}_1 \cdot \mathbf{u}_1 \bmod q) \in \{0, 1\}^{\text{nk}}$$

which satisfies: $h_{\mathbf{A}}(\mathbf{u}_0, \mathbf{u}_1) = \mathbf{u} \Leftrightarrow \mathbf{A}_0 \cdot \mathbf{u}_0 + \mathbf{A}_1 \cdot \mathbf{u}_1 = \mathbf{G} \cdot \mathbf{u} \bmod q$.

Next, we recall the ‘power-of-2’ matrix \mathbf{G} , which is useful for the construction of our zero-knowledge proof: For $\mathbf{g} = (1, 2, \dots, 2^{\kappa-1}) \in \mathbb{Z}_q^1$, set $\mathbf{G} = \mathbf{g} \otimes \mathbf{I}_n \in \mathbb{Z}_q^{\nu \times \nu\kappa}$. Using this ‘gadget’ matrix, we can represent each vector $\mathbf{v} \in \mathbb{Z}_q^\nu$. In the next section, we provide a Merkle-Damgård construction which can be applied to the Merkle-Tree Generator.

3 Attribute-Based Group Signature

Definition 5. An attribute-based group signature scheme consists of the following six algorithms:

$\underline{\text{Setup}}(1^\lambda, 1^n)$: On input security parameter 1^λ and the size of attribute set 1^n , the central authority runs this randomized algorithm to output public parameters param and master secret key msk .
 $\underline{\text{ABKeyGen}}(\text{param}, \text{msk}, \mathbb{A}_i)$: The algorithm is run by the Key Generation Center (KGC). On input public parameters param , master secret key msk and an attribute set \mathbb{A}_i of user i , it generates user’s secret key $\text{sk}_{\mathbb{A}_i}$ corresponding to the user’s attribute set \mathbb{A}_i and a user’s public key pk_i . Furthermore, the algorithm outputs a group public key gpk , an issuing key ik for enrolling new group members by an issuing entity and a group master secret key gmsk for opening the signature by the group manager to trace and identify the signers.

$\langle \text{Join}(\text{param}, \text{gpk}, \text{pk}_i, \text{sk}_{\mathbb{A}_i}) \rangle, \langle \text{Issue}(\text{param}, \text{pk}_i, \text{ik}) \rangle$: This is an interactive protocol to allow new members to join the group. The protocol is run between a user i and an key issuing authority KIA. The key issuing entity outputs a certificate cert_i for user i and stores user's public key pk_i in a registration table.

$\text{ABGSign}(\text{param}, \text{sk}_{\mathbb{A}_i}, \text{m}, \Gamma)$: On input public parameters param , member's secret key $\text{sk}_{\mathbb{A}_i}$, a predicate Γ and a message m it returns a signature σ .

$\text{ABGVerify}(\text{param}, \text{gpk}, \sigma)$: On input public parameters param , group public key gpk and a signature σ the deterministic algorithm verifies the validity of the signature and outputs 1 if the signature is valid. Otherwise it outputs 0.

$\text{ABGOpen}(\text{param}, \text{gmsk}, \sigma)$: On input public parameters param , group master secret key gmsk and a signature σ it outputs either the attribute set \mathbb{A} or \perp .

3.1 Security Definitions

In this section we provide the main security properties of a secure ABGS scheme. The main security properties are attribute and user anonymity, traceability and non-frameability.

Fully anonymity of users. In general, anonymity property of an ABGS scheme means that it is hard for an adversary apart from the group manager to recover the identity of the signer. Similar to the construction in [4], we guarantee collusion incapacity of an adversary with group members by providing the secret keys of all group members to the adversary. Furthermore we give an adversary access to the open oracle in order to allow him to see the results of previous openings. In the following definition we consider an adversary \mathcal{A}_{uan} , who wants to break the fully user anonymity property, and a bit \mathbf{b} which is associated with the security experiment. We assume that the adversary acts in two stages where in the first stage - the so called find stage - it takes as input the set of user's secret keys $\text{sk}_{\mathbb{A}_i}$ and group public key gpk and outputs two identities i_0, i_1 and a message m .

Definition 6 (User anonymity). An ABGS scheme preserves user anonymity if the advantage of an adversary in winning the experiment $\text{Exp}_{\mathcal{A}_{\text{uan}}, \text{ABGS}}^{\text{U-ANO-b}}(1^\lambda, 1^n)$ (as follows below) is negligible:

- (1) $(\text{param}, \text{msk}) \leftarrow \text{Setup}(1^\lambda, 1^n)$
- (2) $(\text{gpk}, \text{ik}, \text{gmsk}, \text{sk}_{\mathbb{A}_i}, \text{sk}_i, \text{pk}_i) \leftarrow \text{ABGKeyGen}(\text{param}, \text{msk}), \text{u}\tilde{\text{sk}} = \{\text{sk}_{\mathbb{A}_i}, \text{sk}_i\}_{i \in [n]}$
- (3) $(\text{state}, i_0, i_1, \text{m}, \Gamma) \leftarrow \mathcal{A}_{\text{uan}}^{\text{ABGOpen}(\cdot)}(\text{find}, \text{param}, \text{gpk}, \text{u}\tilde{\text{sk}})$
- (4) Choose $\mathbf{b} \in \{0, 1\}$; $\sigma^* \leftarrow \text{ABGSign}(\text{param}, \text{sk}_{\mathbb{A}, i_{\mathbf{b}}}, \text{m}, \Gamma)$
- (5) $\mathbf{b}' \leftarrow \mathcal{A}_{\text{uan}}^{\text{ABGOpen}(\cdot)}(\text{guess}, \text{state}, \sigma^*)$

$\text{OABGOpen}(\cdot, \cdot)$ The adversary calls this oracle with some message m and a signature σ . The oracle runs $\text{Open}(\text{gmsk}, \sigma)$ to receive index i which allows to trace malicious signer.

An ABGS scheme is fully anonymous if for any PPT adversary \mathcal{A}_{uan} its advantage is negligible: $\text{Adv}_{\mathcal{A}_{\text{uan}}, \text{ABGS}}^{\text{U-ANO}} = \left| \Pr \left[\text{Exp}_{\mathcal{A}_{\text{uan}}, \text{ABGS}}^{\text{U-ANO-1}} = 1 \right] - \Pr \left[\text{Exp}_{\mathcal{A}_{\text{uan}}, \text{ABGS}}^{\text{U-ANO-0}} = 1 \right] \right|$.

Attribute anonymity. This property means that a verifier should be able to verify a signature corresponding to a predicate without revealing the attribute set. Attribute anonymity is especially useful if there is only one group member with a certain attribute, helping trace back to the identity of the user.

Definition 7 (Attribute anonymity). $\text{Exp}_{\mathcal{A}_{\text{att-ano}}, \text{ABGS}}^{\text{Attr-ANO-b}}(1^\lambda, 1^n)$:

- (1) $(\text{param}, \text{msk}) \leftarrow \text{Setup}(1^\lambda, 1^n)$
- (2) $(\mathbb{A}_{i,0}, \mathbb{A}_{i,1}, \Gamma^*) \leftarrow \mathcal{A}_{\text{att-ano}}(\text{param})$ s.t. $(\Gamma^*(\mathbb{A}_{i,0}) = \Gamma^*(\mathbb{A}_{i,1}) = \mathbf{b}), \mathbf{b} \in \{0, 1\}$
- (3) $\text{sk}_{\mathbb{A}_{i,0}} \leftarrow \text{ABGKeyGen}(\text{param}, \text{msk}, \mathbb{A}_{i,0}), \text{sk}_{\mathbb{A}_{i,1}} \leftarrow \text{ABGKeyGen}(\text{param}, \text{msk}, \mathbb{A}_{i,1})$
- (4) $\mathbf{b}' \leftarrow \mathcal{A}_{\text{att-ano}}^{\text{OABGSign}(\text{param}, \mathbb{A}_{i,\cdot}, \cdot)}(\text{param}, \text{sk}_{\mathbb{A}_{i,0}}, \text{sk}_{\mathbb{A}_{i,1}})$
- (5) If $\mathbf{b} = \mathbf{b}'$, return 1, else return 0.

$\text{OABGSign}(\text{param}, \mathbb{A}'_i, \cdot)$: On input public parameters param and an attribute set \mathbb{A}'_i the oracle runs $\text{sk}_{\mathbb{A}'_i} \leftarrow \text{ABGKeyGen}(\text{param}, \text{msk}, \mathbb{A}'_i)$. Furthermore upon receiving $\text{sk}_{\mathbb{A}'_i}$ it runs $\sigma \leftarrow \text{ABGSign}(\text{param}, \text{sk}_{\mathbb{A}'_i}, \mathbf{m})$ on some message \mathbf{m} . It outputs a signature σ . An ABGS scheme is attribute-anonymous if for any PPT adversary $\mathcal{A}_{\text{att-ano}}$ its advantage is negligible:

$$\text{Adv}_{\mathcal{A}_{\text{uan}}, \text{ABGS}}^{\text{Attr-ANO}} = \left| \Pr \left[\text{Exp}_{\mathcal{A}_{\text{att-ano}}, \text{ABGS}}^{\text{Attr-ANO-1}} = 1 \right] - \Pr \left[\text{Exp}_{\mathcal{A}_{\text{att-ano}}, \text{ABGS}}^{\text{Attr-ANO-0}} = 1 \right] \right|$$

Full-Traceability. We assume that in case of malicious behavior, signer's identity can be revealed by the group manager using manager's secret key. In other words, it means that no collusion of group members should enable to create a valid signature which cannot be opened by the group manager. As mentioned in [3], the group manager could be dishonest and accuse an user in malicious behavior. In order to avoid this dishonest behavior of the user, we can ask the group manager to output a proof together with the index i , after running the `Open` algorithm. The verification of the proof can take place by running an additional algorithm - `Judge` - on input a signature σ , identity i and proof π .

Definition 8 (Full-Traceability). We say that an ABGS scheme is fully traceable if the advantage of an adversary $\mathcal{A}_{\text{f-trace}}$ to win the following experiment is negligible. $\text{Exp}_{\mathcal{A}_{\text{f-trace}}, \text{ABGS}}^{\text{Full-Trace}}(1^\lambda, 1^n)$:

- (1) $(\text{param}, \text{msk}) \leftarrow \text{Setup}(1^\lambda, 1^n)$
- (2) $(\text{gpk}, \text{ik}, \text{gmsk}, \text{sk}_{\mathbb{A}_i}, \text{sk}_i, \text{pk}_i) \leftarrow \text{ABGKeyGen}(\text{param}, \text{msk}), \text{usk} = \{\text{sk}_{\mathbb{A}_i}, \text{sk}_i\}_{i \in [n]}$
- (3) $(\mathbf{m}, \sigma) \leftarrow \mathcal{A}_{\text{f-trace}}^{\text{OABGSign}(\cdot), \text{OABGKeyGen}(\cdot), \text{OOpen}(\cdot)}(\text{gpk}, \text{gmsk})$

If $\text{ABGVerify}(\text{param}, \text{gpk}, \sigma) = 0$, return 0. If $\text{Open}(\text{param}, \text{gmsk}, \sigma) = \perp$, return 1. Let \mathcal{C} denote the list of all opened identities. If $\text{Open}(\text{param}, \text{gmsk}, \sigma) = i$ and $i \notin \mathcal{C}$, then return 1, else return 0.

$\text{OABGSign}(\text{param}, \text{sk}_i, \cdot, \cdot)$: On input public parameters param and an attribute set \mathbb{A}'_i the oracle runs $\text{sk}_{\mathbb{A}'_i} \leftarrow \text{ABGKeyGen}(\text{param}, \text{msk}, \mathbb{A}'_i)$. Furthermore upon receiving $\text{sk}_{\mathbb{A}'_i}$ it runs $\sigma \leftarrow \text{ABGSign}(\text{param}, \text{sk}_{\mathbb{A}'_i}, \mathbf{m})$ on some message \mathbf{m} . It outputs a signature σ .

$\mathcal{O}ABGKeyGen(\text{param}, \text{msk}, \cdot)$: On input public parameters and master secret key, giving an attribute set \mathbb{A}' , the oracle runs $(pk, \text{sk}_{\mathbb{A}'}) \leftarrow ABGKeyGen(\text{param}, \text{msk}, \mathbb{A}')$, where pk denotes all the public key of the $ABGKeyGen$. It outputs a tuple consisting of public keys and secret key $\text{sk}_{\mathbb{A}'}$.

$\mathcal{O}Open(\text{param}, \text{gmsk}, \cdot)$: On input a signature query σ , the oracle returns index $i \leftarrow Open(\text{param}, \text{gmsk}, \sigma)$.

An ABGS scheme is full-traceable if for any PPT adversary $\mathcal{A}_{f\text{-trace}}$ the following advantage is negligible: $\text{Adv}_{\mathcal{A}_{f\text{-trace}}, ABGS}^{\text{Full-Trace}} = \left| \Pr \left[\text{Exp}_{\mathcal{A}_{f\text{-trace}}, ABGS}^{\text{Full-Trace}} = 1 \right] \right|$.

Non-frameability. This security notion means that an adversary is not able to prove that some honest user created a valid signature. This property requires that it is impossible for two or more colluding users to produce a signature which would trace back to the non-colluded group member. As showed by Bellare et al. [3], non-frameability property is considered to be a version of collusion resistance. The two properties are the same in the sense that non-frameability prevents to create a signature which would be opened by a group manager and trace to a different member of the group. An attribute-based group signature scheme that is fully-traceable is automatically secure against framing. Bellare et al. [3] showed how to convert an adversary against framing into an adversary against full-traceability.

3.2 Merkle-Trees and Merkle-Damgård Compression Function for Lattice-Based Algorithms

In this section, we provide a variant of Merkle-Damgård Construction for zero-knowledge arguments for Merkle-tree algorithms which are inspired by the idea of Merkle-tree generators. The main difference is that the parent node is not calculated as an accumulated value of its children nodes. The idea is reversed in the sense that the children nodes of the tree are calculated using the parent node as one of the two inputs of the compression function. This idea is especially useful for attribute-based constructions, where certain attributes are assigned to the users via Merkle-tree construction. We fix a compression function $F : \{0, 1\}^\nu \times \{0, 1\}^\nu \rightarrow \{0, 1\}^\nu$. Let $\bar{\mathbf{A}} = [\mathbf{A} | \mathbf{A}_0 | \mathbf{A}_1 | \dots | \mathbf{A}_l]$ a set of matrices in $\mathbb{Z}_q^{\nu \times \mu}$ and two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{Z}_q^\mu$. The Merkle-Damgård compression function is defined as follows:

$$\begin{aligned} F_0(\mathbf{u}, \mathbf{v}) &= \text{bin}(\mathbf{A} \cdot \mathbf{u} + \mathbf{A}_0 \cdot \mathbf{v}) \\ F_1(\mathbf{u}, \mathbf{F}_0) &= \text{bin}(\mathbf{A}_1 \cdot (\mathbf{d}_{1,\mathbf{u}}\mathbf{u}) + \mathbf{f}_{1,\mathbf{u}}\mathbf{F}_0) \\ F_1(\mathbf{v}, \mathbf{F}_0) &= \text{bin}(\mathbf{A}_1 \cdot (\mathbf{d}_{1,\mathbf{v}}\mathbf{v}) + \mathbf{f}_{1,\mathbf{v}}\mathbf{F}_0) \\ F_2(\mathbf{u}, \mathbf{F}_{1,\mathbf{u}}) &= \text{bin}(\mathbf{A}_2 \cdot (\mathbf{d}_{2,\mathbf{uu}}\mathbf{u}) + \mathbf{f}_{2,\mathbf{uu}}\mathbf{F}_{1,\mathbf{u}}) \\ F_2(\mathbf{v}, \mathbf{F}_{1,\mathbf{u}}) &= \text{bin}(\mathbf{A}_2 \cdot (\mathbf{d}_{2,\mathbf{vu}}\mathbf{v}) + \mathbf{f}_{2,\mathbf{vu}}\mathbf{F}_{1,\mathbf{u}}) \\ F_2(\mathbf{u}, \mathbf{F}_{1,\mathbf{v}}) &= \text{bin}(\mathbf{A}_2 \cdot (\mathbf{d}_{2,\mathbf{uv}}\mathbf{u}) + \mathbf{f}_{2,\mathbf{uv}}\mathbf{F}_{1,\mathbf{v}}) \end{aligned}$$

$$\begin{aligned}
 F_2(\mathbf{v}, \mathbf{F}_{1,\mathbf{v}}) &= \text{bin}(\mathbf{A}_2 \cdot (\mathbf{d}_{2,\mathbf{v}\mathbf{v}}\mathbf{v}) + \mathbf{f}_{2,\mathbf{v}\mathbf{v}}\mathbf{F}_{1,\mathbf{v}}) \\
 &\vdots \\
 F_1(\mathbf{u}, \mathbf{F}_{1-1,\mathbf{u}\dots\mathbf{u}}) &= \text{bin}(\mathbf{A}_1 \cdot (\mathbf{d}_{1,\mathbf{u}\dots\mathbf{u}}\mathbf{u}) + \mathbf{f}_{1,\mathbf{u}\dots\mathbf{u}}\mathbf{F}_{1-1,\mathbf{u}\dots\mathbf{u}}) \\
 &\vdots \\
 F_1(\mathbf{v}, \mathbf{F}_{1-1,\mathbf{v}\dots\mathbf{v}}) &= \text{bin}(\mathbf{A}_1 \cdot (\mathbf{d}_{1,\mathbf{v}\dots\mathbf{v}}\mathbf{v}) + \mathbf{f}_{1,\mathbf{v}\dots\mathbf{v}}\mathbf{F}_{1-1,\mathbf{v}\dots\mathbf{v}})
 \end{aligned}$$

All functions $F_i, i \in [1]$ are elements in \mathbb{Z}_q^μ and are computed modulo q . To provide the zero-knowledge proof for our Merkle-Damgård tree construction we recall the following decomposition-extension technique presented in [27] which is used to represent certain vectors with coordinates in $\{-1, 0, 1\}$.

Let COM be a statistically hiding and computationally binding string commitment scheme. The common input to that scheme is a matrix \mathbf{A} and a vector \mathbf{y} , which belongs to the image of \mathbf{A} . The prover’s auxiliary input is given by vector \mathbf{x} . Before interaction, verifier and prover for an extension matrix $\mathbf{A}' \in \mathbb{Z}_q^{\nu \times 3\mu}$ by adding 2μ zero columns to the matrix \mathbf{A} .

Decomposition: Let $\mathbf{x} = (x_1, \dots, x_t)$ be a t -dimensional vector which we want to represent as a vector in k dimensions with entries in $\{-1, 0, 1\}$. For each $x_i \in \mathbf{x}$ consider the binary representation $x_i = b_{i,0}2^0 + b_{i,1}2^1 + \dots + b_{i,k-1}2^{k-1}$, such that $b_{i,j} \in \{-1, 0, 1\}$ for $j \in \{0, \dots, k-1\}$. We have $\mathbf{x} = \sum_{j=0}^{k-1} 2^j \tilde{\mathbf{u}}_j$.

Extension: For each index $j = 0, \dots, k-1$ we extend $\tilde{\mathbf{u}}_j$ to $\mathbf{u}_j \in \mathbf{S}_{3\mu}$, where $\mathbf{S}_{3\mu}$ is a set of all vectors in $\{-1, 0, 1\}^{2\mu}$, with exactly μ coordinates -1 , μ coordinates 0 , μ coordinates 1 . The extension works as follows: Let $\lambda_j^{(-1)}, \lambda_j^{(0)}, \lambda_j^{(1)}$ be the numbers of coordinates $-1, 0, 1$, respectively, then pick a vector $\mathbf{t}_j \in \{-1, 0, 1\}^{2\mu}$ that has exactly $(\mu - \lambda_j^{(-1)})$ coordinates -1 , $(\mu - \lambda_j^{(0)})$ coordinates 0 , $(\mu - \lambda_j^{(-1)})$ coordinates 1 . Set $\mathbf{u}_j = (\tilde{\mathbf{u}}_j || \mathbf{t}_j)$.

Using extension matrix \mathbf{A}' , we have $\mathbf{A}' \left(\sum_{j=0}^{k-1} 2^j \mathbf{u}_j \right) = \mathbf{y} \pmod q \Leftrightarrow$

$\mathbf{A}\mathbf{x} = \mathbf{y} \pmod q.$

Matrix Extension: On input a matrix $\overline{\mathbf{A}} = [\mathbf{A} | \mathbf{A}_0 | \mathbf{A}_1 | \dots | \mathbf{A}_1] \in \mathbb{Z}_q^{\nu \times (1+2)\mu}$ append 2μ zero-columns to the matrix \mathbf{A} and each \mathbf{A}_j , for $j \in [0, 1]$. The new matrix $\tilde{\mathbf{A}}$ is an element in $\mathbb{Z}_q^{\nu \times (1+2)3\mu}$.

3.3 Merkle-Tree Algorithm Using Merkle-Damgård Compression Function

We assume that our Merkle tree has $N = 2^l$ leaves, where l is the depth of the tree and is based on the lattice-based Merkle-Damgård hash function. The tree generator scheme consists of the following four algorithms:

TSetup(λ): On input security parameter λ , sample $\overline{\mathbf{A}} = [\mathbf{A} | \mathbf{A}_0 | \mathbf{A}_1 | \dots | \mathbf{A}_1]$ outputs $\text{pp} = \overline{\mathbf{A}}$.

Tcalc(pp, R): On input a set $R = \{\mathbf{d}_0, \dots, \mathbf{d}_{N-1}\}$ of N public keys representing the leaves, let $(j_1, \dots, j_1) \in \{0, 1\}^1$ be the binary representation of $j \in \{0, \dots, N-1\}$. Set $\mathbf{d}_j = \mathbf{u}_{j_1, \dots, j_1}$. The Merkle tree is formed using this technique with the N leaves $\mathbf{u}_{0,0,\dots,0}, \dots, \mathbf{u}_{1,1,\dots,1}$ as follows:

(1) Let $i \in [1]$ denote the current depth of the tree. Then the node $\mathbf{u}_{b_1, \dots, b_i} \in \{0, 1\}^{\nu\kappa}$ is defined using the Merkle-Damgård hash function: $F_i(\mathbf{u}, F_{i-1,(\cdot)})$ for the left-side leaf of the parent node and $F_i(\mathbf{v}, F_{i-1,(\cdot)})$ for the right-side leaf of the parent node.

(2) At the depth 0, the root \mathbf{n}_0 is defined as $F_0(\mathbf{u}, \mathbf{v})$, where $\mathbf{u}, \mathbf{v} \in \{0, 1\}^{\nu\kappa} \times \{0, 1\}^{\nu\kappa}$.

Twitness(pp, d): On input a data set R and a value \mathbf{d} , output \perp , if $\mathbf{d} \notin R$. Otherwise, set $\mathbf{d} = \mathbf{d}_j$ for some $j \in \{0, \dots, N-1\}$ output a witness $\mathbf{w} = ((j_1, \dots, j_1), (\mathbf{u}_{j_1, \dots, j_{1-1}, \bar{j}_1}, \dots, \mathbf{u}_{j_1, \bar{j}_2}, \mathbf{u}_{\bar{j}_1})) \in \{0, 1\}^1 \times \{0, 1\}^{1\nu\kappa}$; $\mathbf{u}_{j_1, \dots, j_{1-1}, \bar{j}_1}$ denotes the sibling-node of $\mathbf{u}_{j_1, \dots, j_1}$, and all the sibling-nodes are calculated by Tcalc(R).

TVerify(pp, n₀, d, w): On input root node \mathbf{n}_0 and a pair (\mathbf{d}, \mathbf{w}) , where \mathbf{d} is a \mathbf{w} is the “sibling-path” described as follows: $\mathbf{w} = ((j_1, \dots, j_1), (\mathbf{d}_{j_1}, \mathbf{f}_{j_1}), \dots, (\mathbf{d}_{j_1}, \mathbf{f}_{j_1}), (\mathbf{w}_1, \dots, \mathbf{w}_1)) \in \{0, 1\}^1 \times \{0, 1\}^{21} \times \{0, 1\}^{1\nu\kappa}$, where $(\mathbf{d}_{j_1}, \mathbf{f}_{j_1})$ are the corresponding randomnesses of vectors $\mathbf{u}_{j_1, \dots, j_1}, \dots, \mathbf{u}_{j_1}$. To verify the node \mathbf{d} using root node \mathbf{n}_0 , we first use the fact that $F_0(\mathbf{u}, \mathbf{v}) = \mathbf{n}_0 \Leftrightarrow \mathbf{A} \cdot \mathbf{u} + \mathbf{A}_0 \cdot \mathbf{v} = \mathbb{G} \cdot \mathbf{n}_0 \pmod{q}$. Taking gadget matrix \mathbb{G} , the root node \mathbf{n}_0 and the public parameters \mathbf{A}, \mathbf{A}_0 , we can calculate $\mathbf{z} = (\mathbf{u} || \mathbf{v})$, using Gaussian elimination technique for the equation $\tilde{\mathbf{A}} \cdot \mathbf{z} = \mathbf{n}_0 \pmod{q}$, where $\tilde{\mathbf{A}} = [\mathbf{A} | \mathbf{A}_0]$. The solution \mathbf{z} is split into two vectors, where the left-side of the vector is associated with \mathbf{u} and the right-side is associated with vector \mathbf{v} . Using these vectors, the algorithm calculates $(\mathbf{u}_{j_1, \dots, j_{1-1}, \bar{j}_1}, \dots, \mathbf{u}_{j_1, \bar{j}_2}, \mathbf{u}_{\bar{j}_1})$ and compares it with $(\mathbf{w}_1, \dots, \mathbf{w}_1)$ and computes in the same way the leave \mathbf{d} outputs 1, if the comparison is conform, otherwise it outputs 0.

Theorem 1. *The given Merkle-tree algorithm is secure in the sense of Definition 1, assuming the hardness of the $SIVP_{\hat{O}(n)}$ problem.*

Proof. Let \mathcal{B} be a PPT adversary against the SIS problem from Definition 1. Upon running TSetup(λ), algorithm \mathcal{B} receives $\bar{\mathbf{A}} \in \mathbb{Z}_q^{\nu \times \mu}$. It returns the string $(R = (\mathbf{d}_0, \dots, \mathbf{d}_{N-1}), \mathbf{d}^*, \mathbf{w}^*)$, where $\mathbf{d}^* \notin R$, TVerify($pp, \mathbf{n}_0^*, \mathbf{d}^*, \mathbf{w}^*$) = 1 and $\mathbf{n}_0^* = \text{Tcalc}(R)$.

It sets the witness as $\mathbf{w}^* = ((j_1, \dots, j_1), (\mathbf{d}_{j_1}, \mathbf{f}_{j_1}), \dots, (\mathbf{d}_{j_1}, \mathbf{f}_{j_1}), (\mathbf{w}_1, \dots, \mathbf{w}_1)) \in \{0, 1\}^1 \times \{0, 1\}^{1\nu\kappa}$ and let $j^* \in [0, N-1]$ with binary representation (j_1^*, \dots, j_1^*) . Set $\mathbf{u}_{j_1^*, \dots, j_1^*} = \mathbf{d}_{j^*}, \mathbf{u}_{j_1^*, \dots, j_{1-1}^*}, \dots, \mathbf{u}_{j_1^*, \bar{j}_2^*}, \mathbf{n}_0^*$ denoting the path from the leave \mathbf{d}_{j^*} to the root \mathbf{n}_0^* . Furthermore, compute $\mathbf{z} = (\mathbf{u} || \mathbf{v})$, using Gaussian elimination technique for the equation $\tilde{\mathbf{A}} \cdot \mathbf{z} = \mathbf{n}_0 \pmod{q}$, where $\tilde{\mathbf{A}} = [\mathbf{A} | \mathbf{A}_0]$ and using the values \mathbf{u}, \mathbf{v} compute the path $(\mathbf{u}_{j_1, \dots, j_{1-1}, \bar{j}_1}, \dots, \mathbf{u}_{j_1, \bar{j}_2}, \mathbf{u}_{\bar{j}_1})$ by running the algorithm TVerify. Comparing the two path, we can find the smallest index $\xi \in [l]$, such that $\mathbf{u}_{j_1^*, \dots, j_\xi^*} \neq \mathbf{u}_{\xi}^*$.

3.4 Zero-Knowledge AoK of a Generated Value

In this section we construct a zero-knowledge argument of knowledge system which allows a prover \mathcal{P} to provide a proof to the verifier \mathcal{V} , that \mathcal{P} knows a secret value that is the result of the Merkle-Damgård construction used in our Merkle-tree. In other words, in our zero-knowledge protocol the prover takes as input $\overline{\mathbf{A}} = [\mathbf{A} | \mathbf{A}_0 | \dots | \mathbf{A}_1] \in \mathbb{Z}_q^{\nu \times (1+2)^\mu}$, two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{Z}_q^\mu$ and convinces \mathcal{V} that \mathcal{P} herself is in possession of a value-witness pair (\mathbf{d}, \mathbf{w}) such that holds $\text{TVerify}(\mathbf{n}_0, \mathbf{d}, \mathbf{w}) = 1$. In the following definition we fix the relation $\mathcal{R}_{\text{calc}}$ of our Merkle-tree generator.

Definition 9. *The relation of Merkle-tree generator is given as $\mathcal{R}_{\text{calc}} = \{(\overline{\mathbf{A}}, \mathbf{u}, \mathbf{v}) \in \mathbb{Z}_q^{\nu \times 1\mu} \times \{0, 1\}^{\nu\kappa}; \mathbf{d} \in \{0, 1\}^{\nu\kappa}, \mathbf{w} \in \{0, 1\}^1 : \text{TVerify}(\mathbf{n}_0, \mathbf{d}, \mathbf{w}) = 1\}$.*

Before presenting our Zero-Knowledge (ZK) proof, we recall set $\mathbf{S}_{3\mu}$ consisting of all vectors in $\{-1, 0, 1\}^{3\mu}$, with exactly μ coordinates -1 , μ coordinates 0 and μ coordinates 1 . Furthermore, set \mathbf{S}_μ consists of all permutations of vectors in $\{-1, 0, 1\}^\mu$. Let F_π be the function which transforms a vector $\mathbf{v}' \in \mathbb{Z}_q^{3\mu}$ into $\pi(\mathbf{v}') \in \mathbf{S}_{3\mu}$. The ZK strategy relies on the following property:

Let $\text{Ext}(\mathbf{v})$ denote an extension of \mathbf{v} to a vector $\mathbf{v}' \in \{-1, 0, 1\}^{3\mu}$. For all vectors $\mathbf{v} \in \{-1, 0, 1\}^\mu$ with $\lambda^{(-1)}$ coordinates -1 , $\lambda^{(0)}$ coordinates 0 and $\lambda^{(1)}$ coordinates 1 we have the following equation: $\mathbf{v}'_i \in \text{Ext}(\mathbf{v}_i) \wedge \mathbf{v}_i \in \{-1, 0, 1\}^\mu \Leftrightarrow \text{Ext}(\pi(\mathbf{v}_i)) = F_\pi(\mathbf{v}') \wedge \pi(\mathbf{v}') \in \{-1, 0, 1\}^{3\mu}$. Furthermore we show how to use Merkle-Damgård compress function in the Merkle tree during our zero-knowledge protocol.

Preparation steps. Let $(\overline{\mathbf{A}}, \mathbf{n}_0, \mathbf{d}, \mathbf{w}) \in \mathcal{R}_{\text{calc}}$, such that $w = ((j_1, \dots, j_1), (\mathbf{d}_{j_1}, \mathbf{f}_{j_1}), \dots, (\mathbf{d}_{j_1}, \mathbf{f}_{j_1}), (\mathbf{w}_1, \dots, \mathbf{w}_1)) \in \{0, 1\}^1 \times \{0, 1\}^{21} \times \{0, 1\}^{1\nu\kappa}$, and let $(\mathbf{u}_{j_1, \dots, j_{i-1}, \bar{j}_1}, \dots, \mathbf{u}_{j_1, \bar{j}_2}, \mathbf{u}_{\bar{j}_1})$ be the path generated by the $\text{TVerify}(\mathbf{u}, \mathbf{d}, w)$ algorithm, where each $\mathbf{u}_{j_1, \dots, j_{i-1}, \bar{j}_1}$, $i \in [2, 1]$ is described by the Merkle-Damgård compression function, i.e.

$$\mathbf{u}_{j_1, \dots, j_{i-1}, \bar{j}_1} = F_i \left(\mathbf{u}_{j_1, \dots, j_{i-2}, \bar{j}_{i-1}}, F_{i-1}, \mathbf{u}_{j_1, \dots, j_{i-3}, \bar{j}_{i-2}} \right) \tag{1}$$

Using the gadget matrix G , which was recalled in Sect. 3.2, the Eq. (1) can be represented as:

$$\begin{aligned} & F_i \left(\mathbf{u}_{j_1, \dots, j_{i-2}, \bar{j}_{i-1}}, F_{i-1}, \mathbf{u}_{j_1, \dots, j_{i-3}, \bar{j}_{i-2}} \right) = \mathbf{A}_i \mathbf{d}_i, \mathbf{u}_{j_1, \dots, j_{i-2}, \bar{j}_{i-1}} \mathbf{u}_{j_1, \dots, j_{i-2}, \bar{j}_{i-1}} \\ & + \mathbf{f}_i, \mathbf{u}_{j_1, \dots, j_{i-2}, \bar{j}_{i-1}} F_{i-1}, \mathbf{u}_{j_1, \dots, j_{i-3}, \bar{j}_{i-2}} = \mathbf{A}_i \mathbf{d}_i, \mathbf{u}_{j_1, \dots, j_{i-2}, \bar{j}_{i-1}} \mathbf{u}_{j_1, \dots, j_{i-2}, \bar{j}_{i-1}} \\ & + \prod_{k=1}^i \mathbf{f}_k, \mathbf{u}_{j_1, \dots, j_{k-2}, \bar{j}_{k-1}} \mathbf{A} \mathbf{u} + \prod_{k=1}^i \mathbf{f}_k, \mathbf{u}_{j_1, \dots, j_{k-2}, \bar{j}_{k-1}} \mathbf{A}_0 \mathbf{u} + \mathbf{f}_i, \mathbf{u}_{j_1, \dots, j_{k-2}, \bar{j}_{k-1}} \\ & \cdot \sum_{\ell=1}^{i-1} \left(\prod_{k=1}^{i-1} \mathbf{d}_k, \mathbf{u}_{j_1, \dots, j_{k-2}, \bar{j}_{k-1}} \right) \mathbf{A}_{\ell-1} \mathbf{u}_{j_1, \dots, j_{\ell-2}, \bar{j}_{\ell-1}} = \mathbb{G} \cdot \mathbf{u}_{j_1, \dots, j_{i-1}, \bar{j}_1} \pmod q \end{aligned}$$

To provide zero-knowledgeness, we construct an argument system, where \mathcal{P} convinces \mathcal{V} in zero-knowledge, that \mathcal{P} knows $j_1, \dots, j_i \in \{0, 1\}^l$ and $(\mathbf{u}_{j_1, \dots, j_{i-1}, \bar{j}_1}, \dots, \mathbf{u}_{j_1, \bar{j}_2}, \mathbf{u}_{\bar{j}_1}^-)$ with $(\mathbf{w}_1, \dots, \mathbf{w}_1)$, s.t.

$$\begin{aligned}
 & \mathbf{A}_i \mathbf{d}_{i, \text{Ext}(\mathbf{u}_{j_1, \dots, j_{i-2}, \bar{j}_{i-1}})} \text{Ext}(\mathbf{u}_{j_1, \dots, j_{i-2}, \bar{j}_{i-1}}) \\
 & + \prod_{k=1}^i \mathbf{f}_{k, \text{Ext}(\mathbf{u}_{j_1, \dots, j_{k-2}, \bar{j}_{k-1}})} \mathbf{A} \text{Ext}(\mathbf{u}) + \prod_{k=1}^i \mathbf{f}_{k, \text{Ext}(\cdot)} \mathbf{A}_0 \text{Ext}(\mathbf{v}) \\
 & + \mathbf{f}_{i, \text{Ext}(\mathbf{u}_{j_1, \dots, j_{i-2}, \bar{j}_{i-1}})} \sum_{\ell=1}^{i-1} \left(\prod_{k=1}^{\ell} \mathbf{d}_{k, \text{Ext}(\mathbf{u}_{j_1, \dots, j_{k-2}, \bar{j}_{k-1}})} \right) \\
 & \cdot \mathbf{A}_{\ell-1} \text{Ext}(\mathbf{u}_{j_1, \dots, j_{\ell-2}, \bar{j}_{\ell-1}}) = \mathbb{G} \cdot \mathbf{u}_{j_1, \dots, j_{i-1}, \bar{j}_i} \pmod{q}
 \end{aligned} \tag{2}$$

In order to fulfill the zero-knowledgeness, we apply the extension technique, recalled in Sect. 3.2.:

- (1) Extend the matrix $\bar{\mathbf{A}} = [\mathbf{A} | \mathbf{A}_0 | \dots | \mathbf{A}_1]$ to matrix $\bar{\mathbf{A}}' = [\mathbf{A} | 0^{\nu \times \nu \kappa} \mathbf{A}_0 | 0^{\nu \times \nu \kappa} \dots | \mathbf{A}_i | 0^{\nu \times \nu \kappa}] \in \mathbb{Z}_q^{\nu \times (2+i)3\mu}$.
- (2) Extend the gadget matrix \mathbb{G} to matrix $\mathbb{G}' = [G | 0^{\nu \times \nu \mu}] \in \mathbb{Z}_q^{\nu \times \mu}$.
- (3) Extend the vectors $(\mathbf{u}_{j_1, \dots, j_{i-1}, \bar{j}_1}, \dots, \mathbf{u}_{j_1, \bar{j}_2}, \mathbf{u}_{\bar{j}_1}^-)$ and \mathbf{v} to the corresponding vectors $(\mathbf{u}'_{j_1, \dots, j_{i-1}, \bar{j}_1}, \dots, \mathbf{u}'_{j_1, \bar{j}_2}, \mathbf{u}'_{\bar{j}_1})$ and \mathbf{v}' , all elements in $S_{3\mu}^{\nu \kappa}$ using Hamming weight extension technique.

Let $\mathbf{y}_i = \text{Ext}(\mathbf{u}'_{j_1, \dots, j_{i-1}, \bar{j}_i})$ and $\mathbf{z} = \text{Ext}(\mathbf{v}')$. Using these extensions and those of matrices, defined above, can be applied to the Eq. 2, replacing each matrix and each vector by its extension.

3.5 The Underlying Interactive Protocol

Using the prepared information above, we provide our interactive zero-knowledge protocol. The public parameters are given by ν, q, κ, μ, l .

Common inputs: $\bar{\mathbf{A}} = [\mathbf{A} | \mathbf{A}_0 | \dots | \mathbf{A}_1]$ and \mathbf{u}, \mathbf{v} . \mathcal{P} and \mathcal{V} extend $\bar{\mathbf{A}}$ to $\bar{\mathbf{A}}'$. **Prover's input:** Given by $\mathbf{w} = ((j_1, \dots, j_1), (\mathbf{d}_{j_1}, \mathbf{f}_{j_1}), \dots, (\mathbf{d}_{j_1}, \mathbf{f}_{j_1}), (\mathbf{w}'_1, \dots, \mathbf{w}'_1))$, and $(\mathbf{u}'_{j_1, \dots, j_{i-1}, \bar{j}_1}, \dots, \mathbf{u}'_{j_1, \bar{j}_2}, \mathbf{u}_{\bar{j}_1}^-)'$, $(\mathbf{y}_1, \dots, \mathbf{y}_1)$, \mathbf{v}', \mathbf{z} . \mathcal{P} 's goal is to prove in zero-knowledge that $(\mathbf{u}'_{j_1, \dots, j_{i-1}, \bar{j}_i}, \mathbf{w}'_i) \in (S_{3\mu}^{\nu \kappa})^2$, $\mathbf{y}_i = \text{Ext}(\mathbf{u}'_{j_1, \dots, j_{i-1}, \bar{j}_i})$, $\mathbf{z} = \text{Ext}(\mathbf{v}')$.

- (1) To prove zero-knowledgeness of our protocol, the prover picks randomly $\pi_i, \psi_i \leftarrow_{\mathcal{R}} S_{\mu}$ and convinces \mathcal{V} by the following equations:

$$\pi(\mathbf{u}'_{j_1, \dots, j_{i-1}, \bar{j}_i}) \in S_{3\mu}^{\nu \kappa} \wedge \text{Ext}(\pi(\mathbf{u}'_{j_1, \dots, j_{i-1}, \bar{j}_i})) = F_{\pi}(\mathbf{y}_i) \tag{3}$$

$$\pi(\mathbf{v}') \in S_{3\mu}^{\nu \kappa} \wedge \text{Ext}(\pi(\mathbf{v}')) = F_{\pi}(\mathbf{z}) \tag{4}$$

- (2) To prove zero-knowledgeness of all 1 equations in (2) using extended vectors and matrices, \mathcal{P} samples randomly masking vectors $\mathbf{r}_{u_1}, \dots, \mathbf{r}_{u_{i-1}} \leftarrow \mathbb{Z}_q^{\mu}$,

$\mathbf{r}_u, \mathbf{r}_v \leftarrow \mathbb{Z}_q^\mu$ and random values $\tilde{d}_1, \dots, \tilde{d}_l, \tilde{f}_1, \dots, \tilde{f}_i \in \mathbb{Z}_q$ and shows to the verifier that holds the following equation

$$\begin{aligned} & \mathbf{A}'_i \tilde{\mathbf{d}}_{i, \mathbf{y}_{i-1}} (\mathbf{y}_{i-1} + \mathbf{r}_{u_{i-1}}) + \prod_{k=1}^i \tilde{f}_{k, \mathbf{y}_{k-1}} \mathbf{A}' \text{Ext}((\mathbf{u}') + \mathbf{r}_u) + \prod_{k=1}^i \tilde{f}_{k, \mathbf{y}_{k-1}} \mathbf{A}'_0 \text{Ext}((\mathbf{v}') + \mathbf{r}_v) \\ & + \tilde{f}_{i, \mathbf{y}_{k-1}} \sum_{\ell=1}^{i-1} \left(\prod_{k=1}^{i-1} \tilde{\mathbf{d}}_{k, \mathbf{y}_{k-1}} \right) \mathbf{A}'_{\ell-1} (\mathbf{y}_{\ell-1} + \mathbf{r}_{u_{\ell-1}}) = \mathbb{G} \cdot (\mathbf{y}_i + \mathbf{r}_{u_i}) \pmod{q}. \end{aligned} \quad (5)$$

Commitment: \mathcal{P} samples ρ_1, ρ_2, ρ_3 and sends to \mathcal{V} the following commitment $\text{COM} = (\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3)$:

$$\begin{aligned} \mathbf{C}_1 &= \text{COM} \left(\{ \pi_i, \psi_i \}_{i \in [1]} : \mathbf{A}'_i \tilde{\mathbf{d}}_{i, \mathbf{y}_{i-1}} \mathbf{r}_{u_{i-1}} + \prod_{k=1}^i \tilde{f}_{k, \mathbf{y}_{k-1}} \mathbf{A}' \mathbf{r}_u + \prod_{k=1}^i \tilde{f}_{k, \mathbf{y}_{k-1}} \mathbf{A}'_0 \mathbf{r}_v \right. \\ & \quad \left. + \tilde{f}_{i, \mathbf{y}_{k-1}} \sum_{\ell=1}^{i-1} \left(\prod_{k=1}^{i-1} \tilde{\mathbf{d}}_{k, \mathbf{y}_{k-1}} \right) \mathbf{A}'_{\ell-1} \mathbf{r}_{u_{\ell-1}} - \mathbb{G} \cdot \mathbf{r}_{u_i}; \rho_1 \right)_{i \in [1-1]} \\ \mathbf{C}_2 &= \text{COM}(\{ \pi_i(\mathbf{r}_{u_i}) \}_{i \in [1-1]}; \mathbf{F}_{\pi_i}(\mathbf{y}_i), \mathbf{F}_{\psi}(\mathbf{z}); \rho_2) \\ \mathbf{C}_3 &= \text{COM}(\{ \pi_i(\mathbf{r}_{u_i} + \mathbf{y}_i) \}_{i \in [1-1]}; \mathbf{F}_{\pi_i}(\mathbf{y}_i + \mathbf{r}_{u_i}), \mathbf{F}_{\psi}(\mathbf{z} + \mathbf{r}_v); \rho_3). \end{aligned}$$

Challenge: \mathcal{V} sends a challenge $\text{Ch} \leftarrow_{\mathbf{r}} \{1, 2, 3\}$ to \mathcal{P} .

Response: Prover sends a response corresponding to the challenge Ch .

If $\text{Ch} = 1$: for each $i \in [1-1]$ set $\mathbf{p}_{u'_i} = \pi_i(\mathbf{r}_{u'_i})$; for each $i \in [1]$: $\mathbf{s}_{u'_i} = \pi_i(\mathbf{u}'_i)$, $\mathbf{s}_{u'} = \psi(\mathbf{u}')$, $\mathbf{s}_{v'} = \psi(\mathbf{v}')$, $\mathbf{s}_{y_i} = \mathbf{F}_{\pi_i}(\mathbf{r}_{u_i})$, $\mathbf{s}_v = \mathbf{F}_{\psi}(\mathbf{r}_v)$. The response is given by the following tuple $\text{RSP} := \{ \{ \mathbf{p}_{u'_i} \}_{i \in [1-1]}, \{ \mathbf{s}_{u'_i}, \mathbf{p}_{y_i} \}_{i \in [1]}, \mathbf{s}_{u'}, \mathbf{s}_{v'}, \mathbf{s}_v, \rho_2, \rho_3 \}$.

If $\text{Ch} = 2$: for each $i \in [1-1]$ set $\mathbf{a}_{u'_i} = (\mathbf{r}_{u_i} + \mathbf{u}_i)$, $\mathbf{a}_{u'} = \mathbf{u}' + \mathbf{r}_{u'}$, $\mathbf{a}_{v'} = \mathbf{v}' + \mathbf{r}_{v'}$, $\hat{\pi}_i = \pi_i$, $\hat{\psi}_i = \psi_i$. The response is $\text{RSP} := \{ \{ \mathbf{a}_{u'_i} \}_{i \in [1-1]}, \mathbf{a}_{u'}, \mathbf{a}_{v'}, \hat{\pi}_i, \hat{\psi}_i, \rho_1, \rho_3 \}$.

If $\text{Ch} = 3$: for each $i \in [1-1]$ set $\mathbf{t}_{u_i} = \mathbf{r}_{u_i}$, $\mathbf{t}_{y_i} = \mathbf{r}_{y_i}$, $\mathbf{t}_{u'_i} = \mathbf{r}_{u'_i}$, $\mathbf{t}_{u'} = \mathbf{r}_{u'}$, $\mathbf{t}_{v'} = \mathbf{r}_{v'}$, $\tilde{\pi}_i = \pi_i$, $\tilde{\psi}_i = \psi_i$.

The response is given by $\text{RSP} := \{ \{ \mathbf{t}_{u_i} \}_{i \in [1-1]}, \{ \mathbf{t}_{u'_i}, \mathbf{t}_{y_i} \}_{i \in [1]}, \mathbf{t}_{u'}, \mathbf{t}_{v'}, \tilde{\psi}_i, \tilde{\pi}_i, \rho_1, \rho_2 \}$.

Verification: Distinguish between three cases.

- (1) **Case $\text{Ch} = 1$:** Set RSP as above and check that $\mathbf{s}_{u'_i}, \mathbf{s}_{u'} \in \mathbb{S}_{3\mu}^{\nu\kappa}$, where $i \in [1-1]$. Let $\tilde{\mathbf{s}}_{u'_i} = \text{Ext}(\mathbf{s}_{u'_i})$ and $\tilde{\mathbf{s}}_{y_i} = \text{Ext}(\mathbf{s}_{y_i})$, $\tilde{\mathbf{s}}_{u'} = \text{Ext}(\mathbf{s}_{u'})$, $\tilde{\mathbf{s}}_{v'} = \text{Ext}(\mathbf{s}_{v'})$ and check

$$\begin{aligned} \mathbf{C}_2 &= \text{COM}(\{ \mathbf{p}_{u_i} \}_{i \in [1-1]}, \{ \mathbf{s}_v, \mathbf{s}_{y_i} \}_{i \in [1]}; \rho_2) \\ \mathbf{C}_3 &= \text{COM}(\{ \mathbf{s}_{u'_i} + \mathbf{p}_{u'_i} \}_{i \in [1-1]}, \{ \tilde{\mathbf{s}}_{u'_i} + \mathbf{s}_{y_i}; \mathbf{s}_v + \tilde{\mathbf{s}}_{v'}; \mathbf{s}_u + \tilde{\mathbf{s}}_{u'} \}_{i \in [1]}; \rho_3). \end{aligned}$$

(2) **Case $Ch = 2$:** Set RSP as above and check that:

$$\begin{aligned} \mathcal{C}_2 &= \text{COM} \left(\left\{ \hat{\pi}_i, \hat{\psi}_i \right\}_{i \in [1]}; \mathbf{A}'_i \tilde{\mathbf{d}}_{i, \mathbf{a}'_{u'_i}} + \prod_{k=1}^i \tilde{\mathbf{f}}_{k, \mathbf{a}'_{u'_{k-1}}} \mathbf{A}'_{u'} + \prod_{k=1}^i \tilde{\mathbf{f}}_{k, \mathbf{a}'_{u'_{k-1}}} \mathbf{A}'_0 \mathbf{a}_{v'} \right. \\ &\quad \left. + \tilde{\mathbf{f}}_{i, \mathbf{a}'_{u'_{i-1}}} \sum_{\ell=1}^{i-1} \left(\prod_{k=1}^{\ell-1} \tilde{\mathbf{d}}_{k, \mathbf{a}'_{u'_{k-1}}} \right) \mathbf{A}'_{\ell-1} \mathbf{r}_{u_{\ell-1}} - \mathbb{G} \cdot \mathbf{a}_{u_i}; \rho_1 \right) \\ \mathcal{C}_3 &= \text{COM}(\{ \hat{\pi}_i(\mathbf{a}'_{u'_i}) \}_{i \in [1-1]}; \mathbf{F}_{\hat{\pi}_i}(\mathbf{a}'_{u'_i}), \mathbf{F}_{\hat{\psi}_i}(\mathbf{a}_{v'}); \rho_3) \end{aligned}$$

(3) **Case $Ch = 3$:** Set RSP as above and check that:

$$\begin{aligned} \mathcal{C}_1 &= \text{COM} \left(\left\{ \hat{\pi}_i, \hat{\psi}_i \right\}_{i \in [1]} : \mathbf{A}'_i \tilde{\mathbf{d}}_{i, \mathbf{u}_{i-1}} \mathbf{t}_{u_{i-1}} + \prod_{k=1}^i \tilde{\mathbf{f}}_{k, \mathbf{y}_{k-1}} \mathbf{A}'_{t_{u'}} + \prod_{k=1}^i \tilde{\mathbf{f}}_{k, \mathbf{y}_{k-1}} \mathbf{A}'_0 \mathbf{t}_{v'} \right. \\ &\quad \left. + \tilde{\mathbf{f}}_{i, \mathbf{y}_{i-1}} \sum_{\ell=1}^{i-1} \left(\prod_{k=1}^{\ell-1} \tilde{\mathbf{d}}_{k, \mathbf{y}_{k-1}} \right) \mathbf{A}'_{\ell-1} \mathbf{t}_{y_{\ell-1}} - \mathbb{G} \cdot \mathbf{t}_{u_i}; \rho_1 \right) \\ \mathcal{C}_2 &= \text{COM}(\{ \hat{\pi}_i(\mathbf{t}_{u_i}) \}_{i \in [1-1]}; \mathbf{F}_{\hat{\pi}_i}(\mathbf{t}_{y_i}), \mathbf{F}_{\hat{\psi}_i}(\mathbf{t}_{v'}); \rho_2). \end{aligned}$$

3.6 Analysis of the Interactive Protocol

We summarize the properties of our zero-knowledge interactive protocol in the following theorem, whose proof we skip in this version of the paper due to limited number of pages and refer to the full version of the paper.

Theorem 2. *Let COM denote a statistically hiding and computationally binding commitment scheme. Then, our ZK protocol described above is complete and sound. Furthermore, it fulfills the property of a zero-knowledge argument of knowledge.*

4 Attribute-Based Group Signature on Lattices

Before we can proceed with the construction of our attribute-based group signature from lattices (ABGSL), we need to define a significant building block which will be used in our ABGSL scheme. We present a signature of N message blocks from lattices, whose construction is motivated by the signature from lattices introduced by Libert et al. [25]

Definition 10 (Digital Signature from Lattices).

Setup($1^\lambda, 1^N$): On input a security parameter 1^λ , performs the following steps:

(1) Select a security dimension $\nu > \Omega(\lambda)$, a dimension of the lattice base is $\mu > 2\nu \log q$ and a discrete Gaussian parameter s , an integer $l = \Theta(\lambda)$.

(2) Pick for each $\mathbf{i} \in [1]$ a random matrix \mathbf{B}_i , run **TrapGen** algorithm on input 1^λ and output l uniform random matrix $\mathbf{A}_i \in \mathbb{Z}_q^{\nu \times \mu}$ with the corresponding basis $\mathbf{T}_{\mathbf{A}, \mathbf{i}} \subseteq \Lambda^\perp(\mathbf{A}_i)$, where $\mathbf{i} \in \{0, \dots, 1\}$.

- (3) Select random $\nu \times \mu$ -matrix $\mathbf{A} \in \mathbb{Z}_q^{\nu \times \mu}$.
- (4) Select uniform random ν -vectors $\mathbf{u}, \mathbf{v} \in \mathbb{Z}_q^\nu$.
- (5) Output public parameters $\mathbf{pp} = (\mathbf{A}, \{\mathbf{A}_i\}_{i \in [1]}, \mathbf{u}, \mathbf{v})$ and $\mathbf{sk} = \{\mathbf{T}_{\mathbf{A},i}\}_{i \in [1]}$.

Sign(pp, sk, M): On input public parameters \mathbf{pp} , secret key \mathbf{sk} and an μ -bit message $\mathbf{M} = (m_1, \dots, m_\mu) \in \{0, 1\}$ sign the message according to the following steps:

- (1) Choose random string $\tau \leftarrow_{\mathbf{r}} \{0, 1\}^1$. Run $\text{ExtBasis}(\mathbf{T}_{\mathbf{A}})$ to extent the basis $\mathbf{T}_{\mathbf{A}}$ to the new basis \mathbf{T}_τ belonging to $\mathbf{A}_\tau = [\mathbf{A} | \mathbf{A}_0 + \sum_{j=1}^1 \tau_j \mathbf{A}_j] \in \mathbb{Z}_q^{\nu \times 2\mu}$.
- (2) Taking the values \mathbf{u}, \mathbf{v} compute the value $\mathbf{c}_{\mathbf{M}}$ as a Merkle-Damgård hash function as presented in the Sect. 3.2., where $\mathbf{c}_{\mathbf{M}} = \mathbf{A}_1 \mathbf{d}_1 \mathbf{v} + \mathbf{f}_1 \mathbf{F}_{1-\mu} \mathbf{M}$. Pick $\mathbf{D} \leftarrow_{\mathbf{r}} \mathbb{Z}_q^{\nu \times \mu}$, define $\mathbf{u}_{\mathbf{M}} = \mathbf{u} + \mathbf{D} \cdot \text{bin}(\mathbf{c}_{\mathbf{M}}) \in \mathbb{Z}_q^\nu$.
- (3) Using \mathbf{T}_τ sample a vector $\mathbf{t} \leftarrow_{\mathbf{r}} \mathcal{D}_{\mathbb{Z}^{2\mu}, \mathbf{s}}$ and run $\text{SamplePre}(\mathbf{T}_\tau, \mathbf{t}, \mathbf{u})$. The output is a vector $\mathbf{v} \in \mathbb{Z}^{2\mu}$.

Output the signature $\sigma = (\tau, \mathbf{v}, \mathbf{t})$.

Verify(pp, M, σ): On input public parameters, a message $M = (m_1, \dots, m_\mu)$ and a signature $\sigma = (\tau, \mathbf{v}, \mathbf{t})$ return 1 if $\mathbf{A}_\tau \cdot \mathbf{v} = \mathbf{u} + \mathbf{D} \cdot \text{bin}(\mathbf{c}_{\mathbf{M}}) \pmod q$.

The lattice-based signature scheme defined above is secure under chosen-message attacks assuming that the SIS problem is hard. The proof of security of this scheme is to be skipped due to the page limit. But, we refer to the extended version of the paper for further details and proofs.

Construction. In this section we provide the first attribute-based group signature based on lattices using our zero-knowledge protocol from Sect. 3. The scheme contains certain building blocks such as digital signature from Definition 11 and a lattice-based encryption scheme introduced by Regev [35].

Setup($1^\lambda, 1^l, 1^N$): On input a security parameter 1^λ , an attribute bound l and a group limit of N members the algorithm performs the following steps:

- (1) Select a security dimension $\nu > \Omega(\lambda)$, a dimension of the lattice base is $\mu > 2\nu \log q$ and a discrete Gaussian parameter \mathbf{s} .
- (2) Pick for each $i \in [1]$ a random matrix \mathbf{B}_i , run TrapGen algorithm on input 1^λ and output l uniform random matrix $\mathbf{A}_i \in \mathbb{Z}_q^{\nu \times \mu}$ with the corresponding basis $\mathbf{T}_{\mathbf{A},i} \subseteq \Lambda^\perp(\mathbf{A}_i)$, where $i \in \{0, \dots, l\}$.
- (3) Select random $\nu \times \mu$ -matrix $\mathbf{A} \in \mathbb{Z}_q^{\nu \times \mu}$ and ν -vectors $\mathbf{u}, \mathbf{v} \in \mathbb{Z}_q^\nu$.
- (4) Define a hash function $\mathbf{H} : \{0, 1\}^* \rightarrow \mathbb{Z}_q^{\nu \times 2\mu}$ which maps a certain bit string to an integer matrix of size $\nu \times 2\mu$.
- (5) Output public parameters $\mathbf{pp} = (\mathbf{A}, \{\mathbf{A}_i\}_{i \in [1]}, \mathbf{u}, \mathbf{v})$ and $\mathbf{msk} = \{\mathbf{T}_{\mathbf{A},i}\}_{i \in [1]}$.
- (6) Generate public and secret key of Regev's encryption scheme as follows: Sample a random matrix $\mathbf{B} \leftarrow_{\mathbf{r}} \mathbb{Z}_q^{\nu \times \mu}$ and the corresponding short basis $\mathbf{T}_{\mathbf{B}}$ by running the TrapGen algorithm on input security parameter. It outputs public and secret key of the group manager who runs opening procedure to identify malicious signer. The keys are set equal to $\mathbf{gmsk} = \mathbf{T}_{\mathbf{B}}$ and $\mathbf{gmpk} = \mathbf{B}$.

ABGKeyGen($\mathbf{pp}, \mathbf{msk}, \mathbb{A}_i, \tilde{\mathbb{A}}$): On input public parameters \mathbf{pp} , a master secret key $\mathbf{msk} = \{\mathbf{T}_{\mathbf{A},i}\}_{i \in [1]}$ and a set of attributes \mathbb{A}_i as well as the universe of attributes $\tilde{\mathbb{A}} = \{\mathbf{a}_0, \dots, \mathbf{a}_{N-1}\}$, where $N = 2^l$ the algorithm extracts attribute-based secret keys via the following steps:

(1) Generate the tree from the attribute universe \mathbb{A} , i.e. associate each attribute \mathbf{a}_j with a leaf of the tree \mathbf{d}_j , where $j \in [0, N - 1]$. To do so, we assign to each attribute a binary string in $\{0, 1\}^\nu$ by computing $\text{bin}(\mathbf{A}_i \cdot \mathbf{u}_j \pmod{\mathbf{q}}) = \mathbf{d}_j$.

(2) Let $\mathbf{R} = (\mathbf{d}_0, \dots, \mathbf{d}_{N-1})$ and run $\text{Tcalc}(\mathbf{R})$ to generate the complete tree and the values \mathbf{u}, \mathbf{v} for the hash function of the root node $\mathbf{F}_0(\mathbf{u}, \mathbf{v})$.

(3) Use the set \mathbf{R} and one of the tree leaves \mathbf{d}_j as input of TWitness algorithm to generate the witness of the zero-knowledge proof: $\mathbf{w} =$

$$((j_1, \dots, j_l), (\mathbf{d}_{j_1}, \mathbf{f}_{j_1}), \dots, (\mathbf{d}_{j_l}, \mathbf{f}_{j_l}), (\mathbf{w}_1, \dots, \mathbf{w}_l)) \in \{0, 1\}^l \times \{0, 1\}^{2l} \times \{0, 1\}^{l\nu\kappa},$$

(4) The key issuing authority provides valid credentials for each user i with k distinct attributes $\mathbf{u}_j^{(i)}$, $j \in [k]$, $i \in [N]$. To do so, it first computes a sum of the different attributes and sets $\mathbf{u}^{(i)} = \sum_{j=1}^k \mathbf{u}_j^{(i)}$. It samples values \mathbf{z}_i by running the algorithm $\text{SamplePre}(\mathbf{A}_i, \mathbf{q}, \mathbf{T}_{\mathbf{A},i}, \mathbf{s}, \mathbf{u}^{(i)})$.

(5) The secret key of user i is given as a set of values $\mathbf{sk}_{\mathbb{A}_i} = (\mathbf{z}_i, \mathbf{w}^{(i)}, \mathbf{d}^{(i)})$, where $\mathbf{d}^{(i)} = (\mathbf{d}_{j_1}^{(i)}, \dots, \mathbf{d}_{j_k}^{(i)})$, describes a set of attributes, \mathbb{A}_i of an user i . The group public key is given as $\mathbf{gpk} = \mathbf{pp}$.

(6) It generates the verification key of the user \mathbf{vk}_i as follows: Choose random string $\tau \leftarrow_{\mathbf{r}} \{0, 1\}^l$. Run $\text{ExtBasis}(\sum_{i=1}^l \mathbf{T}_{\mathbf{A}_i} = \mathbf{T}_{\mathbf{A}'})$ to extent $\mathbf{T}_{\mathbf{A}}$ to the new

basis \mathbf{T}_τ belonging to the extended matrix $\mathbf{A}_\tau = [\mathbf{A} | \mathbf{A}_0 + \sum_{j=1}^l \tau_j \mathbf{A}_j] \in \mathbb{Z}_q^{\nu \times 2\mu}$.

Pick randomly a short vector $\mathbf{y}_i \xleftarrow{\mathbf{r}} \mathcal{D}_{\mathbb{Z}^{2\mu}, \mathbf{s}}$. Compute $\mathbf{A}_\tau \cdot \mathbf{y}_i = \mathbf{v}_i$ representing the verification key \mathbf{vk}_i .

(7) The algorithm also generates the secret and public key for the key issuing authority, by running the Setup algorithm of the signature scheme from Definition 11.

Let $\mathbf{pk}_{\text{kia}} = (\mathbf{A}, \{\mathbf{A}_i\}_{i \in [1]}, \mathbf{u}, \mathbf{v})$ and a secret key $\mathbf{sk}_{\text{kia}} = \mathbf{T}_{\mathbf{A}}$.

$\langle \text{Join}(\text{param}, \mathbf{gpk}, \mathbf{pk}_i, \mathbf{sk}_{\mathbb{A}_i}) \rangle, \langle \text{Issue}(\text{param}, \mathbf{pk}_i, \mathbf{ik}) \rangle$: This interactive protocol is initiated by the user i , who takes its verification key $\mathbf{vk}_i := \mathbf{v}_i$ and computes a digital signature as defined in Definition 11, namely $\mathbf{sig}_i(\mathbf{v}_i)$ and the key issuing authority (KIA). The earlier sends its signature $\mathbf{sig}_i(\mathbf{v}_i)$ to KIA. Key issuing authority checks whether the verification key was already used by previous users or not. Then KIA generates a certificate as follows:

(1) First KIA checks the validity of \mathbf{sig}_i and that the corresponding verification key \mathbf{v}_i was not previously used by another user from the group.

(2) It chooses a new identifier represented by a bit-string of size l , i.e. $\mathbf{id}_i \in \{0, 1\}^l$.

It uses its secret and public keys to generate $\mathbf{A}_{\text{id}_i} = [\mathbf{A} | \mathbf{A}_0 + \sum_{j=1}^1 \text{id}_i[j] \mathbf{A}_j] \in \mathbb{Z}_q^{\nu \times 2\mu}$

(3) It runs $\text{ExtBasis}(\mathbf{A}_{\text{id}_i}, \mathbf{T}_A)$ to get an extended basis $\mathbf{T}_{A, \text{id}}$ of lattice $\Lambda_q^\perp(\mathbf{A}_{\text{id}_i})$.

(4) KIA samples a vector $\mathbf{t}_i \leftarrow_r \mathcal{D}_{\mathbb{Z}^{2\mu}, s}$ and runs the algorithm $\text{SamplePre}(\mathbf{T}_{\text{id}_i}, \mathbf{t}_i, \mathbf{u})$. The output is: $\mathbf{s}_i = \mathbf{A}_{\text{id}_i} \cdot \mathbf{t}_i = [\mathbf{A} | \mathbf{A}_0 + \sum_{j=1}^1 \text{id}_i[j] \mathbf{A}_j] \cdot \mathbf{t}_i \in \mathbb{Z}_q^{2\mu}$. Pick random matrices $\mathbf{D}, \mathbf{D}_1, \mathbf{D}_2 \leftarrow_r \mathbb{Z}_q^{3(\nu \times \mu)}$, then the equation above is equal to the following: $\mathbf{u} + \mathbf{D} \cdot \text{bin}(\mathbf{D}_1 \cdot \text{bin}(\mathbf{v}_i) + \mathbf{D}_2 \cdot \mathbf{t}_i) \pmod q$.

(5) KIA outputs a certificate $(\text{id}_i, \mathbf{t}_i, \mathbf{s}_i)$.

ABGSign(param, $\text{sk}_{A_i}, m, \Gamma$): On input user's attribute-based secret key $\text{sk}_{A_i} = (\mathbf{z}_i, \mathbf{w}^{(i)}, \mathbf{d}^{(i)})$, where $\mathbf{w} = ((j_1, \dots, j_1), (\mathbf{d}_{j_1}, \mathbf{f}_{j_1}), \dots, (\mathbf{d}_{j_1}, \mathbf{f}_{j_1}), (\mathbf{w}_1, \dots, \mathbf{w}_1))$, a message m and a predicate Γ the algorithm generates a signature of the message as described in the five steps presented below. According the Bellare et al. model [4] we first involve the one-time signature (OTS) scheme to sign the message. OTS is modeled as random oracle and consists of three algorithms ($\text{Setup}, \text{Sign}, \text{Verify}$). The signing proceeds as follows:

(1) The user runs $\text{Setup}(\lambda)$ algorithm of the One-Time-Signature (OTS) scheme to generate the verification and secret key $(\text{vk}_{\text{ots}}, \text{sk}_{\text{ots}})$. Compute OTS signature of m : $\sigma_{\text{ots}}(\text{sk}_{\text{ots}}, m)$.

(2) The user first hashes the verification and defines a matrix $\mathbf{F} = \text{H}(\text{vk}_{\text{ots}}) \in \mathbb{Z}_q^{1 \times \mu}$.

(3) Taking predicate Γ , convert it to a linear span program matrix $\mathbf{W} \in \mathbb{Z}^{1 \times \mu}$, where each row \mathbf{i} is assigned to the binary attribute with index $\mathbf{i} \in [1]$. Each column $\mathbf{j} \in [1, \mu]$ represents a function of the predicate Γ .

(4) Using encryption's public key $\text{pk} = \mathbf{B}$ and previously generated matrix \mathbf{F} and predicate matrix \mathbf{W} compute the ciphertext of certificate $\text{cert}_i = (\text{id}_i, \mathbf{t}_i, \mathbf{s}_i)$, implementing the values (j_1, \dots, j_1) into the encryption as follows: take vectors $\mathbf{t}_i, \mathbf{s}_i \in (\mathbb{Z}_q^{2\mu})^2$, set $\mathbf{x}_i = \mathbf{t}_i + \mathbf{s}_i$ and choose $\mathbf{e}_1 \leftarrow \chi^{2\mu}, \mathbf{e}_2 \leftarrow \chi^1$. The ciphertext is given by $\mathbf{C} = (\mathbf{c}_1, \mathbf{c}_2) = ([\mathbf{B} | 0] \cdot \mathbf{x}_i + \mathbf{e}_1, [\mathbf{W} | \mathbf{F}] \cdot \mathbf{x}_i + \mathbf{e}_2 + \lfloor \frac{q}{2} \rfloor \cdot (j_1, \dots, j_1, 0, \dots, 0)^T) \in \mathbb{Z}_q^{2\mu} \times \mathbb{Z}_q^1$.

(5) Generate a NIZK proof introduced in the previous section to show the possession of a valid tuple $(\mathbf{z}_i, \mathbf{d}^{(i)}, \mathbf{w}^{(i)}, \mathbf{x}_i)$. Furthermore prove that \mathbf{C} is an encryption of (j_1, \dots, j_1) with random values \mathbf{x}_i . The public input of the protocol consists of the group public key and the ciphertext \mathbf{C} . In order to achieve soundness, the NIZK proof is run $\theta = \omega(\log n)$ times. The proof is given using Fiat-Shamirs heuristic as $\Pi = (\{\text{Com}_i\}_{i=1}^\theta, \text{Ch}, \{\text{RSP}_i\}_{i=1}^\theta)$, where the challenge is a hash-function on input a message m , public parameters and commitment Com_i . The output is signature $\Sigma = (\Pi, \mathbf{C}, \sigma_{\text{ots}})$.

ABGVerify(param, gpk, Σ): Parse signature $\Sigma = (\Pi, \mathbf{C}, \sigma_{\text{ots}})$. Return 1 if $\mathcal{V}(\text{vk}, \mathbf{C}, \Pi, \mathbf{F}) = 1$ and the proof Π verifies correctly. Otherwise return \perp .

$\text{ABGOpen}(\text{param}, \text{gmsk}, \Sigma)$: Taking group master secret key $\text{gmsk} = \mathbf{T}_B$ and signature Σ and public parameters which include the one-time signature verification key vk_{ots} and the hash function H , the opening procedure is as follows:

- (1) Taking the hash function H and the verification key vk_{ots} , it computes $F = H(\text{vk}_{\text{ots}})$ and using the public key B it generates a matrix G_{vk} , such that holds $[B|0] \cdot G_{\text{vk}} = [W|F] \pmod{\mathbf{q}}$.
- (2) Decrypt the ciphertext using the obtained matrix G_{vk} as follows: $c_2 - c_1 \cdot G_{\text{vk}} = (j_1, \dots, j_1, 0, \dots, 0)$.

Theorem 3. *Our ABGSL scheme is fully-anonymous and fully traceable if the underlying NIZK proof is simulation sound and zero-knowledge provable.*

The proof of this theorem is given in Appendix C.

5 Conclusion

In this paper, we formalized a new zero-knowledge argument (ZKA), which represents a suitable tool for lattice-based constructions. The argument uses Merkle-Damgård compression function, as building blocks and recent lattice based constructions are based on the same tree structure. We employed the introduced ZKA to security assurance of the proposed structure of attribute-based group signature. The tree construction of ZKA allows to hide the attributes of our ABGS scheme and to prove in zero-knowledge the correct construction of user’s attribute-based secret key. We provided security proofs for both constructions, ZKA and ABS, where the security of each tool is based on a lattice problem.

A Definitions

Discrete Gaussians. Let L be a subset of \mathbb{Z}^m . For a vector $\mathbf{c} \in \mathbb{R}^m$ and a positive $\sigma \in \mathbb{R}$, define

$$\rho_{\mathbf{s},\mathbf{c}}(\mathbf{x}) = \exp\left(-\pi \frac{\|\mathbf{x} - \mathbf{c}\|^2}{\mathbf{s}^2}\right) \quad \text{and} \quad \rho_{\mathbf{s},\mathbf{c}}(L) = \sum_{\mathbf{x} \in L} \rho_{\mathbf{s},\mathbf{c}}(\mathbf{x}).$$

The discrete Gaussian distribution over L with center \mathbf{c} and parameter \mathbf{s} is given by $\mathcal{D}_{L,\mathbf{s},\mathbf{c}}(\mathbf{y}) = \frac{\rho_{\mathbf{s},\mathbf{c}}(\mathbf{y})}{\rho_{\mathbf{s},\mathbf{c}}(L)}$, for all $\mathbf{y} \in L$. The distribution $\mathcal{D}_{L,\mathbf{s},\mathbf{c}}$ is usually defined over the lattice $L = \Lambda_{\mathbf{q}}^{\perp}(\mathbf{A})$ for $\mathbf{A} \in \mathbb{Z}_{\mathbf{q}}^{n \times m}$.

The security of our construction and the underlying building blocks is based on the hardness of $\text{SIVP}_{\mathcal{O}(n)}$ and LWE problems which we recall in the following two definitions.

B Non-interactive Zero-Knowledge Proof

A non-interactive proof system $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ for a relation R with setup consists of four PPT algorithms: a setup algorithm \mathcal{G} , a common reference string (CRS) generation algorithm \mathcal{K} , a prover \mathcal{P} and a verifier \mathcal{V} . The setup algorithm outputs public parameters I and a commitment key ck . The CRS generation algorithm takes (I, ck) as input and outputs a CRS Σ . The prover \mathcal{P} takes as input (I, Σ, x, ω) , where x is the statement and ω is the witness, and outputs a proof π . The verifier \mathcal{V} takes as input (I, Σ, x, π) and outputs 1 if the proof is acceptable and 0 otherwise. $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ is non-interactive proof system for R if it has the following properties:

Completeness. A non-interactive proof is complete if an honest prover can convince an honest verifier whenever the statement belongs to the language and the prover holds a witness testifying to this fact. For all adversaries \mathcal{A} we have:

$$\begin{aligned} &Pr[(I, ck) \leftarrow \mathcal{G}(1^\lambda); \Sigma \leftarrow \mathcal{K}(I, ck); (x, \omega) \leftarrow \mathcal{A}(I, \Sigma) : \\ &\pi \leftarrow \mathcal{P}(I, \Sigma, x, \omega) : \mathcal{V}(I, \Sigma, x, \pi) = 1 \text{ if } (I, x, \omega) \in R] = 1. \end{aligned}$$

Soundness. A non-interactive proof is sound if it is impossible to prove a false statement x which is not an element of a language L . We say $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ is perfectly sound if for all adversaries \mathcal{A} we have:

$$\begin{aligned} &Pr[(I, ck) \leftarrow \mathcal{G}(1^\lambda); \Sigma \leftarrow \mathcal{K}(I, ck); (x, \pi) \leftarrow \mathcal{A}(I, \Sigma); \\ &\mathcal{V}(I, \Sigma, x, \pi) = 0 \text{ if } x \notin L] = 1. \end{aligned}$$

Knowledge Extraction. We say that $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ is a proof of knowledge for R if there exists a knowledge extractor $\mathcal{E} = (\mathcal{E}_1, \mathcal{E}_2)$ with the following properties: For all PPT adversaries \mathcal{A} we have

$$\begin{aligned} &Pr[(I, ck) \leftarrow \mathcal{G}(1^\lambda); \Sigma \leftarrow \mathcal{K}(I, ck) : \mathcal{A}(I, \Sigma) = 1] \\ &= Pr[I \leftarrow \mathcal{G}(1^\lambda); (\Sigma, \xi) \leftarrow \mathcal{E}_1(I, ck) : \mathcal{A}(I, \Sigma) = 1]. \end{aligned}$$

For all adversaries \mathcal{A} holds

$$\begin{aligned} &Pr[(I, ck) \leftarrow \mathcal{G}(1^\lambda); (\Sigma, \xi) \leftarrow \mathcal{E}_1(I, ck); (x, \pi) \leftarrow \mathcal{A}(I, \Sigma); \\ &\omega \leftarrow \mathcal{E}_2(\Sigma, \xi, x, \pi) : \mathcal{V}(I, \Sigma, x, \pi) = 0 \text{ or } (I, x, \omega) \in R] = 1. \end{aligned}$$

Zero-Knowledge. We say that $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ is a composable NIZK proof if there exists a PPT simulator $(\mathcal{S}_1, \mathcal{S}_2)$ such that for all PPT adversaries \mathcal{A} we have

$$\begin{aligned} &Pr[(I, ck) \leftarrow \mathcal{G}(1^\lambda); \Sigma \leftarrow \mathcal{K}(I, ck) : \mathcal{A}(I, \Sigma) = 1] \\ &\approx Pr[I \leftarrow \mathcal{G}(1^\lambda); (\Sigma, \tau) \leftarrow \mathcal{S}_1(I, ck) : \mathcal{A}(I, \Sigma) = 1], \end{aligned}$$

and for all adversaries \mathcal{A} holds:

$$\begin{aligned} &Pr[(I, ck) \leftarrow \mathcal{G}(1^\lambda); (\Sigma, \tau) \leftarrow \mathcal{S}_1(I, ck); (x, \omega) \leftarrow \mathcal{A}(I, \Sigma, \tau); \\ &\pi \leftarrow \mathcal{P}(I, \Sigma, x, \omega) : \mathcal{A}(\pi) = 1] \\ &\approx Pr[(I, ck) \leftarrow \mathcal{G}(1^\lambda); (\Sigma, \tau) \leftarrow \mathcal{S}_1(I, ck); \\ &(I, x, \omega) \leftarrow \mathcal{A}(I, \Sigma, \tau); \pi \leftarrow \mathcal{S}_2(I, \Sigma, \tau, x) : \mathcal{A}(\pi) = 1], \end{aligned}$$

where \mathcal{A} outputs $(I, x, \omega) \in R$. We obtain a strong notion of zero-knowledge, called composable zero-knowledge [15]. It implies standard zero-knowledge and is simpler to work with, because it separates the computational indistinguishability into two parts considering the CRS and the proofs respectively.

Simulation Soundness. We say that $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ is simulation sound if a PPT adversary \mathcal{A} cannot prove false statements even if he have seen simulated proofs of arbitrary statements: For all PPT adversaries \mathcal{A} we have

$$\begin{aligned} &Pr[(I, ck) \leftarrow \mathcal{S}_1(1^\lambda); (x, \pi) \notin SQL \\ &\wedge x \notin Land\mathcal{V}(I, x, \pi) = 1] = \epsilon(\lambda). \end{aligned}$$

C Preimage Sampling Function

In this section, we recall the notion of preimage sampling functions (PSF) introduced in [13]. The idea of that function is a combination of a trapdoor construction for integer lattices and an efficient discrete Gaussian sampling algorithm. Let $\mathbf{A} \in \mathbb{Z}_q^{\nu \times \mu}$ be a uniform matrix and $\mathbf{T}_\mathbf{A}$ the corresponding basis for the lattice $\Lambda^\perp(\mathbf{A})$, which can be used as a trapdoor for finding small non-zero solution $\mathbf{e} \in \mathbb{Z}_q^\mu$ of the equation $\mathbf{A}\mathbf{e} = \mathbf{0} \pmod q$.

Definition 11 (PSF). *Let λ be a security parameter, ν a security dimension and μ a dimension of the lattice base. Let $\mathbf{s} \geq L\omega(\sqrt{\log m})$ be some discrete Gaussian parameter. A PSF family consists of maps $\mathbf{f}_\mathbf{A} : \mathbb{D}_{\mathbb{Z}^\mu, \mathbf{s}} \rightarrow \mathbb{Z}_q^\nu$ with the domain $\mathbb{D}_{\mathbb{Z}_q^\mu, \mathbf{s}} = \{\mathbf{e} \in \mathbb{Z}^\mu : \|\mathbf{e}\| \leq \sqrt{m}\mathbf{s}\} \subseteq \mathbb{Z}^\mu$ and is specified by the following four algorithms:*

TrapGen (1^λ) : On input security parameter 1^λ it outputs a uniform matrix $\mathbf{A} \in \mathbb{Z}_q^{\nu \times \mu}$ and a basis $\mathbf{T}_\mathbf{A}$ of $\Lambda^\perp(\mathbf{A})$ such that $\|\tilde{\mathbf{T}}_\mathbf{A}\| \leq L$, where L is the circuit depth. The public parameters are (\mathbf{A}, \mathbf{q}) and the preimage-sampling trapdoor is $\mathbf{T}_\mathbf{A}$.

EvalFun $(\mathbf{A}, \mathbf{q}, \mathbf{e})$: On input public parameters (\mathbf{A}, \mathbf{q}) and a point $\mathbf{e} \in \mathbb{D}_{\mathbb{Z}_q^\mu, \mathbf{s}}$, the algorithm outputs the image $\mathbf{f}_\mathbf{A}(\mathbf{e}) = \mathbf{A}\mathbf{e} \pmod q \in \mathbb{Z}_q^\nu$.

SampleDom $(\mathbb{I}^{\mu \times \mu}, \mathbf{s})$: On input the identity matrix $\mathbb{I}^{\mu \times \mu}$ and a Gaussian parameter \mathbf{s} , it outputs a vector $\mathbf{e} \leftarrow \text{SampleGaussian}(\mathbb{I}^{\mu \times \mu}, \mathbf{s}, 0)$, i.e. $\mathbf{e} \sim \mathcal{D}_{\mathbb{Z}_q^\mu, \mathbf{s}}$.

(SampleGaussian $(\mathbb{I}^{\mu \times \mu}, \mathbf{s}, \mathbf{c})$) algorithm works as follows. On input a basis $\mathbf{I}^{\mu \times \mu}$ for a Lattice $\Lambda \subset \mathbb{R}^\mu$ a parameter $\mathbf{s} \geq \omega(\sqrt{m})$ and a center $\mathbf{c} \in \mathbb{R}^\mu$, it outputs a lattice vector $\mathbf{x} \in \Lambda$, such that $\mathbf{x} \sim \mathcal{D}_{\Lambda, \mathbf{s}, \mathbf{c}}$.

SamplePre $(\mathbf{A}, \mathbf{q}, \mathbf{T}_\mathbf{A}, \mathbf{s}, \mathbf{u})$: On input public parameters (\mathbf{A}, \mathbf{q}) and a trapdoor $\mathbf{T}_\mathbf{A}$, a Gaussian parameter \mathbf{s} and a target image $\mathbf{u} \in \mathbb{Z}_q^\nu$, the algorithm samples $\mathbf{e} \in \mathbb{D}_{\mathbb{Z}_q^\mu, \mathbf{s}}$ from $\mathcal{D}_{\mathbb{Z}^\mu, \mathbf{s}}$, such that $\mathbf{A}\mathbf{e} = \mathbf{u} \pmod q$. It first finds a solution $\mathbf{c} \in \mathbb{Z}^\mu$ in the linear system $\mathbf{A}\mathbf{c} = \mathbf{u} \pmod q$.

It samples a vector $\mathbf{d} \leftarrow \text{SampleGaussian}(\mathbf{T}_\mathbf{A}, \mathbf{s}, -\mathbf{c}) \sim \mathcal{D}_{\Lambda^\perp(\mathbf{A}, \mathbf{s}, -\mathbf{c})}$ and outputs vector $\mathbf{e} = \mathbf{c} + \mathbf{d} \in \mathbb{Z}^\mu$.

Lemma 1. *There exists a PPT algorithm ExtBasis , that takes as input a matrix $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$ and a basis \mathbf{T}_A of $\Lambda_q^\perp(\mathbf{A})$, where $\mathbf{A} \in \mathbb{Z}_q^{n \times m'}$ is a submatrix of \mathbf{B} and outputs a basis \mathbf{T}_B of the extended lattice $\Lambda_q^\perp(\mathbf{B})$ with the property $\|\widetilde{\mathbf{T}}_B\| \leq \|\widetilde{\mathbf{T}}_A\|$.*

D Security Analysis of Theorem 3

Proof. In order to provide the proof of this theorem we are using the following lemmas:

Lemma 2. *If the underlying public key encryption systems are IND-CCA secure and the NIZK proof is simulation sound and zero-knowledge, then our ABGSL scheme is fully-anonymous under the hardness of $\text{SIVP}_{O(\lambda)}$ problem.*

Lemma 3. *Our ABGS scheme is attribute anonymous under the hardness of $\text{SIVP}_{O(\lambda)}$ problem and if the underlying public key encryption scheme is IND-CCA secure and the underlying NIZK proofs is simulation-sound and computationally zero-knowledge provable.*

Lemma 4. *If the underlying public key encryption is IND-CCA secure, digital signature scheme is unforgeable against chosen message attacks and the NIZK proofs are simulation sound, then our ABGS scheme is fully-traceable under the hardness of $\text{SIVP}_{O(\lambda)}$ problem.*

Due to page limit we only provide a sketch of Lemma 1. The full proof will be given in the full version of this paper.

Proof of Lemma 2. Let \mathcal{A}_{uan} be an adversary against the user’s full-anonymity in the ABGSL scheme. We design an adversary $\mathcal{B}_\gamma \in (\mathcal{B}_{\text{SIVP}}, \mathcal{B}_{\text{pke}}, \mathcal{B}_{\text{sig}})$ against the $\text{SIVP}_{O(\lambda)}$ problem or against IND-CCA security of the underlying encryption scheme or against unforgeability of the underlying signature scheme, respectively. We show how to construct \mathcal{B}_γ to simulate \mathcal{A}_{uan} .

Setup: Algorithm $\mathcal{B}_{\text{SIVP}}$ simulates public parameters and master secret key by first sampling the following vectors: $\mathbf{a}_i^1, \dots, \mathbf{a}_i^\mu \in (\mathbb{Z}_q^\nu)^\mu$, where $i \in [0, 1]$. It sets $\mathbf{A} = [\mathbf{a}_0^1 | \dots | \mathbf{a}_0^\mu]$ and analogously $\mathbf{A}_i = [\mathbf{a}_i^1 | \dots | \mathbf{a}_i^\mu]$, for each $i \in [1, 1]$. $\mathcal{B}_{\text{SIVP}}$ samples for each $i \in [l]$ a uniformly random matrix \mathbf{B}_i and uses TrapGen algorithm on input these matrices to generate $msk = \mathbf{T}_{A,i}$. To simulate public and secret key of the underlying encryption scheme, \mathcal{B}_{pke} runs its $\text{Setup}(1^\lambda)$ algorithm of Regev’s encryption scheme on input security parameter and outputs and a master secret key $\text{gmpk} = \mathbf{B}$, $\text{gmsk} = \mathbf{T}_B$. \mathcal{B}_{pke} forwards these values to \mathcal{A}_{uan} . In order to simulate secret and public keys of key issuing authorities, algorithm \mathcal{B}_{sig} proceeds similarly to algorithm \mathcal{B}_{pke} by running its own Setup algorithm and outputting $\text{sk}_{kia}, \text{pk}_{kia}$. The detailed description of the adversary \mathcal{A}_{uan} is given in the following experiment:

- (1.) $(\text{vk}_{\text{ots}}, \text{sk}_{\text{ots}}) \leftarrow \text{Setup}_{\text{ots}}(1^\lambda)$
- (2.) $(\text{gmpk}, \text{gmsk}) \leftarrow \text{Setup}_e(1^\lambda)$

- (3.) $(\mathbf{pk}_{\text{kia}}, \mathbf{sk}_{\text{kia}}) \leftarrow \text{Setup}_s(1^\lambda)$
- (4.) $(\text{crs}, \mathbf{R}') \leftarrow \text{SIM}(\text{generate}, \lambda)$
- (5.) Set $\text{gpk} = (\lambda, \mathbf{R}', \text{gmpk}, \mathbf{pk}_{\text{kia}}, \mathbf{vk}_{\text{ots}})$

For all users $i \in [n]$ run $(\mathbf{pk}_i, \mathbf{sk}_i) \leftarrow \text{Setup}_s(1^\lambda)$.

Compute $\text{cert}_i \leftarrow \text{Sign}(\mathbf{sk}_{\text{kia}}, \langle i, \mathbf{pk}_i \rangle)$. Make oracle queries to $\mathcal{O}\text{Setup}$ and $\mathcal{O}\text{Decrypt}$ of the public key encryption scheme.

Queries to $\mathcal{O}\text{ABGOpen}(\cdot, \cdot)$: Whenever \mathcal{A}_{uan} calls its opening oracle on input a message \mathbf{m} and a signature σ , algorithm \mathcal{B}_γ simulates these opening queries by first simulating the secret key of the group manager. In case the oracle's output is \mathbf{m} , it returns 1 to \mathcal{A}_{uan} adversary.

To simulate user's attribute-based secret key, algorithm \mathcal{B}_γ is invoked and proceeds as follows taking as input public parameters param and the simulated master secret key $\text{msk} = \{\mathbf{T}_{\mathbf{A}, i}\}_{i \in [1]}$. It chooses a random set of attributes it wants to be challenged on, $\mathbb{A}_i = \{a_1, \dots, a_\kappa\}$, where $\kappa \in [0, N - 1]$. $\mathcal{B}_{\text{SIVP}}$ associates each attribute a_j with a leaf of the Merkle-tree d_j , where $j \in [0, N - 1]$ by assigning to each attribute a_j a binary string in $\{0, 1\}^\nu$ via the following computation $(\text{bin} \cdot \mathbf{A}_i \cdot \mathbf{u}_j \bmod \mathbf{q}) = \mathbf{d}_j$. Let $\mathbf{R} = (\mathbf{d}_0, \dots, \mathbf{d}_{N-1})$. $\mathcal{B}_{\text{SIVP}}$ runs $\text{Tcalc}(\mathbf{R})$ to generate the complete tree and the values \mathbf{u}, \mathbf{v} for the hash function of the root node $F_0(\mathbf{u}, \mathbf{v})$. It uses the set \mathbf{R} and one of the tree leaves \mathbf{d}_j as input of TWitness algorithm to generate the witness of the zero-knowledge proof:

$$\mathbf{w} = ((j_1, \dots, j_1), (\mathbf{d}_{j_1}, \mathbf{f}_{j_1}), \dots, (\mathbf{d}_{j_1}, \mathbf{f}_{j_1}), (\mathbf{w}_1, \dots, \mathbf{w}_1)) \in \{0, 1\}^1 \times \{0, 1\}^{21} \times \{0, 1\}^{1\nu\kappa},$$

To simulate the valid credentials provided by the key issuing authority for each user i with k distinct attributes $\mathbf{u}_j^{(i)}$, $j \in [k]$, $i \in [N]$, $\mathcal{B}_{\text{SIVP}}$ first computes a sum of the different attributes and sets $\mathbf{u}^{(i)} = \sum_{j=1}^k \mathbf{u}_j^{(i)}$. It samples values \mathbf{z}_i by running the algorithm $\text{SamplePre}(\mathbf{A}_i, \mathbf{q}, \mathbf{T}_{\mathbf{A}, i}, \mathbf{s}, \mathbf{u}^{(i)})$. It returns the attribute-based secret key $\text{usk}[i] = \mathbf{sk}_{\mathbb{A}_i} = (\mathbf{z}_i, \mathbf{w}^{(i)}, \mathbf{d}^{(i)})$, where the tuple $\mathbf{d}^{(i)} = (\mathbf{d}_{j_1}^{(i)}, \dots, \mathbf{d}_{j_k}^{(i)})$, describes a set of attributes, \mathbb{A}_i of an user i .

Challenge: When \mathcal{A}_{uan} outputs $(\text{state}, i_0, i_1, \mathbf{m})$, it picks a bit $\mathbf{b} \in \{0, 1\}$ and computes a signature $\sigma_{\mathbf{b}} \leftarrow \text{ABGSign}(\text{param}, \text{usk}[i_{\mathbf{b}}], \mathbf{m}, \Gamma)$, simulator invokes its $\mathcal{B}_{\text{SIVP}}$, who randomly simulates two messages $\mathbf{m}_0, \mathbf{m}_1$.

Furthermore \mathcal{A}_{uan} invokes the \mathcal{B}_{ots} algorithm to simulates the keys of OTS scheme by running $(\mathbf{vk}_{\text{ots}}, \mathbf{sk}_{\text{ots}}) \leftarrow \text{Setup}_{\text{ots}}$. The verification key \mathbf{vk}_{ots} will be a part of the NIZK proof. $\mathcal{B}_{\text{SIVP}}$ signs \mathbf{vk}_{ots} using simulated secret key $\text{usk}[i]$, where the secret key simulation is given by a random guess with probability $1/|\mathcal{K}|$ with the key space \mathcal{K} . The guessing probability reduces \mathcal{B}_γ 's advantage to win the game. If the guess of the keys does not match with the real secret key, the simulation aborts. The signature procedure continues as follows: Taking \mathbf{K} and the verification key \mathbf{vk}_{ots} as a message, it runs encapsulation algorithm of the underlying DEM scheme, $\tilde{\sigma} = \text{Encrypt}(\mathbf{vk}_{\text{ots}})$. Furthermore \mathcal{B}_{pke} of the underlying encryption scheme is invoked, which outputs a ciphertext encrypting user's certificate $\text{cert}_{i_{\mathbf{b}}}$, and signature $\tilde{\sigma}$, i.e. $\mathbf{C} \leftarrow \text{Encrypt}(\text{gmpk}, \langle i_{\mathbf{b}}, \mathbf{pk}_{i_{\mathbf{b}}}, \text{cert}_{i_{\mathbf{b}}}, \tilde{\sigma}, \mathbf{R}' \rangle)$,

where R' is a randomness used in the NIZK proof. Finally taking as input a message m , verification key vk_{ots} , ciphertext C and the corresponding proof π , \mathcal{B}_{sig} runs the signature algorithm of the underlying OTS scheme and outputs $\sigma_{ots} \leftarrow \text{Sign}(m, vk_{ots}, C, \pi)$. Furthermore, simulator runs the NIZK proof π_1 from the ABS scheme to show the possession of a valid tuple $(z_i, d^{(i)}, w^{(i)}, x_i)$. Furthermore, it proves that C is an encryption of (j_1, \dots, j_1) with random values x_i . The final signature is equal to $\Sigma = (C, \pi)$. We note that whenever \mathcal{A}_{uan} is submitting a query (C, π') to the opening oracle, simulator invokes its \mathcal{B}_{pke} algorithm and forwards the query to its decryption oracle. Finally it outputs a bit b and terminates the simulation.

Distinguisher for Zero-Knowledge. Distinguisher involved in the NIZK proof is given in the following description of the algorithm $\mathcal{D}(\text{choose}, \lambda, R')$:

- (1.) $(vk_{ots}, sk_{ots}) \leftarrow \text{Setup}_{ots}(1^\lambda)$
 - (2.) $(gmpk, gmsk) \leftarrow \text{Setup}_e(1^\lambda)$
 - (3.) $(pk_{kia}, sk_{kia}) \leftarrow \text{Setup}_s(1^\lambda)$
 - (4.) $(crs, R') \leftarrow \text{SIM}(\text{generate}, \lambda)$
 - (5.) Set $gpk = (\lambda, R', gmpk, pk_{kia}, vk_{ots})$
- End for:
- (a.) $(state, i_0, i_1, m^*, vk_{ots}^*, \Gamma^*) \leftarrow \mathcal{A}_{uan}^{\text{ABGOpen}(\cdot)}(\cdot);$
 - (b.) $b \in \{0, 1\}, R \in \{0, 1\}^\lambda;$
 - (c.) $C^* \leftarrow \text{Encrypt}(gmpk, (i_b, pk_{i_b}, cert_{i_b}, \tilde{\sigma}^*, R'));$
 - (d.) $\sigma_{ots} \leftarrow \text{Sign}_{ots}(m^*, vk_{ots}^*, C^*, \pi^*).$

We note that distinguisher \mathcal{D} can answer any queries submitted by \mathcal{A}_{uan} , because it is in possession of group manager's secret key, which can be used to open the signatures. The output of the challenge phase is a signature given as (pk_e, pk_s, m, C) together with a witness. In the second stage, distinguisher takes as input a proof π and creates a group signature $\Sigma = (C, \pi, \sigma_{ots})$ and outputs it to the adversary \mathcal{A}_{uan} . Finally, the distinguisher \mathcal{D} outputs the same value as that one of the output of \mathcal{A}_{uan} .

Soundness of NIZK proof. In order to prove simulation soundness of the NIZK proof, we consider the following game where an adversary \mathcal{A}_{ss} against simulation soundness of NIZK is playing against a challenger, who is represented by the adversary against our ABGS scheme:

- (1.) $(vk_{ots}, sk_{ots}) \leftarrow \text{Setup}_{ots}(1^\lambda)$
 - (2.) $(gmpk, gmsk) \leftarrow \text{Setup}_e(1^\lambda)$
 - (3.) $(pk_{kia}, sk_{kia}) \leftarrow \text{Setup}_s(1^\lambda)$
 - (4.) $(crs, R') \leftarrow \text{SIM}(\text{generate}, \lambda)$
 - (5.) Set $gpk = (\lambda, R', gmpk, pk_{kia}, vk_{ots})$
- End for:
- (a.) $m^*, \Gamma^*, \sigma^* \leftarrow \mathcal{A}_{uan}^{\text{ABGOpen}(\text{param}, gmsk, \cdot)}(\text{param}, msk, \cdot);$
 - (b.) $C \leftarrow \text{Encrypt}(pk_e, (i_b, pk_{i_b}, cert_{i_b}, \sigma_b, R'));$
 - (c.) $\sigma_{ots} \leftarrow \text{Sign}_{ots}(m^*, vk_{ots}^*, C^*, \pi^*);$
 - (d.) $\pi \leftarrow \text{SIM}(\text{prove}, crs, \text{param}, m^*, \sigma^*, sk_A, \Gamma^*).$

Make oracle queries to $\mathcal{O}ABGKeyGen$ to simulate user's attribute-based secret key sk_{A_1} .

Run $\text{Verify}(\text{param}, \sigma_{\text{ots}}, \pi, \mathcal{C})$ of the NIZK proof. If \mathcal{A}_{uan} outputs a valid tuple $(\sigma_{\text{ots}}, \pi', \mathcal{C})$, output $(\text{param}, \text{crs}, \sigma_{\text{ots}}, \pi', \mathcal{C})$.

Due to the page limit we provide only the final result of adversary's success. For the detailed analysis of this proof, we refer to the later full version of this paper. Finally we conclude that the advantage of an adversary \mathcal{A}_{uan} is given by the following combined inequation:

$$\text{Adv}_{\mathcal{A}_{\text{uan}}, \text{ABGS}}^{\text{U-ANO}} \leq \text{Adv}_{\mathcal{A}_{\text{ss}}, \text{ABGS}}^{\text{Sim-Sound}} + \text{Adv}_{\mathcal{A}_{\text{indPKE}}}^{\text{IND-CCA}} + \text{Adv}_{\mathcal{A}_{\text{zk}}, \text{NIZK}}^{\text{Zero-Knowledge}}$$

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_28
2. Ali, S.T., Amberker, B.: Attribute-based group signature without random oracles with attribute anonymity. *Int. J. Inf. Comput. Secur.* **6**(2), 109–132 (2014)
3. Bellare, M., Micciancio, D., Warinschi, B.: Foundations of group signatures: formal definitions, simplified requirements, and a construction based on general assumptions. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 614–629. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-39200-9_38
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). https://doi.org/10.1007/978-3-540-30574-3_11
5. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_4
6. Boyen, X.: Lattice mixing and vanishing trapdoors: a framework for fully secure short signatures and more. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 499–517. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13013-7_29
7. Boyen, X., Waters, B.: Full-domain subgroup hiding and constant-size group signatures. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 1–15. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-71677-8_1
8. Chaum, D., Van Heyst, E.: Group signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991). https://doi.org/10.1007/3-540-46416-6_22
9. El Bansarkhani, R., El Kaafarani, A.: Post-quantum attribute-based signatures from lattice assumptions. IACR Cryptology ePrint Archive, 2016, p. 823 (2016)
10. Emura, K., Miyaji, A., Omote, K.: A dynamic attribute-based group signature scheme and its application in an anonymous survey for the collection of attribute statistics. *Inf. Media Technol.* **4**(4), 1060–1075 (2009)
11. Escala, A., Herranz, J., Morillo, P.: Revocable attribute-based signatures with adaptive security in the standard model. In: Nitaj, A., Pointcheval, D. (eds.) AFRICACRYPT 2011. LNCS, vol. 6737, pp. 224–241. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21969-6_14

12. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, pp. 197–206. ACM (2008)
13. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC 2008, pp. 197–206. ACM (2008)
14. Gordon, S.D., Katz, J., Vaikuntanathan, V.: A group signature scheme from lattice assumptions. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 395–412. Springer, Heidelberg (2010)
15. 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). https://doi.org/10.1007/11935230_29
16. Herranz, J., Laguillaumie, F., Libert, B., Ràfols, C.: Short attribute-based signatures for threshold predicates. In: Dunkelman, O. (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 51–67. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27954-6_4
17. Jia, X., Yupu, H., Juntao, G., Wen, G., Xuelian, L.: Attribute-based signatures on lattices. *J. China Univ. Posts Telecommun.* **23**(4), 83–90 (2016)
18. Khader, D.: Attribute based group signature with revocation. IACR Cryptology ePrint Archive 2007, p. 241 (2007)
19. Khader, D.: Attribute based group signatures. IACR Cryptology ePrint Archive 2007, p. 159 (2007)
20. Laguillaumie, F., Langlois, A., Libert, B., Stehlé, D.: Lattice-based group signatures with logarithmic signature size. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. LNCS, vol. 8270, pp. 41–61. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-42045-0_3
21. Langlois, A., Ling, S., Nguyen, K., Wang, H.: Lattice-based group signature scheme with verifier-local revocation. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 345–361. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54631-0_20
22. 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). https://doi.org/10.1007/978-3-642-13190-5_4
23. Li, J., Kim, K.: Attribute-based ring signatures. IACR Cryptology EPrint Archive 2008, p. 394 (2008)
24. Liang, X., Cao, Z., Shao, J., Lin, H.: Short group signature without random oracles. In: Qing, S., Imai, H., Wang, G. (eds.) ICICS 2007. LNCS, vol. 4861, pp. 69–82. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-77048-0_6
25. Libert, B., Ling, S., Mouhartem, F., Nguyen, K., Wang, H.: Signature schemes with efficient protocols and dynamic group signatures from lattice assumptions. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016. LNCS, vol. 10032, pp. 373–403. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53890-6_13
26. Libert, B., Ling, S., Nguyen, K., Wang, H.: Zero-knowledge arguments for lattice-based accumulators: logarithmic-size ring signatures and group signatures without trapdoors. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9666, pp. 1–31. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_1
27. Ling, S., Nguyen, K., Stehlé, D., Wang, H.: Improved zero-knowledge proofs of knowledge for the ISIS problem, and applications. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 107–124. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36362-7_8

28. Maji, H.K., Prabhakaran, M., Rosulek, M.: Attribute-based signatures: achieving attribute-privacy and collusion-resistance. IACR Cryptology ePrint Archive 2008, p. 328 (2008)
29. Maji, H.K., Prabhakaran, M., Rosulek, M.: Attribute-based signatures. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 376–392. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19074-2_24
30. Mao, X.-P., Chen, K.-F., Long, Y., Wang, L.-L.: Attribute-based signature on lattices. J. Shanghai Jiaotong Univ. (Sci.) **19**(4), 406–411 (2014)
31. Micciancio, D., Vadhan, S.P.: Statistical zero-knowledge proofs with efficient provers: lattice problems and more. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 282–298. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_17
32. Nguyen, P.Q., Zhang, J., Zhang, Z.: Simpler efficient group signatures from lattices. In: Katz, J. (ed.) PKC 2015. LNCS, vol. 9020, pp. 401–426. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46447-2_18
33. Okamoto, T., Takashima, K.: Efficient attribute-based signatures for non-monotone predicates in the standard model. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 35–52. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19379-8_3
34. Patel, B.K., Jinwala, D.: Anonymity in attribute-based group signatures. In: Thilagam, P.S., Pais, A.R., Chandrasekaran, K., Balakrishnan, N. (eds.) ADCONS 2011. LNCS, vol. 7135, pp. 495–504. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29280-4_58
35. Regev, O.: On lattices, learning with errors, random linear codes and cryptography. In: STOC 2005, pp. 84–93. ACM (2005)
36. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM (JACM) **56**(6), 34 (2009)
37. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_32
38. Shahandashti, S.F., Safavi-Naini, R.: Threshold attribute-based signatures and their application to anonymous credential systems. In: Preneel, B. (ed.) AFRICACRYPT 2009. LNCS, vol. 5580, pp. 198–216. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02384-2_13
39. Zhang, Y., Hu, Y., Jiang, M.: An attribute-based signature scheme from lattice assumption. Wuhan Univ. J. Nat. Sci. **20**(3), 207–213 (2015)



Security Analysis of Improved Cubic UOV Signature Schemes

Kyung-Ah Shim^(✉), Namhun Koo, and Cheol-Min Park

Division of Integrated Mathematics, National Institute for Mathematical Sciences,
Daejeon, Republic of Korea
{kashim,nhkoo,mpcm}@nims.re.kr

Abstract. At ICISC 2016, Duong *et al.* proposed two signature schemes based on multivariate quadratic equations, CSSv and SVSv by improving the security of the cubic UOV against Hashimoto's attack. They claimed that the schemes were secure against all known attacks. We show that the schemes are insecure against key recovery attack using good keys and HighRank attacks. From a practical point of view, we are able to break their parameter at an 128-bit security level in 2 min by using the HighRank attack.

Keywords: Equivalent key · Key recovery attack using good keys
Multivariate-quadratic scheme · HighRank attack

1 Introduction

The existence of a sufficiently large quantum computer that is capable of implementing Shor's algorithm [13] would be a real-world threat to break RSA, Diffie-Hellman key exchange, DSA and ECDSA the most widely used public-key cryptography in practice. Public-key cryptography based on multivariate quadratic equations (MQ-PKC) is one of the most promising alternatives for classical PKC. MQ-PKC is based on the NP-hard problem of solving random systems of quadratic equations over finite fields, known as the MQ-problem. Since the first MQ-encryption scheme was proposed by Imai and Matsumoto [9], there have been proposed a number of MQ-schemes. However, most of MQ-schemes have been broken except HFEv- variants [11,12] and Unbalanced Oil and Vinegar (UOV) variants as signature schemes [2,8].

At Inscrypt 2015, Nie *et al.* [10] proposed a cubic UOV (CUOV) which is a variant of UOV based on cubic polynomials with shorter signatures and a smaller private key than UOV and Rainbow. However, Hashimoto [7] showed that equivalent secret keys of CUOV could be recovered easily. Recently, Duong *et al.* [4] proposed two new MQ-signature schemes, CSSv and SVSv. They claimed that they were secure against Hashimoto's attack and all other known attacks of MQ-schemes. In this paper, we show that their schemes are entirely broken by presenting key recovery attack using good keys and HighRank attacks.

The rest of the paper is organized as follows. In Sect. 2, we describe the two MQ-signature scheme, SVSv and CSSv. We present key recovery attacks using good keys and HighRank attacks on the schemes in Sects. 3 and 4, respectively. Concluding remarks are given in Sect. 5.

2 Improved Cubic UOV Signature Schemes

We describe two improved UOV signature schemes, SVSv and CSSv in [4].

2.1 SVSv

Key generation. Let \mathbb{F}_q be a finite field of q elements and id_k be an identity map on \mathbb{F}_q^k for $k \geq 1$. A central map \mathcal{F} of SVSv is defined by $\mathcal{F} = \bar{F} \circ (\hat{\mathcal{F}} \times id_v)$. The map $\hat{\mathcal{F}} : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^o$ is given by $\hat{\mathcal{F}}(\mathbf{x}) = (\hat{f}^{(1)}(\mathbf{x}), \dots, \hat{f}^{(o)}(\mathbf{x}))$ for o randomly chosen affine polynomials $\hat{f}^{(1)}, \dots, \hat{f}^{(o)}$, where $n = o + v + r$ for $o, v, r \in \mathbb{N}$. The map $\bar{\mathcal{F}} : \mathbb{F}_q^o \times \mathbb{F}_q^{v+r} \rightarrow \mathbb{F}_q^o$ is given by $\bar{\mathcal{F}}(\mathbf{x}) = (\bar{f}^{(1)}(\mathbf{x}), \dots, \bar{f}^{(o)}(\mathbf{x}))$, where

$$\begin{cases} \bar{f}^{(1)} = x_1^2 + g_1(y_{o+1}, \dots, y_n), \\ \bar{f}^{(2)} = x_1 \cdot x_2 + g_2(y_{o+1}, \dots, y_{o+v}), \\ \dots \\ \bar{f}^{(o)} = x_{o-1} \cdot x_o + g_o(y_{o+1}, \dots, y_{o+v}), \end{cases}$$

and g_1, \dots, g_o are randomly chosen quadratic polynomials. A public key is given as $\mathcal{P} = \mathcal{T} \circ \mathcal{F} \circ \mathcal{S} : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^o$ for two randomly chosen invertible affine map $\mathcal{T} : \mathbb{F}_q^o \rightarrow \mathbb{F}_q^o$ and $\mathcal{S} : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^n$ to hide the special structure of \mathcal{F} . Then a secret key is $(\mathcal{T}, \mathcal{F}, \mathcal{S})$. The parameter r is defined so that symmetric matrices of the quadratic parts of the polynomials in \mathcal{F} have the same rank:

$$\begin{cases} r = 2 & \text{if } q \equiv 0 \pmod{2} \ \& \ v \equiv 0 \pmod{2}, \\ r = 1 & \text{otherwise.} \end{cases}$$

Sign. Given a message \mathbf{m} , do the followings after computing $\mathbf{h} = h(\mathbf{m}) \in \mathbb{F}_q^o$:

1. Compute $\mathbf{w} = \mathcal{T}^{-1}(\mathbf{h})$.
2. Choose random vinegar values $(y_{o+1}, \dots, y_n) \in \mathbb{F}_q^o$ and substitute them into g_1, \dots, g_o . Compute $x_1 = \sqrt{w_1 - g_1} = \begin{cases} (w_1 - g_1)^{1/2} & q \equiv 1 \pmod{2} \\ (w_1 - g_1)^{q/2} & q \equiv 0 \pmod{2} \end{cases}$. If $x_1 = 0$, choose other random vinegar values. Inductively, compute $x_i = (w_i - g_i)/x_{i-1}$ for $i = 2, \dots, o$. If $x_i = 0$ for $i = 2, \dots, o$, choose other random vinegar values. Solve the linear system given by $\hat{f}^{(1)}, \dots, \hat{f}^{(o)}$. If there is no solution, choose other random vinegar values and try again.
3. Compute $\mathbf{z} = \mathcal{S}^{-1}(y_1, \dots, y_n)$. Then, $\mathbf{z} \in \mathbb{F}_q^n$ is a signature of \mathbf{m} .

Verify. Given (\mathbf{z}, \mathbf{m}) , check $\mathcal{P}(\mathbf{z}) = h(\mathbf{m})$. If it holds, accept the signature, otherwise reject it.

2.2 CSSv

Key generation. A central map $\mathcal{F} : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^o$ of CSSv is defined by $\mathcal{F} = \bar{\mathcal{F}} \circ (\hat{\mathcal{F}} \times id_v)$. The map $\hat{\mathcal{F}}$ is given by $\hat{\mathcal{F}}(\mathbf{x}) = (\hat{f}^{(1)}(\mathbf{x}), \dots, \hat{f}^{(o)}(\mathbf{x}))$, where

$$\begin{cases} \hat{f}^{(1)} = \sum_{i=1}^n \sum_{j=i}^n a_{ij}^{(1)} \cdot x_i x_j + \sum_{i=1}^n b_i^{(1)} \cdot x_i + c^{(1)}, \\ \hat{f}^{(2)} = \sum_{i=1}^n b_i^{(2)} \cdot x_i + c^{(2)}, \\ \vdots \\ \hat{f}^{(o)} = \sum_{i=1}^n b_i^{(o)} \cdot x_i + c^{(o)}, \end{cases}$$

$\hat{f}^{(1)}$ is a random quadratic polynomial and $\hat{f}^{(2)}, \dots, \hat{f}^{(o)}$ are affine maps in variables x_1, \dots, x_n . The map $\bar{\mathcal{F}} : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^o$ is given by $\bar{\mathcal{F}}(\mathbf{x}) = (\bar{f}^{(1)}(\mathbf{x}), \dots, \bar{f}^{(o)}(\mathbf{x}))$, where

$$\begin{cases} \bar{f}^{(1)} = x_1 + g_1(x_{o+1}, \dots, x_n), \\ \bar{f}^{(2)} = x_1 \cdot x_2 + g_2(x_{o+1}, \dots, x_n), \\ \vdots \\ \bar{f}^{(o)} = x_{o-1} \cdot x_o + g_o(x_{o+1}, \dots, x_n), \end{cases}$$

$\bar{f}^{(2)}$ is a cubic polynomial and the other polynomials are quadratic. The central map \mathcal{F} consists of one cubic polynomial $f^{(2)}$ and $(o - 1)$ quadratic polynomials $f^{(1)}, f^{(3)}, \dots, f^{(o)}$. Two invertible affine maps $\mathcal{S} : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^n$ and $\mathcal{T} : \mathbb{F}_q^o \rightarrow \mathbb{F}_q^o$ are used as hide functions, where \mathcal{S} is chosen at random and a matrix T representing \mathcal{T} is

$$T = \begin{pmatrix} \star_{1 \times 1} & \star_{1 \times 1} & \star_{1 \times (o-2)} \\ \star_{(o-1) \times 1} & 0_{(o-1) \times 1} & \star_{(o-1) \times (o-2)} \end{pmatrix} \in \mathbb{F}_q^{o \times o} \tag{1}$$

A public map \mathcal{P} is $\mathcal{P} = (p^{(1)}, \dots, p^{(o)}) = \mathcal{T} \circ \mathcal{F} \circ \mathcal{S} : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^o$. A secret key is $(\mathcal{T}, \mathcal{F}, \mathcal{S})$. Note that $p^{(1)}$ is cubic, the other public polynomials are quadratic due to the structure of \mathcal{T} . Signing and verification of CSSv are similar to those of SVSv.

3 Key Recovery Attacks on CSSv and SVSv

Now, we present key recovery attacks (KRAs) on CSSv and SVSv. In MQ-schemes, there exist a large number of different secret keys for a given public key [16]. Informally, suppose that $\langle \mathcal{P}, (S, \mathcal{F}, T) \rangle$ is a public/secret key pair of an MQ-scheme, we call (S', \mathcal{F}', T') is an equivalent key of (S, \mathcal{F}, T) if $\mathcal{P} = S \circ \mathcal{F} \circ T = S' \circ \mathcal{F}' \circ T'$, where S' and T' are invertible affine maps, and \mathcal{F}' preserves all zero coefficients of \mathcal{F} . If an attacker can find any of the equivalent keys then he can forge signatures on any messages. Thus, the attacker tries to find equivalent keys with a simple structure. Refer to [14–16] for more details on equivalent keys and good keys.

3.1 Key Recovery Attacks on CSSv

Phase 1. Let $h = id_1 \times (h^{(2)}, \dots, h^{(o)}) \times id_v : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^n$ be an affine map, where $h^{(j)} = \hat{f}^{(j)} = \sum_{i=1}^n b_i^{(j)} \cdot x_i + c^{(j)}$ for $2 \leq j \leq o$. We define $\hat{\mathcal{F}}'$ by $(\hat{\mathcal{F}} \times id_v) \circ h^{-1}$ which can be expressed by $\hat{f}'^{(1)} \times id_{n-1}$, where $\hat{f}'^{(1)}(x_1, \dots, x_n)$ is a quadratic polynomial. We set $\mathcal{F}_0 = \bar{\mathcal{F}} \circ \hat{\mathcal{F}}' = (f_0^{(1)}, f_0^{(2)}, f_0^{(3)}, \dots, f_0^{(o)})$ given by

$$\begin{cases} f_0^{(1)} = \hat{f}'^{(1)}(x_1, \dots, x_n) + g_1(x_{o+1}, \dots, x_n), \\ f_0^{(2)} = \hat{f}'^{(1)}(x_1, \dots, x_n) \cdot x_2 + g_2(x_{o+1}, \dots, x_n), \\ f_0^{(j)} = \bar{f}^{(j)} = x_{j-1} \cdot x_j + g_j(x_{o+1}, \dots, x_n), \quad \text{for } 3 \leq j \leq o. \end{cases} \quad (2)$$

The public key is written as

$$\begin{aligned} \mathcal{P} &= \mathcal{T} \circ \mathcal{F} \circ \mathcal{S} = \mathcal{T} \circ \bar{\mathcal{F}} \circ (\hat{\mathcal{F}} \times id_v) \circ \mathcal{S} = \mathcal{T} \circ \bar{\mathcal{F}} \circ (\hat{\mathcal{F}}' \circ h) \circ \mathcal{S} \\ &= \mathcal{T} \circ (\bar{\mathcal{F}} \circ \hat{\mathcal{F}}') \circ (h \circ \mathcal{S}) = \mathcal{T} \circ \mathcal{F}_0 \circ \mathcal{S}', \end{aligned} \quad (3)$$

where $\mathcal{S}' = h \circ \mathcal{S} : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^n$ is also an affine map. Since $f_0^{(2)}$ is the only cubic map in the central map \mathcal{F}_0 in (3), it occurs in only $p^{(1)}$ due to the special structure of \mathcal{T} . Thus, $p^{(1)}$ is the only cubic polynomial in the public map. We set $\mathcal{P}' = (p'^{(1)}, \dots, p'^{(o-1)}) : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^{o-1}$ with $p'^{(j)} = p^{(j+1)}$ for $1 \leq j \leq o-1$ by removing $p^{(1)}$ from the public map. We also remove the first row and the second column of T and $f_0^{(2)}$ of \mathcal{F}_0 . We define T' by a $(o-1) \times (o-1)$ submatrix of T removed the first row and the second column from T and $T' : \mathbb{F}_q^{o-1} \rightarrow \mathbb{F}_q^{o-1}$ by an affine map corresponding T' . We define a quadratic map $\mathcal{F}' = (f'^{(1)}, f'^{(2)}, \dots, f'^{(o-1)}) : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^{o-1}$ by

$$\begin{cases} f'^{(1)} = f_0^{(1)} = \hat{f}'^{(1)}(x_1, \dots, x_n) + g_1(x_{o+1}, \dots, x_n), \\ f'^{(2)} = f_0^{(3)} = x_2 \cdot x_3 + g_3(x_{o+1}, \dots, x_n), \\ \vdots \\ f'^{(o-1)} = f_0^{(o)} = x_{o-1} \cdot x_o + g_o(x_{o+1}, \dots, x_n). \end{cases} \quad (4)$$

Then we get $\mathcal{P}' = T' \circ \mathcal{F}' \circ \mathcal{S}'$ by (3). From now, we mount the attacks on this structure $\mathcal{P}' = T' \circ \mathcal{F}' \circ \mathcal{S}'$.

Phase 2. Our attack consists of $o-2$ steps: in each step N for $N \in \{1, \dots, o-2\}$, we remove the variable x_N from all but the first public polynomial of \mathcal{P}' . This is done by finding a good key \widetilde{S}_N'' and \widetilde{T}_N'' of \mathcal{P}' . At the end of each step, we remove the first public polynomial and put it as $\mathcal{F}''^{(o-N)}$ because it is the only polynomial that contains x_N , and repeat the procedure for the rest of the public polynomials.

Step 1. Now, we want to remove x_1 in the public map. Let $P^{(i)}$ be the i -th symmetric matrix of the quadratic part of the public map \mathcal{P}' . We assume that

dimensions of all public matrices are n . Observe that x_1 occurs in $f^{(1)}$ which is the first polynomial of the central map \mathcal{F}' . Now we denote that

$$T' = \begin{pmatrix} t'_{1,1} & t'_{1,2} & \cdots & t'_{1,j} & \cdots & t'_{1,o-1} \\ t'_{2,1} & t'_{2,2} & \cdots & t'_{2,j} & \cdots & t'_{2,o-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ t'_{j,1} & t'_{j,2} & \cdots & t'_{j,j} & \cdots & t'_{j,o-1} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ t'_{o-1,1} & t'_{o-1,2} & \cdots & t'_{o-1,j} & \cdots & t'_{o-1,o-1} \end{pmatrix}.$$

Then we get $\mathcal{P}' = T' \circ (\mathcal{F}' \circ \mathcal{S}') =$

$$\begin{cases} t'_{1,1} \cdot (f^{(1)} \circ \mathcal{S}') + t'_{1,2} \cdot (f^{(2)} \circ \mathcal{S}') + \cdots + t'_{1,o-1} \cdot (f^{(o-1)} \circ \mathcal{S}') \\ t'_{2,1} \cdot (f^{(1)} \circ \mathcal{S}') + t'_{2,2} \cdot (f^{(2)} \circ \mathcal{S}') + \cdots + t'_{2,o-1} \cdot (f^{(o-1)} \circ \mathcal{S}') \\ \vdots \\ t'_{j,1} \cdot (f^{(1)} \circ \mathcal{S}') + t'_{j,2} \cdot (f^{(2)} \circ \mathcal{S}') + \cdots + t'_{j,o-1} \cdot (f^{(o-1)} \circ \mathcal{S}') \\ \vdots \\ t'_{o-1,1} \cdot (f^{(1)} \circ \mathcal{S}') + t'_{o-1,2} \cdot (f^{(2)} \circ \mathcal{S}') + \cdots + t'_{o-1,o-1} \cdot (f^{(o-1)} \circ \mathcal{S}'). \end{cases}$$

We suggest to find a linear combination of two public polynomials so that $f^{(1)}$ no longer occurs. Without loss of generality, we choose $P^{(1)}$ and $P^{(2)}$. By the basic elimination method, we see that $t'_{2,1} \cdot P^{(1)} - t'_{1,1} \cdot P^{(2)}$ or $P^{(2)} + (-t'_{2,1} \cdot t'_{1,1}^{-1}) \cdot P^{(1)}$ does not contain $(f^{(1)} \circ \mathcal{S}')$. It is related to the following MinRank instance:

$$\text{find } \lambda \in \mathbb{F}_q \text{ such that } \text{Rank} \left(P^{(2)} + \lambda P^{(1)} \right) < o - 1. \tag{5}$$

Then $\lambda = -t'_{2,1} \cdot t'_{1,1}^{-1}$ is a solution of (5) if $t_{1,1} \neq 0$. Observe that $\lambda = -t'_{2,j} \cdot t'_{1,j}^{-1}$ for $2 \leq j \leq o - 1$ are also solutions of (5), but there may be no solution of (5). We will discuss this later. Similarly, $\lambda = -t'_{k,1} \cdot t'_{1,1}^{-1}$ is a solution of the following analogous problem of (5)

$$\text{find } \lambda \in \mathbb{F}_q \text{ such that } \text{Rank} \left(P^{(k)} + \lambda P^{(1)} \right) < o - 1 \tag{6}$$

for each $3 \leq k \leq o - 1$. We denote

$$\widetilde{\mathcal{T}}''_1 = \begin{pmatrix} 1 & 0 \cdots 0 \cdots 0 \\ \widetilde{t''_{2,1}} & 1 \cdots 0 \cdots 0 \\ \vdots & \vdots \cdots \vdots \vdots \vdots \\ \widetilde{t''_{j,1}} & 0 \cdots 1 \cdots 0 \\ \vdots & \vdots \vdots \vdots \cdots \vdots \\ \widetilde{t''_{o-1,1}} & 0 \cdots 0 \cdots 1 \end{pmatrix} := \begin{pmatrix} 1 & 0 \cdots 0 \cdots 0 \\ -t'_{2,1}/t'_{1,1} & 1 \cdots 0 \cdots 0 \\ \vdots & \vdots \cdots \vdots \vdots \vdots \\ -t'_{j,1}/t'_{1,1} & 0 \cdots 1 \cdots 0 \\ \vdots & \vdots \vdots \vdots \cdots \vdots \\ -t'_{o-1,1}/t'_{1,1} & 0 \cdots 0 \cdots 1 \end{pmatrix} \tag{7}$$

then $\mathcal{T}'_1 = \widetilde{\mathcal{T}}''_1 \circ \mathcal{T}'$ is of the form $\mathcal{T}'_1 = \begin{pmatrix} t'_{1,1} & t'_{1,2} & \cdots & t'_{1,o-1} \\ 0 & t'_{2,2} & \cdots & t'_{2,o-1} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & t'_{j,2} & \cdots & t'_{j,o-1} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & t'_{o-1,2} & \cdots & t'_{o-1,o-1} \end{pmatrix}$. We apply \mathcal{T}'_1

to the left side of \mathcal{P}' then $\widetilde{\mathcal{T}}''_1 \circ \mathcal{P}' = \widetilde{\mathcal{T}}''_1 \circ (\mathcal{T}' \circ \mathcal{F}' \circ \mathcal{S}') = (\widetilde{\mathcal{T}}''_1 \circ \mathcal{T}') \circ (\mathcal{F}' \circ \mathcal{S}') = \mathcal{T}'_1 \circ (\mathcal{F}' \circ \mathcal{S}')$. Consequently, only the first polynomial of $\mathcal{T}'_1 \circ (\mathcal{F}' \circ \mathcal{S}')$ contains $(f'^{(1)} \circ \mathcal{S}')$, where $\mathcal{T}'_1 \circ (\mathcal{F}' \circ \mathcal{S}')$ is written as

$$\left\{ \begin{array}{l} t'_{1,1} \cdot (f'^{(1)} \circ \mathcal{S}') + t'_{1,2} \cdot (f'^{(2)} \circ \mathcal{S}') + \cdots + t'_{1,o-1} \cdot (f'^{(o-1)} \circ \mathcal{S}'), \\ t'_{2,2} \cdot (f'^{(2)} \circ \mathcal{S}') + \cdots + t'_{2,o-1} \cdot (f'^{(o-1)} \circ \mathcal{S}'), \\ \vdots \\ t'_{j,2} \cdot (f'^{(2)} \circ \mathcal{S}') + \cdots + t'_{j,o-1} \cdot (f'^{(o-1)} \circ \mathcal{S}'), \\ \vdots \\ t'_{o-1,2} \cdot (f'^{(2)} \circ \mathcal{S}') + \cdots + t'_{o-1,o-1} \cdot (f'^{(o-1)} \circ \mathcal{S}'). \end{array} \right. \quad (8)$$

Now, we remove x_1 in $S'^{(k)}$ for $2 \leq k \leq n$, where $\mathcal{S}' = (S'^{(1)}, S'^{(2)}, S'^{(3)}, \dots, S'^{(n)})$. Observe that we substitute $S'^{(k)}$ with x_k of \mathcal{F}' when we compute $\mathcal{F}' \circ \mathcal{S}'$. If only $f'^{(1)}$ contains x_1 then $S'^{(1)}$ occurs only in the first polynomial of $\mathcal{F}' \circ \mathcal{S}'$, so does x_1 . For it, we want to find $\mathcal{S}'_1 = \mathcal{S}' \circ \widetilde{\mathcal{S}}''_1$ of the form

$$\mathcal{S}'_1 = \begin{pmatrix} s'_{1,1} & s'_{1,2} & \cdots & s'_{1,n} \\ 0 & s'_{2,2} & \cdots & s'_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & s'_{j,2} & \cdots & s'_{j,n} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & s'_{n,2} & \cdots & s'_{n,n} \end{pmatrix} = \mathcal{S}' \cdot \widetilde{\mathcal{S}}''_1, \quad (9)$$

where all but the first column of \mathcal{S}'_1 consist of arbitrary values. We denote $\widetilde{X} =$

$$X^{-1} \text{ for any } X. \text{ So we need to find } \widetilde{\mathcal{S}}''_1 \text{ of the form } \widetilde{\mathcal{S}}''_1 = \begin{pmatrix} \widetilde{s''_{1,1}} & 0 & \cdots & 0 & \cdots & 0 \\ s''_{2,1} & 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ s''_{j,1} & 0 & \cdots & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \widetilde{s''_{n,1}} & 0 & \cdots & 0 & \cdots & 1 \end{pmatrix},$$

where $\widetilde{s''_{j,1}} \in \mathbb{F}_q$ for $1 \leq j \leq n$. From the first column of the second equality of (9), we get

$$\begin{pmatrix} s'_{1,1} \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \mathcal{S}' \cdot \begin{pmatrix} \widetilde{s''_{1,1}} \\ \widetilde{s''_{2,1}} \\ \vdots \\ \widetilde{s''_{j,1}} \\ \vdots \\ \widetilde{s''_{n,1}} \end{pmatrix} \text{ and } \begin{pmatrix} \widetilde{s''_{1,1}} \\ \widetilde{s''_{2,1}} \\ \vdots \\ \widetilde{s''_{j,1}} \\ \vdots \\ \widetilde{s''_{n,1}} \end{pmatrix} = \mathcal{S}'^{-1} \begin{pmatrix} s'_{1,1} \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

If $\widetilde{\mathcal{S}'_{1,1}} = 0$, then $\widetilde{s''_{1,1}} = 0$, so $\widetilde{\mathcal{S}''_1}$ is not invertible. Otherwise, $s'_{1,1}$ can be any nonzero value in \mathbb{F}_q . Without loss of generality, we fix $s'_{1,1} = (\widetilde{s''_{1,1}})^{-1}$ and so $\widetilde{s''_{1,1}} = 1$. Then we get

$$\widetilde{\mathcal{S}''_1} = \begin{pmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 \\ \widetilde{s''_{2,1}} & 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \widetilde{s''_{j,1}} & 0 & \cdots & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \widetilde{s''_{n,1}} & 0 & \cdots & 0 & \cdots & 1 \end{pmatrix}. \tag{10}$$

After applying $\widetilde{\mathcal{S}''_1}$ to the right side of $\widetilde{\mathcal{T}''_1} \circ \mathcal{P}' = \mathcal{T}'_1(\mathcal{F}' \circ \mathcal{S}')$ of (8), we get

$$\begin{aligned} (\widetilde{\mathcal{T}''_1} \circ \mathcal{P}') \circ \widetilde{\mathcal{S}''_1} &= (\mathcal{T}'_1 \circ \mathcal{F}' \circ \mathcal{S}') \circ \widetilde{\mathcal{S}''_1} = (\mathcal{T}'_1 \circ \mathcal{F}') \circ (\mathcal{S}' \circ \widetilde{\mathcal{S}''_1}) = (\mathcal{T}'_1 \circ \mathcal{F}') \circ \mathcal{S}'_1 \\ &= \mathcal{T}'_1 \circ (\mathcal{F}' \circ \mathcal{S}'_1) \end{aligned}$$

so that

$$\mathcal{T}'_1 \circ (\mathcal{F}' \circ \mathcal{S}'_1) = \begin{cases} t'_{1,1} \cdot (f'^{(1)} \circ \mathcal{S}'_1) + t'_{1,2} \cdot (f'^{(2)} \circ \mathcal{S}'_1) + \cdots + t'_{1,o-1} \cdot (f'^{(o-1)} \circ \mathcal{S}'_1) \\ t'_{2,2} \cdot (f'^{(2)} \circ \mathcal{S}'_1) + \cdots + t'_{2,o-1} \cdot (f'^{(o-1)} \circ \mathcal{S}'_1) \\ \vdots \\ t'_{j,2} \cdot (f'^{(2)} \circ \mathcal{S}'_1) + \cdots + t'_{j,o-1} \cdot (f'^{(o-1)} \circ \mathcal{S}'_1) \\ \vdots \\ t'_{o-1,2} \cdot (f'^{(2)} \circ \mathcal{S}'_1) + \cdots + t'_{o-1,o-1} \cdot (f'^{(o-1)} \circ \mathcal{S}'_1). \end{cases}$$

Since x_1 does not occur in $(f'^{(k)} \circ \mathcal{S}'_1)$ for all k except $k = 1$, x_1 occurs in only the first polynomial of $\mathcal{T}'_1 \circ (\mathcal{F}' \circ \mathcal{S}'_1)$. If $\mathcal{F}''_1 = \widetilde{\mathcal{T}''_1} \circ \mathcal{P}' \circ \widetilde{\mathcal{S}''_1} = \mathcal{T}'_1 \circ \mathcal{F}' \circ \mathcal{S}'_1$ then we can see that $(\mathcal{F}''_1, \mathcal{S}''_1, \mathcal{T}''_1)$ is a good key of $(\mathcal{F}', \mathcal{S}', \mathcal{T}')$ (see Fig. 1, (7) and (10)). To find this good key, we need to solve a system of the form

$$\mathcal{F}''_{i,j}^{(k)} = \sum_{a=1}^{o-1} \sum_{b=1}^n \sum_{c=1}^n \mathcal{P}'_{bc}{}^{(a)} \widetilde{t''_{k,a}} \widetilde{s''_{i,b}} \widetilde{s''_{j,c}}. \tag{11}$$

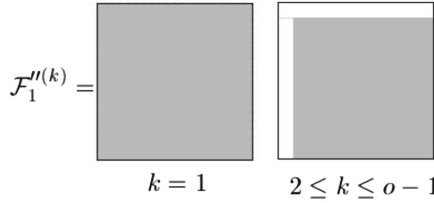


Fig. 1. Form of $\mathcal{F}_1'' = \widetilde{T}_1'' \circ \mathcal{P}' \circ \widetilde{S}_1'' = T_1' \circ \mathcal{F}' \circ S_1'$.

We get equations from $i = 1$ or $j = 1$, but we fix $i = 1$ for convenience. We get $o - 2$ cubic polynomials for $j = 1$, but we ignore the cubic polynomials because we have enough number of quadratic equations. Since $\widetilde{s}_{j,j}'' = 1, \widetilde{s}_{j,c}'' = 0$ ($j \neq c$) and $\widetilde{t}_{k,k}'' = 1, \widetilde{t}_{k,a}'' = 0$ ($a \neq 1, k$), the equations in (11) are changed to

$$\mathcal{F}_{1,j}^{(k)} = \mathcal{P}'_{1j}{}^{(k)} + \mathcal{P}'_{1j}{}^{(1)} \widetilde{t}_{k,1}'' + \sum_{b=2}^n (\mathcal{P}'_{bj}{}^{(k)} + \mathcal{P}'_{bj}{}^{(1)} \widetilde{t}_{k,1}'') \widetilde{s}_{1,b}'' \tag{12}$$

for $2 \leq j \leq n$ and $2 \leq k \leq o - 1$. That is, we get $(n - 1)(o - 2)$ quadratic equations from (12) with variables $\widetilde{t}_{k,1}''$ and $\widetilde{s}_{1,b}''$ for $2 \leq k \leq o - 1$ and $2 \leq b \leq n$. This system is also easily solvable in practice [6]. Thomae considered a further easy method to solve the system (12) in [15]. If we successively recover \mathcal{F}_1'' , we put $\mathcal{F}''^{(1)}$ as the first polynomial of \mathcal{F}_1'' (the only polynomial that x_1 occurs) and remove x_1 from \mathcal{F}_1'' . We denote $\mathcal{P}'_1 : \mathbb{F}_q^{n-1} \rightarrow \mathbb{F}_q^{o-2}$ by a new public map after removing the first polynomial from \mathcal{F}_1'' . We remove the first polynomial of the central map \mathcal{F}' and denote it by \mathcal{F}'_1 . Then we get $\mathcal{P}'_1 = T'_1 \circ \mathcal{F}'_1 \circ S'_1$.

Step $N(2 \leq N \leq o - 2)$. In the previous step $N - 1$, we get $\mathcal{P}'_{N-1} = T'_{N-1} \circ \mathcal{F}'_{N-1} \circ S'_{N-1}$ that x_N only occurs in the first polynomial of \mathcal{F}'_{N-1} . In this step, we also find \widetilde{T}''_N and \widetilde{S}''_N , which have similar forms to (7) and (10), respectively, that the variable x_N occurs only in the first polynomial of $\mathcal{F}''_N = \widetilde{T}''_N \circ \mathcal{P}'_{N-1} \circ \widetilde{S}''_N$, similar to the previous steps. Then we put $\mathcal{F}''^{(N)}$ as the first polynomial of \mathcal{F}''_N which is the only polynomial such that x_N occurs, and remove it from \mathcal{F}''_N . We denote $\mathcal{P}'_N : \mathbb{F}_q^{n-N} \rightarrow \mathbb{F}_q^{o-N-1}$ by a new public map after removing the first polynomial from \mathcal{F}''_N . We remove the first polynomial of the central map \mathcal{F}'_{N-1} and denote x_N by \mathcal{F}'_N . Then we get $\mathcal{P}'_N = T'_N \circ \mathcal{F}'_N \circ S'_N$, where $T'_N = \widetilde{T}''_N \circ T'_{N-1}$ and $S'_N = S'_{N-1} \circ \widetilde{S}''_N$, which is similar to the end of the previous step.

Last Step. After Step $o - 2$, we denote $\mathcal{F}''^{(o-1)}$ by the remaining public polynomials. We define

$$\begin{aligned} \mathcal{F}'' &= (\mathcal{F}''^{(1)}, \dots, \mathcal{F}''^{(o-1)}) : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^{o-1} \\ \overline{T}'' &= \overline{T}''_{o-2} \circ \dots \circ \overline{T}''_1, \quad \overline{S}'' = \overline{S}''_1 \circ \dots \circ \overline{S}''_{o-2}, \end{aligned} \tag{13}$$

where $\overline{S''_N}$ and $\overline{T''_N}$ for $1 \leq N \leq o - 2$ are determined by the following $n \times n$ matrices and $(o - 1) \times (o - 1)$ matrices, respectively, as:

$$\overline{S''_N} = \begin{pmatrix} I_{(N-1) \times (N-1)} & 0_{(N-1) \times (n-N+1)} \\ 0_{(n-N+1) \times (N-1)} & S''_N \end{pmatrix}, \quad \overline{T''_N} = \begin{pmatrix} I_{(N-1) \times (N-1)} & 0_{(N-1) \times (o-N)} \\ 0_{(o-N) \times (N-1)} & T''_N \end{pmatrix}.$$

Let T'' and S'' be affine maps determined by the matrices $(\overline{T''})^{-1}$ and $(\overline{S''})^{-1}$, respectively. Then $(\mathcal{F}'', \mathcal{S}'', T'')$ is an equivalent key of $(\mathcal{F}', \mathcal{S}', T')$.

Our attacks on CSSv are similar to those on MQQ-ENC scheme in [6, 15]. So, complexity for our attack can be estimated similarly as in [6, 15]. By Theorem 3.44 of [15], complexities of our KRAs are $O(q^2mn^4)$, where q is even, and is $O(qmn^4)$, where q is odd. Complexity of solving the system (12) for finding good keys on CSSv is determined by Step 1, because the number of variables for finding good keys in other steps is bounded by the number of variables in Step 1. Complexity of solving a semi-regular (random) system of m quadratic equations in n variables over \mathbb{F}_q by HF5 [1] can be estimated as

$$C_{HF5}(q, m, n) = \min_{k \geq 0} q^k \cdot \mathcal{O} \left(\left(m \cdot \binom{n - k + d_{reg} - 1}{d_{reg}} \right)^\alpha \right),$$

where the degree of regularity d_{reg} is the index of the first non-positive coefficient in the $S_{m,n} = \frac{(1 - z^2)^m}{(1 - z)^n}$ and $2 \leq \alpha \leq 3$ is the linear algebra constant of solving a linear system. In Table 1, we summarize lower bounds ($\alpha = 2$) of the complexities of KRAs using good keys on CSSv for suggested parameters in [4] using HF5 [1].

Table 1. Lower bonds of complexities of our attacks on CSSv for the suggested parameters.

Security level	Parameter (q, o, v)	Claimed security [4]	KRAs using good keys
80	$(2^8, 26, 13)$	2^{80}	2^{50}
100	$(2^8, 34, 17)$	2^{100}	2^{54}
128	$(2^8, 44, 22)$	2^{128}	2^{58}

Phase 3: How to Forge Signatures. We have recovered an equivalent key $(\mathcal{F}'', \mathcal{S}'', T'')$ of \mathcal{P}' i.e., $\mathcal{P}' = T'' \circ \mathcal{F}'' \circ \mathcal{S}''$. To invert the map \mathcal{F}'' , we first choose y_{o+1}, \dots, y_n at random, and plug them into $\mathcal{F}''^{(o-1)}$ to get a univariate quadratic equation with the variable x_o . If we solve it, then plug y_o, \dots, y_n into $\mathcal{F}''^{(o-2)}$ to get a univariate quadratic equation with the variable x_{o-1} . Similar process for $\mathcal{F}''^{(N)}$ to find x_N for $N = o - 3, \dots, 1$. If any of the quadratic equations is not solvable, we choose other values y_{o+1}, \dots, y_n and try again. Since $(\mathcal{F}'', \mathcal{S}'', T'')$ is the equivalent key of \mathcal{P}' not \mathcal{P} , we cannot easily find $\mathcal{F}^{(2)}$ or its linear combination with $\mathcal{F}''^{(k)}$, where $1 \leq k \leq o - 1$ from $p^{(1)}$. Thus, if we succeed to find a valid signature from the equivalent key $(\mathcal{F}'', \mathcal{S}'', T'')$, we should

check that it satisfies the equation $p^{(1)}$. For it, let $\mathbf{d} = (d_1, d_2, \dots, d_o) \in \mathbb{F}_q^o$. We can find $\mathbf{z} \in \mathbb{F}_q^n$ such that $\mathcal{P}'(\mathbf{z}) = (d_2, \dots, d_o) \in F_q^{o-1}$ using the equivalent key $(\mathcal{F}'', \mathcal{S}'', \mathcal{T}'')$. Then z is a valid signature for $(p^{(1)}(\mathbf{z}), d_2, \dots, d_o)$. The probability that $p^{(1)}(\mathbf{z}) = d_1$ is about $\frac{1}{q}$ if we assume that the image of $p^{(1)}$ is uniform.

3.2 Key Recovery Attacks on SVSv and SVSv2

Since the central map of SVSv is very similar to that of CSSv, the attacks on CSSv can be easily applied to SVSv.

Phase 1. Since $(\hat{\mathcal{F}} \times id_v) : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^n$ is an affine map, $\mathcal{S}' = (\hat{\mathcal{F}} \times id_v) \circ \mathcal{S} : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^n$ is also an affine map. Then we see that

$$\mathcal{P} = \mathcal{T} \circ \mathcal{F} \circ \mathcal{S} = \mathcal{T} \circ (\bar{\mathcal{F}} \circ (\hat{\mathcal{F}} \times id_v)) \circ \mathcal{S} = \mathcal{T} \circ \bar{\mathcal{F}} \circ ((\hat{\mathcal{F}} \times id_v) \circ \mathcal{S}) = \mathcal{T} \circ \bar{\mathcal{F}} \circ \mathcal{S}'$$

So, we can consider $\bar{\mathcal{F}}$ as the central map of SVSv.

Phase 2. A main difference between the two schemes is that the central map of CSSv has one cubic polynomial, while that of SVSv has all quadratic polynomials. Unlike the attacks on CSSv, we first remove x_o from the public map using the fact that x_o only occurs in $\bar{f}^{(o)}$. Hence, we do similar process in the previous section in order of decreasing indices of variables, for SVSv. Except the above difference, the attack on SVSv is the same as that on CSSv resulting in the same complexity.

In [3], the authors showed that SVSv was insecure against HighRank attacks and then proposed a modified version, SVSv2. Our KRAs can be easily applied to SVSv2. Unlike SVSv, SVSv2 uses $\mathcal{S} : \mathbb{F}_q^{n+s} \mapsto \mathbb{F}_q^n$ instead of $\mathcal{S} : \mathbb{F}_q^n \mapsto \mathbb{F}_q^n$. For a projection map $\pi_n : \mathbb{F}_q^{n+s} \mapsto \mathbb{F}_q^n$ with $\pi_n(x_1, \dots, x_{n+s}) = (x_1, \dots, x_n)$, we let $\mathcal{F}' : \mathbb{F}_q^{n+s} \mapsto \mathbb{F}_q^o := \bar{\mathcal{F}} \circ \pi_n$, and \mathcal{S}_0 be an affine map in \mathbb{F}_q^{n+s} such that $\mathcal{S} = \pi_n \circ \mathcal{S}_0$. Then $\mathcal{S}' := (\hat{\mathcal{F}} \times id_{v+s}) \circ \mathcal{S}_0$ is also an affine map in \mathbb{F}_q^{n+s} . Since $\mathcal{F}' = \bar{\mathcal{F}} \circ \pi_n$ and $(\hat{\mathcal{F}} \times id_v) \circ \pi_n = \pi_n \circ (\hat{\mathcal{F}} \times id_{v+s})$, we get

$$\begin{aligned} \mathcal{F} \circ \mathcal{S} &= (\bar{\mathcal{F}} \circ (\hat{\mathcal{F}} \times id_v)) \circ \mathcal{S} = (\bar{\mathcal{F}} \circ (\hat{\mathcal{F}} \times id_v)) \circ (\pi_n \circ \mathcal{S}_0) = \bar{\mathcal{F}} \circ ((\hat{\mathcal{F}} \times id_v) \circ \pi_n) \circ \mathcal{S}_0 \\ &= \bar{\mathcal{F}} \circ (\pi_n \circ (\hat{\mathcal{F}} \times id_{v+s})) \circ \mathcal{S}_0 = (\bar{\mathcal{F}} \circ \pi_n) \circ ((\hat{\mathcal{F}} \times id_{v+s}) \circ \mathcal{S}_0) = \mathcal{F}' \circ \mathcal{S}'. \end{aligned}$$

Although the central map of SVSv2 has $v + s$ vinegar variables unlike v vinegar variables of SVSv, the map \mathcal{F}' also has the same vulnerability as in SVSv. Thus, the same attack can be applied to SVSv2.

4 HighRank Attack on CSSv and SVSv

In HighRank attacks, one tries to identify the variables appearing the lowest number of times in the central polynomials. Here, we show that CSSv and SVSv are entirely broken by the HighRank attacks.

4.1 HighRank Attack on CSSv

Now, we present an efficient HighRank attack on CSSv by eliminating an variable which appears in only one polynomial to remove the cubic polynomial in the central map. The central map of CSSv is defined by $\mathcal{F} = \bar{\mathcal{F}} \circ (\hat{\mathcal{F}} \times id_v)$. We let $\hat{\mathcal{F}}' = (y_1, \hat{f}^{(2)}, \dots, \hat{f}^{(o)})$ and $\bar{\mathcal{F}}' = (f^{(1)} \circ (\hat{\mathcal{F}}' \times id_v)^{-1}, f^{(2)} \circ (\hat{\mathcal{F}}' \times id_v)^{-1}, \bar{f}^{(3)}, \dots, \bar{f}^{(o)})$. Then \mathcal{F} can be written as $\mathcal{F} = \bar{\mathcal{F}} \circ (\hat{\mathcal{F}} \times id_v) = \bar{\mathcal{F}}' \circ (\hat{\mathcal{F}}' \times id_v)$. Since $\hat{\mathcal{F}}'$ consists of affine polynomials and \mathcal{S} is affine map, we can get an affine map $\mathcal{S}' = (\hat{\mathcal{F}}' \times id_v) \circ \mathcal{S}$. We let $\mathcal{P}_{\pi_1} = \pi_1 \circ \mathcal{P}$ for $\pi_1(x_1, \dots, x_o) = (x_2, \dots, x_o)$ and $\bar{\mathcal{F}}'_{\pi_2} = \pi_2 \circ \bar{\mathcal{F}}'$ for $\pi_2(x_1, \dots, x_o) = (x_1, x_3, \dots, x_o)$. Note that $\bar{\mathcal{F}}'_{\pi_2} = (f'^{(1)}, f'^{(2)}, \dots, f'^{(o-1)}) : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^{o-1}$ is defined by

$$\begin{cases} f'^{(1)} = f^{(1)} \circ (\hat{\mathcal{F}}' \times id_v)^{-1}, \\ f'^{(2)} = x_2 \cdot x_3 + g_3(x_{o+1}, \dots, x_n), \\ \vdots \\ f'^{(o-1)} = x_{o-1} \cdot x_o + g_o(x_{o+1}, \dots, x_n). \end{cases} \tag{14}$$

We define an affine map \mathcal{T}' satisfying $\pi_1 \circ \mathcal{T} = \mathcal{T}' \circ \pi_2$. Then the matrix T' representing the affine map \mathcal{T}' is an $(o - 1) \times (o - 1)$ sub-matrix of T obtained by removing the first row and the second column of T . Finally, we get

$$\mathcal{P}_{\pi_1} = \pi_1 \circ \mathcal{T} \circ \bar{\mathcal{F}}' \circ \mathcal{S}' = \mathcal{T}' \circ \pi_2 \circ \bar{\mathcal{F}}' \circ \mathcal{S}' = \mathcal{T}' \circ \bar{\mathcal{F}}'_{\pi_2} \circ \mathcal{S}'.$$

We assume that the secret key of CSSv consists of the central map $\bar{\mathcal{F}}'_{\pi_2}$, and two affine maps \mathcal{T}' and \mathcal{S}' . Note that \mathcal{P}_{π_1} is obtained by removing the first component of \mathcal{P} and $\bar{\mathcal{F}}'_{\pi_2}$ is obtained by removing the second component of $\bar{\mathcal{F}}'$. Thus, \mathcal{P}_{π_1} and $\bar{\mathcal{F}}'_{\pi_2}$ have no cubic polynomial as components. We define the matrix $F^{(k)}$ to be the symmetric matrix associated to the homogenous quadratic part of the k -th component of $\bar{\mathcal{F}}'_{\pi_2}$ for $k = 1, \dots, o - 1$. Similarly, we define the matrix $P^{(k)}$ to be the symmetric matrix associated to the homogenous quadratic part of the k -th component of \mathcal{P}_{π_1} for $k = 1, \dots, o - 1$ and S' to be an $n \times n$ matrix determined by the linear part of \mathcal{S}' .

Now, we want to find two matrices \tilde{S} and \tilde{T} for the public key such that

$$S' \cdot \tilde{S} = \begin{pmatrix} \tilde{S}_o & * \\ 0 & * \end{pmatrix} \text{ and } \tilde{T} \cdot T' = \begin{pmatrix} * & * & \dots & * \\ 0 & * & \dots & * \\ \vdots & \ddots & & \\ 0 & * & \dots & * \end{pmatrix}, \text{ where } \tilde{S}_o \text{ is a } o \times o \text{ matrix.}$$

Phase 1. We focus on the variables appearing the lowest number of times in the central map. Note that the variable x_1 appears in only one polynomial of \mathcal{F}'_{π_2} . We can find a linear combination of two public polynomials such that x_1 no longer occurs. This can be done by finding $\lambda \in \mathbb{F}_q$ such that $\text{Rank}(P^{(i)} + \lambda P^{(1)}) < n$ for $i \neq 1$ as in [6]. However, it may happen that $\text{Rank}(P^{(i)} + \lambda P^{(1)})$ remains unchanged for all $\lambda \in \mathbb{F}_q$ in finite fields of even characteristic. This is because that the rank of an alternating bilinear form is always even. To cover all cases, we find $\lambda_1, \lambda_2 \in \mathbb{F}_q$ simultaneously such that

$$\text{Rank}((P^{(2)} + \lambda_1 P^{(1)}) || (P^{(3)} + \lambda_2 P^{(1)})) < n. \quad (15)$$

Let M be the $n \times 2n$ matrix $(P^{(2)} + \lambda_1 P^{(1)}) || (P^{(3)} + \lambda_2 P^{(1)})$. Then M is not alternating. After finding $\lambda_1, \lambda_2 \in \mathbb{F}_q$ satisfying Eq. (15), we compute the kernel of M whose elements are the parts of \tilde{S} . This kernel can be used to find another $\lambda_i \in \mathbb{F}_q$ such that

$$\text{Rank}((P^{(4)} + \lambda_3 P^{(1)}) || \dots || (P^{(o-1)} + \lambda_{o-2} P^{(1)})) < n. \quad (16)$$

For $v \in \text{Ker}(M)$, there exist λ_i for $i = 3, \dots, o-2$ such that

$$v \cdot (P^{(4)} + \lambda_3 P^{(1)}) || \dots || (P^{(o-1)} + \lambda_{o-2} P^{(1)}) = 0 \quad (17)$$

since the variable x_1 appears in only one polynomial of $\tilde{\mathcal{F}}_{\pi_2}$. Equation (17) implies that $v \cdot P^{(1)}$ is dependent to $v \cdot P^{(i)}$ for $i = 4, \dots, o-1$. Hence we can get λ_i for $i = 3, \dots, o-2$ from $(v \cdot P^{(1)}, v \cdot P^{(j)})$ for $j = 4, \dots, o-1$. We denote $P_j^{(i+1)} + \lambda_{ij} P_j^{(1)}$ by $P_{j+1}^{(i)}$ for $i = 1, \dots, o-2$ and $j \geq 0$, where $P_0^{(i)} = P^{(i)}$. Then $(S'^{\top})^{-1} \cdot P_1^{(i)} \cdot S'^{-1}$ are a linear combination of $F^{(2)}, \dots, F^{(o-1)}$. Note that the variables x_2 and x_o appear in only one polynomial of $F^{(2)}, \dots, F^{(o-1)}$. Hence we find $\lambda_1, \lambda_2 \in \mathbb{F}_q$ such that

$$\text{Rank}((P_1^{(2)} + \lambda_1 P_1^{(1)}) || (P_1^{(3)} + \lambda_2 P_1^{(1)})) < n - 1. \quad (18)$$

Let $M_i = (P_i^{(2)} + \lambda_1 P_i^{(1)}) || (P_i^{(3)} + \lambda_2 P_i^{(1)})$ for $i \geq 0$, where $M_0 = M$. Then we have $\text{Ker}(M_i) \subset \text{Ker}(M_{i+1})$. We solve Eq. (17) for $P_1^{(1)}$ using $v \in \text{Ker}(M_2) \setminus \text{Ker}(M_1)$. In this case, there are two different solutions $(\lambda_1, \dots, \lambda_{o-3})$ since we can eliminate $F^{(2)}$ or $F^{(o-1)}$ to get $P_2^{(i)}$. We take one solution to give a bigger $\text{Ker}(M_i)$ than the previous kernel. We repeat this process until the dimension of $\text{Ker}(M_i)$ becomes o . Finally, we extend the column vectors of \tilde{S} to a basis

of \mathbb{F}_q^n and let $\tilde{T} = \begin{pmatrix} 1 & 0 \\ \lambda^{\top} & I_{o-2} \end{pmatrix}$ where I_{o-2} is the identity matrix of order $o-2$

and $\lambda = (\lambda_1, \dots, \lambda_{o-2})$ such that $P_1^{(k-1)} = P_0^{(k)} + \lambda_{k-1} P_0^{(1)}$ for $k = 2, \dots, o-1$. Then complexity of our attack is $O(n^{3w})$, where $2 \leq w < 3$ is the linear algebra constant. It follows from Theorem 5 in [6].

Phase 2: How to Forge Signatures. We get k -th components of $\tilde{T} \circ \mathcal{P}_{\pi_1} \circ \tilde{S}$ for $k = 2, \dots, o-1$ as

$$\tilde{S}^{\top} \cdot (S'^{\top} \cdot \sum_{j=2}^n \mathbf{t}_{kj} F^{(j)} \cdot S') \cdot \tilde{S} = \begin{pmatrix} \tilde{F}_1^{(k)} & * \\ 0 & \tilde{F}_2^{(k)} \end{pmatrix}.$$

Thus, we get

$$\tilde{F}_1^{(k)} = \tilde{S}_o^{\top} \cdot \sum_{j=2}^n \mathbf{t}_{kj} F_1^{(k)} \cdot \tilde{S}_o \text{ and } \tilde{F}_2^{(k)} = \tilde{S}_o^{\top} \cdot \sum_{j=2}^n \mathbf{t}_{kj} F_2^{(k)} \cdot \tilde{S}_o$$

and separate $F_1^{(k)}$ and $F_2^{(k)}$ in $F^{(k)}$. Since $F_1^{(k)}$ consists of sparse polynomials, we can solve $\tilde{F}(x) = (\tilde{F}_1^{(1)}(x), \dots, \tilde{F}_1^{(o)}(x)) = \delta$ for $\delta \in \mathbb{F}_q^o$ using Gröbner basis method. Algorithm 1 summarizes how to forge signatures of CSSv.

Algorithm 1. Signature Forgery of CSSv

INPUT: \tilde{S}, \tilde{T} obtained from the HighRank attack, a public key \mathcal{P} and a message \mathbf{m}

OUTPUT: \mathbf{z} such that $\mathcal{P}(\mathbf{z}) = h(\mathbf{m})$

- 1: Randomly choose $(e_1, \dots, e_v) \in \mathbb{F}_q^v$.
 - 2: Let $x = (x_1, \dots, x_o, e_1, \dots, e_v)$ and $\tilde{T}' = \begin{pmatrix} 1 & \mathbf{0} \\ \mathbf{0}^\top & \tilde{T} \end{pmatrix}$, where $\mathbf{0}$ is an $1 \times (o - 1)$ zero matrix.
 - 3: Using Gröbner basis method, compute a solution μ of $(\tilde{T}' \circ \mathcal{P} \circ \tilde{S})(x) = \tilde{T}'(h(\mathbf{m}))$.
 - 4: Compute $\mathbf{z} = \tilde{S}(\mu)$ which is a signature of \mathbf{m} .
-

Practical Attack. We implement this attack on CSSv with MAGMA v 2.19-10 which contains an efficient implementation of Faugère’s F4 algorithm using Gröbner basis [5]. The experiments are performed on Intel Xeon E5-2687W CPU 3.1 GHz with 256 GB RAM. The running times in Table 2 are the whole time to mount the High-Rank attack including the time to solve each quadratic system by using Gröbner basis. In Table 2, d_{reg} is the largest degree appearing during the computation of Gröbner basis.

Table 2. Results of practical attacks on CSSv for the suggested parameters.

CSSv(q, o, v)	($2^8, 26, 13$)	($2^8, 34, 17$)	($2^8, 44, 22$)
Security level	80	100	128
d_{reg}	3	3	3
Time (second)	6.53	24.97	122.24

4.2 HighRank Attack on SVSv

The same HighRank attack can be easily mounted to SVSv since the central map of SVSv is very similar to that of CSSv, moreover its central map consists of quadratic polynomials. In SVSv, x_i for $1 \leq i \leq o - 1$ appears only two polynomials, $(\tilde{f}^{(i)}, \tilde{f}^{(i+1)})$ in the central map $\hat{\mathcal{F}}$ and x_o appears only one polynomial $\tilde{f}^{(o)}$. According to the result in [17], to achieve a 128-bit security level, the lowest number of times of all the variables in the central polynomials should be appeared in more than 15 polynomials. In [3], the authors proposed an improved version of SVSv, SVSv2, for preventing the HighRank attack. However, SVSv2 is still insecure against the attack since they modified SVSv by replacing two affine maps maintaining the central map $\hat{\mathcal{F}}$. Thus, they have the same variables appearing the lowest number of times in the central polynomials only once or twice.

5 Conclusion

We have shown that the two MQ-signature schemes, CSSv and SVSv are insecure against the key recovery attacks using good keys and the HighRank attacks. In particular, we have succeeded to forge signatures of CSSv on any message at an 128-bit security level in 2 min by using the HighRank attacks.

References

1. Battale, L., Faugère, J.C., Perret, L.: Solving polynomial systems over finite fields: improved analysis of the hybrid approach. In: ISSAC 2012, pp. 67–74. ACM (2012)
2. Ding, J., Schmidt, D.: Rainbow, a new multivariable polynomial signature scheme. In: Ioannidis, J., Keromytis, A., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 164–175. Springer, Heidelberg (2005). https://doi.org/10.1007/11496137_12
3. Duong, D.H., Petzoldt, T.A., Wang, Y., Takagi, T.: Revisiting the Cubic UOV signature scheme, Cryptology ePrint Archive: Report 2016/1079
4. Duong, D.H., Petzoldt, A., Wang, Y., Takagi, T.: Revisiting the Cubic UOV signature scheme. In: Hong, S., Park, J.H. (eds.) ICISC 2016. LNCS, vol. 10157, pp. 223–238. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-53177-9_12
5. Faugère, J.C.: A new efficient algorithm for computing Gröbner bases (F4). *J. Pure Appl. Algebra* **139**, 61–88 (1999)
6. Faugère, J.-C., Gligoroski, D., Perret, L., Samardjiska, S., Thomae, E.: A polynomial-time key-recovery attack on MQQ cryptosystems. In: Katz, J. (ed.) PKC 2015. LNCS, vol. 9020, pp. 150–174. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46447-2_7
7. Hashimoto, Y.: On the security of Cubic UOV. IACR eprint archive. <http://eprint.iacr.org/2016/788>
8. Kipnis, A., Patarin, J., Goubin, L.: Unbalanced oil and vinegar signature schemes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 206–222. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_15
9. Matsumoto, T., Imai, H.: Public quadratic polynomial-tuples for efficient signature-verification and message-encryption. In: Barstow, D., et al. (eds.) EUROCRYPT 1988. LNCS, vol. 330, pp. 419–453. Springer, Heidelberg (1988). https://doi.org/10.1007/3-540-45961-8_39
10. Nie, X., Liu, B., Xiong, H., Lu, G.: Cubic unbalance oil and vinegar signature scheme. In: Lin, D., Wang, X.F., Yung, M. (eds.) Inscrypt 2015. LNCS, vol. 9589, pp. 47–56. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-38898-4_3
11. Patarin, J., Courtois, N., Goubin, L.: QUARTZ, 128-bit long digital signatures. In: Naccache, D. (ed.) CT-RSA 2001. LNCS, vol. 2020, pp. 282–297. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45353-9_21
12. Petzoldt, A., Chen, M.-S., Yang, B.-Y., Tao, C., Ding, J.: Design principles for HFEv-based multivariate signature schemes. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015 Part I. LNCS, vol. 9452, pp. 311–334. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48797-6_14
13. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.* **26**, 1484–1509 (1997)
14. Thomae, E.: A generalization of the rainbow band separation attack and its applications to multivariate schemes, IACR Cryptology ePrint Archive (2012). <http://eprint.iacr.org/2012/223>

15. Thomae, E.: About the security of multivariate quadratic public key schemes, Dissertation thesis by Dipl. math. E. Thomae, RUB (2013)
16. Wolf, C., Preneel, B.: Large superfluous keys in Multivariate Quadratic asymmetric systems. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 275–287. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-30580-4_19
17. Yang, B.-Y., Chen, J.-M.: Building secure tame-like multivariate public-key cryptosystems: the new TTS. In: Boyd, C., González Nieto, J.M. (eds.) ACISP 2005. LNCS, vol. 3574, pp. 518–531. Springer, Heidelberg (2005). https://doi.org/10.1007/11506157_43
18. Yasuda, T., Ding, J., Takagi, T., Sakurai, K.: A variant of rainbow with shorter secret key and faster signature generation. In: AsiaPKC, pp. 57–62 (2013)

Network and System Security



Evaluating the Impact of Juice Filming Charging Attack in Practical Environments

Weizhi Meng¹(✉), Wang Hao Lee², Zhe Liu³, Chunhua Su⁴, and Yan Li⁵

¹ Department of Applied Mathematics and Computer Science,
Technical University of Denmark, Kongens Lyngby, Denmark
weme@dtu.dk

² Infocomm Security Department, Institute for Infocomm Research,
Singapore, Singapore

³ APSIA, Interdisciplinary Centre for Security, Reliability and Trust,
University of Luxembourg, Luxembourg, Luxembourg

⁴ Division of Computer Science, University of Aizu, Aizuwakamatsu, Japan

⁵ Advanced Digital Sciences Center, Singapore, Singapore

Abstract. Nowadays, smartphones are widely adopted in people's daily lives. With the increasing capability, phone charging has become a basic requirement and a large number of public charging facilities are under construction for this purpose. However, public charging stations may open a hole for cyber-criminals to launch various attacks, especially charging attacks, to steal phone user's private information. Juice filming charging (JFC) attack is one such threat, which can refer users' sensitive information from both Android OS and iOS devices, through automatically monitoring and recording phone screen during the whole charging period. Due to the potential damage of JFC attacks, there is a need to investigate its influence in practical scenarios. Motivated by this, in this work, we firstly conduct a large user survey with over 2500 participants about their awareness and attitude towards charging attacks. We then for the first time investigate the impact of JFC attack under three practical scenarios. Our work aims to complement the state-of-the-art and stimulate more research in this area.

Keywords: Smartphone privacy · Android and iOS · Video recording Charging station · Juice filming charging attack · Practical evaluation

1 Introduction

Mobile devices are widely adopted by millions of people, in which the number of smartphone users is forecast to grow from 1.5 billion in 2014 to around 2.5 billion in 2019. International Data Corporation (IDC) reported that phone shipments grew 5.3% from 344.7 million in the second quarter of 2016, and vendors shipped a total of 362.9 million smartphones worldwide in the third quarter of 2016 [6].

Current smartphones are able to provide various tasks, so that more and more users are likely to store their personal and private data on the phones. Due to the increasing capability, people are often using smartphones in their daily lives (e.g., playing gaming app, video-chatting with friends), which may greatly increase the demand of recharging their mobile devices. To meet this requirement, more and more public charging stations are under construction.

For instance, Singapore Power (SP) promised to deploy up to 200 free mobile charging stations for SG50 [24]. These stations will be launched progressively in various busy locations including hospitals, tertiary institutions, libraries and supermarkets, and will become available in one to two years. In particular, each station will be equipped with 10 individual slots, which contain multiple charging connectors such as mini and micro USBs that can fit most mobile phones and tablets. These charging facilities can greatly benefit smartphone users; however, they may also expose a big threat on smartphone privacy and security, since we are not sure that these charging facilities are not maliciously controlled by cyber-criminals (e.g., charging station developers and managers, Government agencies). For example, Lau *et al.* [8] in 2013 presented *Mactans*, a malicious charger that can launch malware injection attacks using BeagleBoard after users connect their phones to the charger. Spolaor *et al.* [23] proposed *PowerSnitch*, a malicious application that can refer users' data by analyzing power consumption over a USB charging cable during the charging period. As a result, there is a significant need to pay more attention to the defence of charging threats.

Mactans and *PowerSnitch* can work on either iOS or Android devices, while a scalable charging attack was developed by Meng *et al.*, called juice filming charging (JFC) attack, which can be effective on both Android and iOS platforms [16]. This attack can steal users' sensitive information through automatically monitoring and recording phone screen (including users' input) during the period of phone charging, as long as people keep charging and interacting with their phones. Moreover, such attack can be launched automatically by integrating with OCR technology [16]. As JFC attack does not install any piece on phone's side or require any permission from users, it may have a large impact on users' privacy and increase the difficulty of detection. Previous studies have verified that current anti-virus software are unable to detect JFC attacks [15, 16].

Contributions. In literature, several simulated scenarios had been investigated regarding charging threat, but there is no real evaluation under practical environments. Due to the potential damage of charging attacks, in this work, we focus on JFC attack and conduct an empirical study to investigate its influence in three practical environments for the first time. In particular, we conduct a large survey to study users' attitude towards charging attacks and investigate the influence of JFC attack based on practical setup. The contributions of our work can be summarized as below.

- We conduct a large survey with over 2500 participants to explore users' attitude and awareness towards charging attacks. There are two ways to distribute the questions: online form and paper form. The collected results

describe a security concern that most phone users are not aware of charging threat.

- We then introduce how to launch JFC attack with a cloud and investigate its practical impact on users' privacy in three practical locations. To our knowledge, this is the first work that evaluates the influence of JFC attack in real scenarios. We are particularly interesting in the number and the total size of collected videos, which determine how much information can be extracted.

The remaining parts are organized as follows. Section 2 introduces the background of JFC attack. Section 3 describes our survey and analyzes the obtained results. In Sect. 4, we describe how to setup JFC attack in real scenarios and investigate its practical influence. We discuss related studies in Sect. 5 and conclude this work in Sect. 6.

2 Background

JFC attack is able to steal users' private information through automatically video-capturing phone screens when users are playing their phones (or phone screen awake) during the whole charging period [15]. This attack does not need to install any additional parts or ask for any permissions on phone's and user's side. By integrating with OCR technology, JFC attack can provide seven features: (1) can be easy to implement but quite efficient; (2) with less user awareness; (3) does not need to install any additional apps or components on phones; (4) does not need to ask for any permissions; (5) be hard to be detected by current anti-malware software; (6) can be scalable and effective on both Android OS and iOS devices; and (7) can automatically handle collected videos and extract information.

Threat model. There are two basic assumptions: (1) phone charging is a basic and common demand for smartphone users, and (2) most smartphone users would not treat public chargers as highly sensitive or dangerous. It is not hard to observe that many smartphone users charge their phones in public places such as airports, subways, shops and so on. Generally, charging attacks can be divided into either *public* or *private*. In particular, a public charging attack works mainly based on a public charger like charging interfaces provided by airports, while a private charging attack often utilizes a private charger from friends or other familiar persons.

Basic idea. The design of JFC attack is based on the observation that no permission would be asked when plugging iPhones or Android phones to a projector, but the projector can automatically display the phone screen. In addition, there are no compelling notification on the screen when the device is being plugged, or the indicators are very small and last only few seconds. Taking advantage of these, JFC attack can automatically video-record users' inputs by using a VGA/USB interface. This attack reveals that the phone display can be leaked

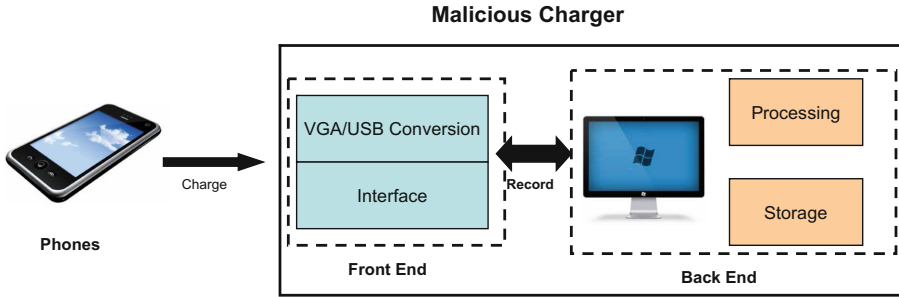


Fig. 1. The high-level architecture of juice filiming charging attack.

through a standard micro USB connector that uses the Mobile High-Definition Link (MHL) standard. For iPhones, the lightning connector is used.

The high-level setup of JFC attack is depicted in Fig. 1. When users connect their phones to JFC charger facilities, the phone screens can be video-captured into video files in the back-end. These collected sensitive videos can be stored and processed to extract private information.

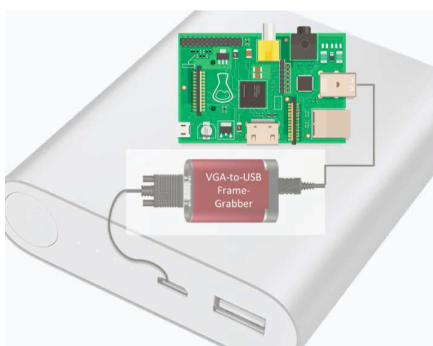
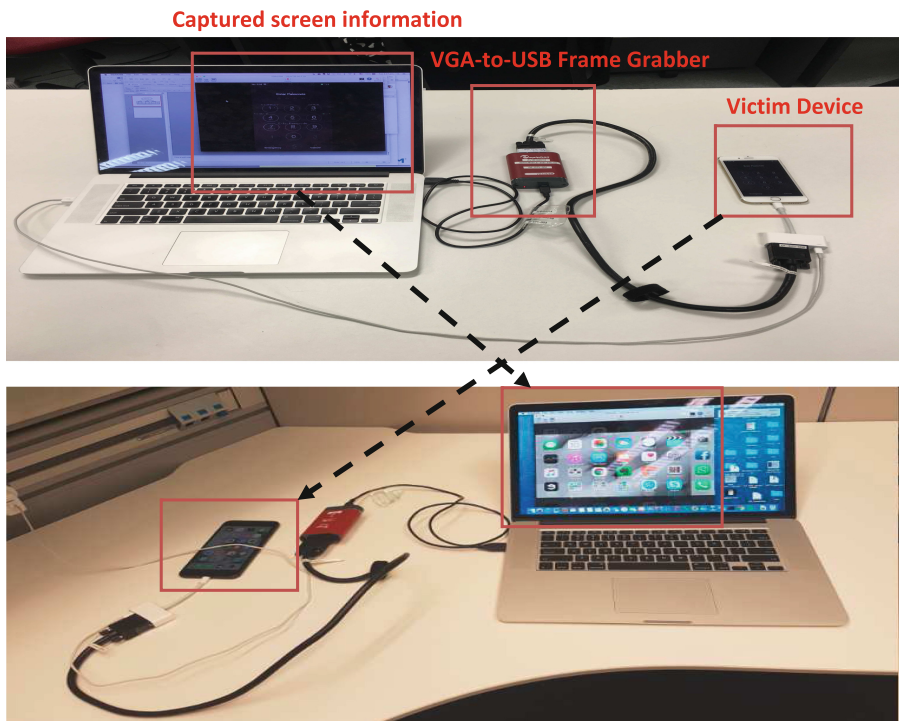
Real setup. To implement JFC attack, choosing an appropriate VGA/USB interface is critical as there are many alternatives online. The previous studies like [15, 16] employed a hardware interface called *VGA2USB* from Epiphan, which is particularly a full-featured VGA/RGB frame grabber, and is responsible for sending a digitized video signal from VGA to USB.¹

The real setup is shown in Fig. 2: the connected iPhone screen could be captured in the computer end. It is easy to imagine that all phone screen information would be captured by JFC attack including users' inputs such as typed passwords, PIN code, email address, used application types and so on. It is worth noting that the hardware interface and other cables can be replaced by smaller devices or hidden by a power bank, which only provides an external charging cable as shown in Fig. 3: Fig. 3(a) shows how to construct a JFC-based power bank (e.g., [16]) and Fig. 3(b) describes a charger box that can be used to launch JFC attack.²

Collected private information. In Fig. 4, we present several images of collected phone screen via JFC attack. In particular, Fig. 4(a) shows the captured screen for inputting a 6-digit PIN on an iPhone, Fig. 4(b) presents the captured screen of bank login, and Fig. 4(c) shows the captured screen of Line chat. These examples indicate that various information can be extracted by analyzing the recorded videos, and that JFC attack may become a big threat for smartphone privacy and security.

¹ <http://www.epiphan.com/products/vga2usb/>.

² <http://www.coolthings.com/life-spot-smartphone-charging-station/>.

**(a)****(b)****Fig. 3.** (a) The construction of JFC-based power bank and (b) a charger box.

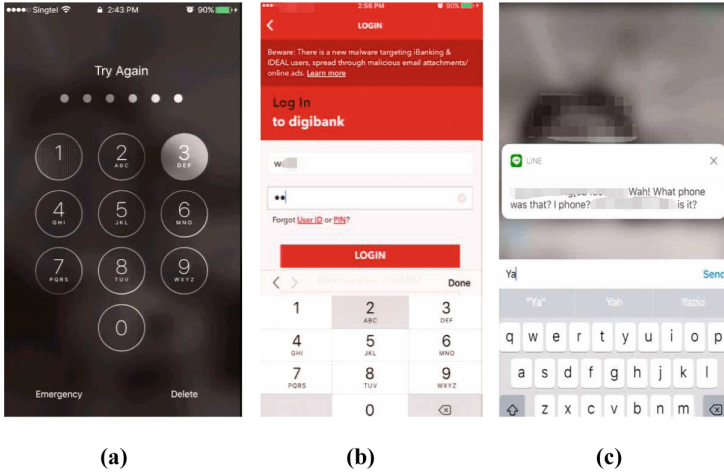


Fig. 4. Collected private information by JFC attack. (a) PIN Input, (b) Bank Login, and (c) Line Chat.

3 User Awareness

User awareness is a critical factor that affects the impact of a security threat such as malware spread, spam and charging threat. In this section, we conduct a large survey including over 2500 participants, with the purpose of investigating users' awareness and attitude towards malware, charging attacks and public charger usage. The survey results aim to complement existing studies like [15, 16].

Participants. We have two ways to distribute our questions through either online form or paper form. A total of 2570 participants attended our study and gave their feedback including students, engineers, professors, researchers, teachers, business people and senior people. All participants are volunteers and have no security background (i.e., without attending any security related courses before). They are aged from 18 to 65 and the detailed information of participants is summarized in Table 1. It is worth noting that up to 64.2% of them are currently using Android phones and 1783 participants were distributed by an online form.

Survey results. The main survey questions and users' feedback are summarized in Table 2. It is noticeable that 1253 (48.8%) of the participants had installed one kind of anti-virus software on their smartphones. Encouragingly, there are 1082 (42.1%) participants can specify the name of at least one smartphone malware. Similar to the previous study [16], these two questions present that common smartphone users have paid more attention to defend against malware (i.e., nearly half of them had installed a security mechanism such as anti-virus to protect their phones from malicious applications).

For the questions regarding smartphone charging, it is found that 1768 (68.8%) of the participants had the need to recharge their phones in public places.

Table 1. Information of participants in the study.

Occupation	Male	Female
Students	773	801
Engineers	108	115
Professors/teachers	52	68
Researchers	101	130
Business people	102	91
Senior people	119	110

Table 2. User feedback in the survey about malware, charging threat and charger usage.

Questions	# of Yes	# of No
Have you installed any anti-virus software on your smartphones?	1253	1317
Can you specify any kind of smartphone malware?	1082	1488
Do you have the need to charge your phone in public places like airport?	1768	802
Are you willing to use a public charging station (e.g., in airport, shops)?	1455	1115
Do you have the potential to interact with your phone during charging like chatting with friends?	1678	892
Do you know any charging attack (i.e., attacks through a USB charging cable)?	582	1988

With the increase of phone usage, this number seems to continue increasing. Due to the demand of charging, 1455 (56.6%) of the participants adopt the use of a public charging station in several places (e.g., in shops, airports). During the charging period, up to 1678 (65.2%) participants reported that they were likely to interact with the phone. For example, they may check their emails and chat with their friends or family members. Unfortunately, 1988 (77.3%) of them were not aware of charging attacks.

Overall, the survey results demonstrate that users would pay less attention to smartphone charging threat as compared to malicious applications (malware); thus, charging attacks have a large potential to cause more victims than malicious applications due to the lack of user awareness in practice. These observations are in line with the observations in former studies [15, 16].

4 Practical Impact

According to the above survey results, JFC attack has the potential to collect users' private information in large. In this section, different from the former

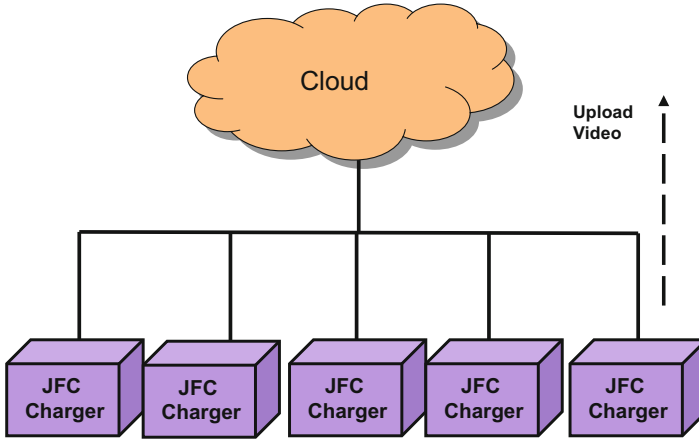


Fig. 5. The high-level deployment for JFC chargers.

research [15,16], our motivation is to investigate the impact of JFC attack in practical environments. We have two particular interests:

- The number of videos that could be collected from JFC attacks each day.
- The total size of recorded video that could be extracted for private information each day.

Deployment. To launch JFC attack in practical scenarios, we seek approval and collaborated with three organizations: a company (with over 200 personnel), a university and a business hall. In particular, we deployed five JFC chargers for each environment, where the chargers can keep uploading the recorded videos to a cloud in the back-end, as shown in Fig. 5. After uploading the videos successfully, the chargers can delete the corresponding videos locally in order to save space for new videos. The back-end was capable of 250 G hardware space, where one minute-video may need 30M space. The video processing with OCR technology can be referred to [16]. The deployed location for each environment is described as below.

- *Company environment.* Five chargers were deployed in one main dining room, where most personnel would spend their time having breakfast, lunch and even dinner.
- *University environment.* Five chargers were deployed at the ground floor of two teaching buildings, so that students can use when they have a rest.
- *Business hall environment.* Five chargers were deployed around the hall, so that visitors can use when queuing up (i.e., waiting for the number).

Data Collection and Results. To protect users' privacy, we seek users' approval and all data will be deleted after processing. At least one IT administrator helped monitor the whole process and make sure all steps are correct.

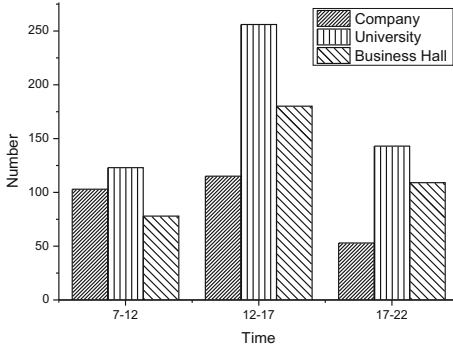


Fig. 6. The average number of collected videos for five days.

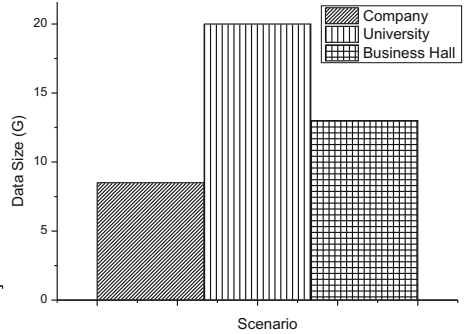


Fig. 7. The average size of collected videos for five days.

Table 3. Extracted information in practical environments.

User information	User information
Android unlock pattern/PIN for iPhones	Gmail account and content
Other email account and content (e.g., Sina, 163)	Social networking account (e.g., Facebook, Twitter, Wechat, QQ)
Bank account and bank message	Visited website content
Social networking chat history (e.g., Facebook, Twitter, Wechat, QQ)	Installed mobile applications
Email passwords (web-login)	Phone number list
Smartphone settings	Personal photos

We performed JFC attack in each environment for five days (i.e., from Monday to Friday). The average number of recorded videos is shown in Fig. 6. The opening hours for both the company and the business hall are from 8 am to 8 pm, so that we mainly recorded the information from 7 am to 10 pm by considering the university environment. The figure shows that JFC chargers can collect more than 250, 500 and 350 videos each day for company environment, university environment and business hall environment, respectively. More specifically, it is found that JFC chargers could collect the highest number of videos during the time period of 12–17. This is because most of their phones were out of power due to the usage in the morning, and there is a high possibility of charging their phones during this period.

Intuitively, each video has a different length and size, in which a longer video may provide more private information about a smartphone user. As a result, one of our interests is to explore the total size of collected videos. Figure 7 depicts the average total size of collected videos for each environment. It is noticeable that JFC chargers could collect 8.5 G, 20 G and 13 G data each day for company environment, university environment and business hall environment, respectively. From these collected videos, we can extract a large amount of private infor-

mation about users, as shown in Table 3, such as Android unlock pattern, PIN for iPhones, Email account and content, social networking chat history, visited website, personal photos and so on. On the whole, our results validate that JFC attack can make a large impact on smartphone users' privacy. There is a need to increase users' awareness towards such attack.

5 Related Work

In literature, physical side channel is believed to be an effective method to infer users' private information and data. Such attacks are often based on oily residues left on the touchscreen. Aviv *et al.* [1] had explored the feasibility of smudge attacks on touch screens. They considered different lighting angles and light sources and the results indicated that the pattern could be partially identifiable in 92% and fully in 68% of the tested lighting and camera settings. Zhang *et al.* [27] proposed a fingerprint attack against tapped passwords via a keypad instead of graphical passwords. Their experiments on various platforms including iPad, iPhone and Android phone demonstrated that the attack can reveal more than 50% of the passwords in most cases. Raguram *et al.* [21] presented that automated reconstruction of text typed on a mobile device's virtual keyboard is possible via compromising reflections such as those of the phone in the user's sunglasses. Their results showed that their approach could reconstruct fluent translations of the recorded data.

Charging attacks are often ignored by phone users. To our knowledge, Lau *et al.* [8] designed *Mactans*, an early malicious charger that used BeagleBoard to conduct malware injection on iOS smartphones. However, a major drawback is that their attack requires users to unlock the phone screen and install developer licenses in advance. Spolaor *et al.* [23] described *PowerSnitch*, a malicious application that can refer personal data on smartphones by analyzing power consumption over a USB charging cable. The scalability is a major limitations for this charging attack. Juice filming charging (JFC) attack [15,16] is a scalable charging attack, which works on both Android and iOS devices, and can record screen information during the whole charging period, without the need to request any permission or action to unlock phone screen. In this work, we conduct an empirical study to investigate the impact of JFC attack in practical environments. Some other related work can be referred to [17–19].

6 Conclusion

As compared to mobile malicious applications, charging threats are often ignored by the literature. In this paper, we focus on juice filming charging (JFC) attack, which has the capability of referring users' private data from both Android OS and iOS devices, through automatically monitoring and recording phone screen during the charging period. The rationale is that screen information can be leaked through a standard micro USB connector that employs the Mobile High-Definition Link (MHL) standard.

Due to the potential damage of charging threat, we focus on JFC attack and perform an empirical study for the first time to investigate the impact of JFC attacks in practical environments. In particular, we conduct a user survey with over 2500 participants about their awareness and attitude towards charging attacks, and then investigate the impact of JFC chargers in three practical scenarios like company environment, university environment and business hall. The results validate that JFC attack would have a large impact on smartphone users' privacy. Our work aims to stimulate more research in this area and raise user awareness of such threat.

Acknowledgment. We would like to thank all participants for their efforts made in the survey and the collaborating organizations for assisting the real deployment and evaluation.

References

1. Aviv, A.J., Gibson, K., Mossop, E., Blaze, M., Smith, J.M.: Smudge attacks on smartphone touch screens. In: Proceedings of the 4th USENIX Conference on Offensive Technologies (WOOT), pp. 1–7. USENIX Association, Berkeley (2010)
2. Dagon, D., Martin, T., Starner, T.: Mobile phones as computing devices: the viruses are coming!. *IEEE Pervasive Comput.* **3**(4), 11–15 (2004)
3. De Luca, A., Hang, A., Brudy, F., Lindner, C., Hussmann, H.: Touch me once and i know it's you! Implicit authentication based on touch screen patterns. In: Proceedings of the 2012 ACM Annual Conference on Human Factors in Computing Systems (CHI), pp. 987–996. ACM, New York (2012)
4. Feng, T., Liu, Z., Kwon, K.-A., Shi, W., Carbutary, B., Jiang, Y., Nguyen, N.: Continuous mobile authentication using touchscreen gestures. In: Proceedings of the 2012 IEEE Conference on Technologies for Homeland Security (HST), pp. 451–456. IEEE, USA (2012)
5. Frank, M., Biedert, R., Ma, E., Martinovic, I., Song, D.: Touchalytics: on the applicability of touchscreen input as a behavioral biometric for continuous authentication. *IEEE Trans. Inf. Forensics Secur.* **8**(1), 136–148 (2013)
6. IDC: Smartphone Momentum Still Evident with Shipments Expected to Reach 1.2 Billion in 2014 and Growing 23.1% Over 2013. <http://www.idc.com/getdoc.jsp?containerId=prUS24857114>
7. Juice Jacking Vulnerability for iOS. <https://www.infotransec.com/news/juice-jacking-vulnerability-ios>
8. Lau, B., Jang, Y., Song, C.: Mactans: injecting malware into iOS devices via malicious chargers. Blackhat, USA (2013)
9. Li, L., Zhao, X., Xue, G.: Unobservable re-authentication for smartphones. In: Proceedings of the 20th Annual Network and Distributed System Security Symposium (NDSS), pp. 1–16 (2013)
10. Li, W., Meng, W.: An empirical study on email classification using supervised machine learning in real environments. In: Proceedings of the 2015 IEEE International Conference on Communications (ICC), pp. 7438–7443. IEEE, (2015)
11. Meng, Y., Wong, D.S., Schlegel, R., Kwok, L.: Touch gestures based biometric authentication scheme for touchscreen mobile phones. In: Kutylowski, M., Yung, M. (eds.) *Inscrypt 2012*. LNCS, vol. 7763, pp. 331–350. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38519-3_21

12. Meng, Y., Li, W., Kwok, L.-F.: Enhancing click-draw based graphical passwords using multi-touch on mobile phones. In: Janczewski, L.J., Wolfe, H.B., Sheno, S. (eds.) SEC 2013. IAICT, vol. 405, pp. 55–68. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39218-4_5
13. Meng, W., Li, W., Kwok, L.F.: EFM: enhancing the performance of signature-based network intrusion detection systems using enhanced filter mechanism. *Comput. Secur.* **43**, 189–204 (2014)
14. Meng, W., Wong, D.S., Furnell, S., Zhou, J.: Surveying the development of biometric user authentication on mobile phones. *IEEE Commun. Surv. Tutor.* **17**(3), 1–10 (2015)
15. Meng, W., Lee, W.H., Murali, S.R., Krishnan, S.P.T.: Charging me and I know your secrets! Towards juice filming attacks on smartphones. In: Proceedings of the Cyber-Physical System Security Workshop (CPSS), in Conjunction with AsiaCCS 2015. ACM (2015)
16. Meng, W., Lee, W.H., Murali, S.R., Krishnan, S.P.T.: JuiceCaster: towards automatic juice filming attacks on smartphones. *J. Netw. Comput. Appl.* **68**, 201–212 (2016)
17. Meng, W., Lee, W.H., Krishnan, S.P.T.: A framework for large-scale collection of information from smartphone users based on juice filming attacks. In: Proceedings of the Singapore Cyber Security R&D Conference (SG-CRC), pp. 99–106, January 2016
18. Meng, W., Fei, F., Li, W., Au, M.H.: Harvesting Smartphone Privacy through Enhanced Juice Filming Charging Attacks. In: Proceedings of the 20th Information Security Conference (ISC) (2017)
19. Meng, W., Jiang, L., Wang, Y., Li, J., Zhang, J., Xiang, Y.: JFCGuard: Detecting juice filming charging attack via processor usage analysis on smartphones. *Comput. Secur.* 13 p. (2018, in press). <https://doi.org/10.1016/j.cose.2017.11.012>
20. Ossmann, M., Osborn, K.: Multiplexed Wired Attack Surfaces. Black Hat USA (2013). <https://media.blackhat.com/us-13/US-13-Ossmann-Multiplexed-Wired-Attack-Surfaces-WP.pdf>
21. Raguram, R., White, A.M., Goswami, D., Monroe, F., Frahm, J.-M.: iSpy: automatic reconstruction of typed input from compromising reflections. In: Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS), pp. 527–536. ACM, New York (2011)
22. Sae-Bae, N., Memon, N., Isbister, K., Ahmed, K.: Multitouch gesture-based authentication. *IEEE Trans. Inf. Forensics Secur.* **9**(4), 568–582 (2014)
23. Spolaor, R., Abudah, L., Moonsamy, V., Conti, M., Poovendran, R.: No free charge theorem: a covert channel via USB charging cable on mobile devices. In: Gollmann, D., Miyaji, A., Kikuchi, H. (eds.) ACNS 2017. LNCS, vol. 10355, pp. 83–102. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-61204-1_5
24. Singapore Power to provide 200 free mobile phone charging stations for SG50 (2015). <http://www.straitstimes.com/singapore/singapore-power-to-provide-200-free-mobile-phone-charging-stations-for-sg50>
25. The Original USB Condom. <http://int3.cc/products/usbcondoms>
26. Xu, N., Zhang, F., Luo, Y., Jia, W., Xuan, D., Teng, J.: Stealthy video capturer: a new video-based spyware in 3G smartphones. In: Proceedings of the 2nd ACM Conference on Wireless Network Security (WiSec), pp. 69–78. ACM, New York (2009)
27. Zhang, Y., Xia, P., Luo, J., Ling, Z., Liu, B., Fu, X.: Fingerprint attack against touch-enabled devices. In: Proceedings of the 2nd ACM Workshop on Security and Privacy in Smartphones and Mobile Devices (SPSM), pp. 57–68. ACM, New York (2012)



Reading Network Packets as a Natural Language for Intrusion Detection

Mamoru Mimura^(✉) and Hidema Tanaka

National Defense Academy, 1-10-20 Hashirimizu, Yokosuka, Kanagawa, Japan
mim@nda.ac.jp

Abstract. Detecting unknown malicious traffic is a challenging task. There are many behavior-based detection methods which use the characteristic of drive-by-download attacks or C&C traffic. However, many previous methods specialize the attack techniques. Thus, the adaptability is restricted. Moreover, they need to decide the feature vectors every attack method. This paper proposes a generic detection method which does not depend on attack methods and does not need devising feature vectors. This method reads network packets as a natural language with Paragraph Vector an unsupervised algorithm, and learns the feature automatically to detect malicious traffic. This paper conducts timeline analysis and cross-dataset validation with the multiple datasets which contain captured traffic from Exploit Kit (EK). The best F-measure achieves 0.98 in the timeline analysis and 0.97 on the other dataset. Finally, the result shows that using Paragraph Vector is effective on unseen traffic in a linguistic approach.

Keywords: Drive by download · C&C · Neural network
Bag of Words · Word2vec · Paragraph Vector · Doc2vec
Support Vector Machine

1 Introduction

In recent years, Drive-by Download attack (DbD attack) and Spear Phishing attack are main attack techniques on the Internet. In general, intrusion detection techniques on a network are classified roughly into pattern-matching-based methods and methods using blacklists. The pattern-matching-based methods are effective, if the malicious traffic contains a unique string pattern. IDS uses fixed strings or regular expression to describe the signatures. However, recent Exploit Kits (EKs) (e.g. Angler EK, RIG EK) communicate via a standard protocol to imitate benign http traffic. EK is a software kit designed to run on web servers, with the purpose of identifying software vulnerabilities in client machines communicating with it. EK discovers and exploits vulnerabilities to upload and execute malicious code on the client via standard protocols. Some queries do not contain the EK specific strings. Therefore, it is difficult to describe the signatures. In this case, IDS can use the malicious destination server (e.g. Landing site, C&C

server) address as the signature. A firewall or a proxy server can also use the malicious destination server address as the blacklist. However, the attacker can change the malicious destination servers to evade detection by network devices. Some attackers use compromised hosts as stepping stones. Therefore, using the blacklist does not play a critical role. In addition, the malicious server address has to be already-known before the cyberattack. Thus, detecting unknown malicious traffic is a challenging task.

There are many behavior-based detection methods to detect unknown malicious traffic. These methods capture the characteristics of DbD attacks [22–24] or C&C traffic [9, 10], and detect unseen malicious traffic. Many previous methods, however, can detect only DbD attacks or C&C traffic. Because these methods usually use different detection techniques. If attackers modify the attack techniques, these previous methods barely detect unseen malicious traffic. Besides security researchers need to devise the feature vectors to capture the characteristics. Furthermore, some attackers still use protocols other than http or https. For instance, recent WannaCry ransomware uses Server Message Block (SMB) to compromise Windows machines, load malware, and propagate to other machines in a network. SMB is a transport protocol used by Windows computers for a wide variety of purposes such as file or printer sharing.

This paper focuses on the characteristic that Neural Network (NN) learns feature vector representation automatically. In this paper, we presume network packets are written in a natural language, and attempt to learn the difference of benign traffic and malicious traffic automatically with Paragraph Vector. Paragraph Vector is an unsupervised algorithm that learns fixed-length feature representations from variable-length pieces of texts. Then we input the extracted feature vectors with the label into supervised learning models to classify benign traffic and malicious traffic. The key idea of this research is reading network packets as a natural language.

This paper proposes a generic detection method, which does not depend on attack methods and does not need devising feature vectors. Our generic detection method does not rely on protocols and attack techniques, and does not demand devising feature vectors. This paper conducts timeline analysis with the dataset which contains captured traffic from Exploit Kit (EK) between 2014 and 2017. We used the most up-to-date captured traffic which was downloaded from the website Malware-Traffic-Analysis.net [1]. This paper also demonstrates cross-dataset validation by showing that an automated feature extraction scheme learned from one dataset can be used successfully for classification on another dataset. The best F-measure achieves 0.98 in the timeline analysis and 0.97 on the other dataset. Finally, the result shows that using Paragraph Vector is effective on unseen traffic in a linguistic approach.

The main contributions of this paper are four-fold: (1) Proposed a generic detection method which did not rely on protocols and attack techniques. (2) Verified that the proposed method could detect up-to-date EKs. (3) Verified that the proposed method was effective on another dataset. (4) Verified that using Paragraph Vector is effective on unseen traffic in a linguistic approach.

The rest of the paper is organized as follows. Next section briefly discusses related works and makes clear the difference among our method and previous methods. Section 3 describes Natural Language Processing (NLP) techniques which include Paragraph Vector. Section 4 proposes a generic detection method based on the NLP techniques. Section 5 shows experimental results applying the proposed method to the multiple datasets. Section 6 discusses the results, and reveals the performance and effectiveness.

2 Related Work

In general, the main studies of network intrusion detection include signature-based detection and behavior-based detection. Signature-based detection relies on an existing signature database to detect known malicious traffic, and barely detects unknown malicious traffic. Therefore, many behavior-based detection methods are proposed. For example, some methods focused on the traffic classification from packet traces [4–8]. Analyzing packets is, however, becoming intractable on broadband networks. The alternative approach is classification based on network logs such as DNS records, NetFlow or proxy server logs. There are several methods which use NetFlow [9, 10], DNS records [11–14] and proxy server logs [15–22]. However, recent Exploit Kits (EKs) (e.g. Angler EK, RIG EK) communicate via a standard protocol to imitate normal http communication. Furthermore, some attackers use compromised hosts as stepping stones. Thus, detecting recent EKs from logs is becoming a challenging task. Therefore, many methods use additional contents to distinguish malicious traffic from seemingly benign traffic. For instance, some methods analyze JavaScript code to detect these EKs [23, 24].

These previous methods utilize the characteristic of DbD attacks or C&C traffic well. However, their good contrivance can backfire. Many previous methods specialize the attack techniques, and the adaptability is limited. In addition, many previous methods using machine learning technique demand devising feature vectors to distinguish malicious traffic. In other words, the essence of the previous works was how to extract feature vectors from network traffic, logs and contents. Our method is fundamentally different and based on the other viewpoint. Our method is a generic intrusion detection method which can detect many attack techniques with a simple technique. Our method does not demand devising feature vectors. Because our method learns the difference of benign traffic and malicious traffic automatically with NN.

3 Natural Language Processing (NLP) Technique

3.1 Word2vec

To calculate various measures to characterize a text, we have to transform the text into a vector. Word2vec [2] is a model that produces word embedding.

Word embedding is the collective name for a set of language modeling and feature learning techniques in NLP where words from the vocabulary are mapped to vectors of real numbers. This model is a shallow, two-layer neural network that is trained to reconstruct linguistic contexts of words. This model takes as its input a large corpus of text and produces a vector space, with each unique word in the corpus being assigned a corresponding vector in the space. Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located in close proximity to each other in the space. Word2vec is based on the distributional hypothesis, which motivates that the meaning of a word can be gauged by its context. Thus, if two words occur in the same position in two sentences, they are very much related either in semantics or syntactic. Word2vec utilizes two algorithms to produce a distributed representation of words. One is Continuous-Bag-of-Words (CBoW), and the other is skip-gram. In the CBoW algorithm, the model predicts the current word from a window of surrounding context words. In the skip-gram algorithm, the model uses the current word to predict the surrounding window of context words. Word2vec enables to calculate the semantic similarity between two words and infer similar words semantically. However, Word2vec is a model that merely produces word embedding. To calculate the semantic similarity between two documents, this model has to be extended.

3.2 Paragraph Vector (Doc2vec)

An extension of Word2vec to construct embedding from entire documents has been proposed [3]. This extension is called Doc2vec or Paragraph2vec, and has been implemented. Doc2vec is based on the same distributional hypothesis, which motivates that the meaning of a sentence can be gauged by its context. Thus, if two sentences occur in the same position in two paragraphs, they are very much related either in semantics or syntactic in the same way. Doc2vec utilizes two algorithms to produce Paragraph Vector a distributed representation of entire documents. One is Distributed-Memory (DM), and the other is Distributed-Bag-of-Words (DBoW). DM is an extension of CBoW, and the only change in this model is adding a document ID as a window of surrounding context words. DBoW is an extension of skip-gram, and the current word was replaced by the current document ID. Doc2vec enables to calculate the semantic similarity between two documents and infer similar documents semantically. Some implementations support also inference of document embedding on unseen documents. This function is important to develop a practical system to detect unseen malicious traffic. Because, unseen malicious traffic might include an unknown word (e.g. newly-changed FQDN, random strings).

4 Proposed Method

4.1 Translating Network Packets into a Language

The key idea of our method is reading network packets as a natural language. In order to apply NLP techniques, network packets have to be translated and

```

192.168.204.175 → 206.188.192.114 TCP 66 49380 → 80 [SYN] Seq=0 Win=8192
Len=0 MSS=1460 WS=4 SACK_PERM=1
206.188.192.114 → 192.168.204.175 TCP 60 80 → 49380 [SYN, ACK] Seq=0 Ack=1
Win=64240 Len=0 MSS=1460
192.168.204.175 → 206.188.192.114 TCP 60 49380 → 80 [ACK] Seq=1 Ack=1
Win=64240 Len=0

```

Fig. 1. Summary lines in a transport layer.

```

10.180.0.14 → 10.180.0.254 DNS 82 Standard query 0xe854 A www.antiqueceramics.tk
10.180.0.14 → 10.180.0.254 DNS 82 Standard query 0xe854 A www.antiqueceramics.tk
10.180.0.254 → 10.180.0.14 DNS 371 Standard query response 0xe854 A
www.antiqueceramics.tk CNAME antiqueceramics.tk A 195.20.34.1 A 195.20.34.2 NS
d.ns.tk NS c.ns.tk NS b.ns.tk NS a.ns.tk A 194.0.41.1 AAAA 2001:678:5c::1 A 194.0.39.1
AAAA 2001:678:54::1 A 194.0.40.1 AAAA 2001:678:58::1 A 194.0.38.1 AAAA
2001:678:50::1
10.180.0.14 → 195.20.34.1 HTTP 316 GET / HTTP/1.0
195.20.34.1 → 10.180.0.14 HTTP 60 HTTP/1.0 203 Non-Authoritative Information
(text/html)

```

Fig. 2. Summary lines in an application layer.

separated into words. However, reading network packets is becoming intractable on broadband networks. Therefore, we need a lightweight translation method. To translate network packets into a language, we use TShark [25] a network protocol analyzer, which captures and decodes packet data from a network. TShark displays a summary line for each received packet. The summary line consists of some fields such as source and destination IP address, protocol, size and basic contents. Our method uses these fields as separated words.

Figure 1 shows summary lines in a transport layer. In a transport layer, our method uses protocol, port number, size and flags. Our method ignores other parameters, because there are a very wide range of the types.

Figure 2 shows summary lines in an application layer. In an application layer, our method uses all fields after protocol. Furthermore, our method separates FQDN (Fully Qualified Domain Name) by “dot” (.). Then we can derive top level domain name, sub domain name and so on, which means the country, organization, use or the purpose (e.g. www, mail). Our method separates path by “slash” (/) and “dot” (.), “question mark” (?), “equal” (=) and “and” (&). Then, we can derive the directory name, file name, extension from the path. We can also derive the variable names and values from the query string, which are used in the running program on the server.

4.2 Proposed Method

Figure 3 shows an overview of our method. First, our method constructs a corpus from known malicious traffic and benign traffic. Each traffic is translated and separated by the previously mentioned method. In this paper, we use 2 reading methods. Table 1 shows a summary of the reading methods. Each method reads only each layer, and collects 100 summary lines for a paragraph.

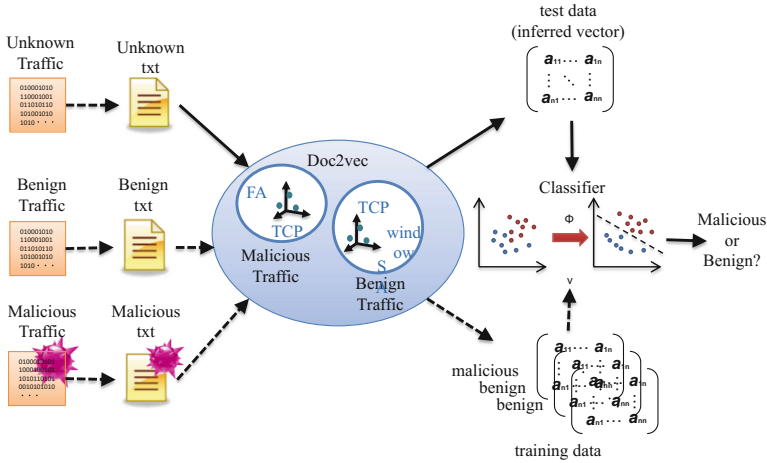


Fig. 3. An overview of the proposed method.

Table 1. A summary of the reading methods

Method	Layer	Contents
Method 1	Transport layer	Protocol, port number, size
Method 2	Application layer	FQDN, path, user agent

Then, the Doc2vec constructs a vector space from the corpus, and converts each paragraph into vectors with the labels. These labeled vectors are training data for a classifier. In this time, we use Support Vector Machine (SVM) for the classifier. A SVM model is a representation of the training data as points in space, mapped so that the training data of the separate categories are divided by a clear gap that is as wide as possible. Test data are mapped into that same space and predicted to belong to a category based on which side of the gap they fall. Given a set of training data, each labeled as belonging to one or the other of two categories, this training algorithm builds a model that assigns new examples to one category or the other.

After that, we convert unknown traffic into vectors. These unlabeled vectors are test data for the classifier. Finally, we input these unlabeled vectors to the trained classifier, and can obtain a predicted label. The predicted label is either malicious or benign.

4.3 Implementation

The proposed method was developed by Python-2.7 with open source machine learning libraries, gensim-1.01 [26] and scikit-learn-0.18.0 [27].

Gensim is a Python library to realize unsupervised semantic modelling from plain text, and includes a Doc2vec model. Table 2 shows the parameters for the

Table 2. The parameters for the Doc2vec model

Dimensionality of the feature vectors	100
Window	15
Number of epochs	30
Training algorithm	DBoW

Doc2vec model. We set the dimensionality of the feature vectors 100, and chose DBoW which was an extension of skip-gram. The window is the maximum distance between the predicted word and context words used for prediction within a document. Scikit-learn is a machine-learning library for Python that provides tools for data mining with a focus on machine learning. The proposed method uses a SVC function with a linear kernel for SVM.

5 Experiment

5.1 Dataset

To reveal the effectiveness to up-to-date EKs, we use captured pcap files from EKs between 2014 and 2017 (MTA dataset), which were downloaded from the website Malware-Traffic-Analysis.net [1]. We also use D3M (Drive-by Download Data by Marionette) dataset and NCD (Normal Communication Data in MWS-Cup 2014) for the cross-dataset validation. These datasets are parts of MWS datasets [28], and include pcap files. MTA and D3M contain malicious traffic and NCD contains benign traffic. Table 3 shows the detail.

The MTA dataset contains traffic from the following EKs, Angler EK, Fiesta-EK, FlashPack-EK, Magnitude EK, Neutrino EK, Nuclear EK and RIG EK. D3M is a set of packet traces collected from the web-client, high-interaction honeypot system, which is based on Internet Explorer on Windows OS with several vulnerable plugins, such as Adobe Reader, Flash Player, Java and so on. This dataset contains traffic from EKs (e.g. Blackhole EK, Elenore, Mpack).

Table 3. The detail of the datasets.

MTA		D3M	
Year	Size	Year	Size
2014	238M	2010	130M
2015	186M	2011	24.8M
2016	373M	2012	33.2M
2017	109M	2013	14.6M
-	-	2014	23.3M
-	-	2015	334M

We extracted summary lines from these pcap files with TShark. After that, we compounded the malicious summary lines and the benign summary lines into a dataset at the same rate. We split the dataset into training data and test data to conduct timeline analysis and cross-dataset validation. The proposed method uses only training data to construct a corpus. Because, our method presumes that the test data is completely unknown traffic in practical condition. To compare the characteristics of our method, we also use a Bag-of-Words (BoW) model. BoW is a simplifying representation used in natural language processing. The most common type of features calculated from BoW is the number of times a term appears in the sentence. In the timeline analysis, we chose an annual traffic as training data, and the subsequent traffic is the test data.

5.2 Result

In this experiment, we use 3 metrics: Precision (P), Recall (R) and F-measure (F). Table 4 shows the results of the timeline analysis.

Contrary to expectations, BoW was generally more effective than Doc2vec with the method 1. The both F-measures maintain a generally constant values over three years, and the best one has reached 0.96. This result means that word frequency is the most distinctive feature in a transport layer.

Doc2vec was generally more effective than BoW with the method 2. The best F-measure has reached 0.98 in the next year, and the both F-measures gradually decrease. This result means that Doc2vec capture distinctive features other than word frequency in an application layer.

Table 4. The result of the timeline analysis.

Method	Training data	Test data	Model	NCD (Benign)			MTA (Malicious)		
				P	R	F	P	R	F
1	2014	2015	BoW	0.97	0.93	0.95	0.93	0.97	0.95
			Doc2vec	0.94	0.80	0.86	0.93	0.95	0.88
		2016	BoW	1.00	0.92	0.96	0.92	1.00	0.96
			Doc2vec	0.97	0.78	0.87	0.82	0.97	0.89
		2017	BoW	0.99	0.92	0.95	0.92	0.99	0.96
			Doc2vec	0.99	0.80	0.89	0.83	1.00	0.91
2	2014	2015	BoW	0.95	0.80	0.87	0.83	0.96	0.89
			Doc2vec	0.99	0.98	0.98	0.98	0.99	0.98
		2016	BoW	0.88	0.83	0.85	0.84	0.89	0.86
			Doc2vec	0.88	0.97	0.92	0.96	0.87	0.92
		2017	BoW	0.84	0.81	0.83	0.83	0.85	0.84
			Doc2vec	0.77	0.98	0.86	0.96	0.69	0.80

Table 5 shows the results of the cross-dataset validation.

Table 5. The result of the cross-dataset validation.

Method	Training data	Test data	Model	NCD (Benign)			MTA (Malicious)		
				P	R	F	P	R	F
1	MTA	D3M	BoW	0.57	0.96	0.71	0.90	0.36	0.52
			Doc2vec	0.65	0.89	0.75	0.86	0.59	0.70
	D3M	MTA	BoW	0.40	0.96	0.56	0.93	0.25	0.40
			Doc2vec	0.59	0.93	0.72	0.95	0.66	0.78
2	MTA	D3M	BoW	0.89	0.88	0.88	0.87	0.89	0.88
			Doc2vec	0.98	0.96	0.97	0.96	0.98	0.97
	D3M	MTA	BoW	0.74	0.98	0.84	0.97	0.67	0.79
			Doc2vec	0.79	0.99	0.88	0.99	0.74	0.85

BoW was not effective at all with the method 1. Furthermore, even Doc2vec was not effective enough. Therefore, BoW is not effective in the other environment at all. Besides, it is difficult to detect unseen malicious traffic from transport layer information in the other environment.

Doc2vec was generally more effective than BoW with the method 2. In the case that we used MTA for training data, the best F-measure has reached 0.97. This result means that MTA is superior to D3M as a training data. Moreover, using Doc2vec is effective to detect unseen malicious traffic from application layer information in the other environment.

6 Discussion

6.1 Accuracy

As the results of the experiments, using Doc2vec in an application layer was the most effective. In the timeline analysis, the best F-measure achieves 0.98 in the next year. The F-measure achieves 0.80 even in 3 years. In the cross-dataset validation, the best F-measure achieves 0.97. Thus, our method is precise, if we can obtain good training data.

6.2 Mechanism

In a transport layer, word frequency was the most distinctive feature. The most frequent words were traffic sizes. In a sense, this is rote memorization. In fact, BoW was not effective in the other environment at all. Furthermore, even Doc2vec was not effective enough. Therefore, we concluded that it was difficult to detect unseen malicious traffic from transport layer information.

In an application layer, Doc2vec captured distinctive features other than word frequency. In fact, Doc2vec was also effective in the other environment. Needless to say, word frequency is a fundamental element in a linguistic approach. However, unique word count in network traffic is unrestricted. Representing unrestricted traffic with only word frequency has serious limitations. Doc2vec

is based on the distributional hypothesis that words occurring in similar context tend to have similar meanings. Doc2vec enables to calculate the semantic similarity between two traffic and infer similar traffic semantically. We believe network traffic in an application layer has the context, and is like a natural language. Hence, we concluded that our method could detect unseen malicious traffic.

6.3 Adaptability

Our method can detect a variety of malicious traffic in the same method. All we have to do is input malicious and benign pcap files. Our method can detect malicious traffic regardless of the attack techniques. No prior knowledge of the attack techniques is required. Our method is available in every protocols which TShark can analyze in an application layer. If attackers change the attack techniques or protocols, our methods learn the characteristic automatically. Besides our method does not demand devising feature vectors. Hence, our method is adaptable to many attack techniques.

6.4 Durability

Our method learns the difference of benign traffic and malicious traffic automatically with neural networks. In neural networks, it is difficult to specify what feature of an input data a specific feature map captures. This means that an attacker cannot recognize the features either. Therefore, an attacker has no effective countermeasure to evade this method. The only option is imitating normal communication completely. Thus, our method is effective and durable in the long term.

6.5 Practical Use

The proposed method was effective in the other dataset. This means the proposed method is powerful and versatile. In this paper, we used malicious and benign pcap files. We can obtain these malicious pcap files easily from the websites, which disclose malicious traffic data. We can also obtain benign pcap files easily from everywhere. Thus, the proposed method has a few constraints in practical use.

7 Conclusion

In this paper, we proposed how to construct a corpus from network packets and a generic detection method, which does not depend on attack methods and does not demand devising feature vectors. This paper conducted timeline analysis with MTA dataset which contains captured traffic from EKs between 2014 and 2017. This paper also demonstrated cross-dataset validation which uses MTA

dataset and D3M dataset. Consequently, the proposed method can detect up-to-date traffic from EKs. We verified that the proposed method was effective over three years, and effective on the other dataset too. The proposed method achieved the F-measure of 0.98 in the timeline analysis and the F-measure of 0.97 on the other dataset. This result means that using Paragraph Vector is effective on unseen traffic in a linguistic approach.

In this paper, we presumed network packets were written in a natural language. We can presume any other logs such as IDS alerts, firewall logs, SIEM (Security Information and Event Management) events are written in a natural language in the same manner. We believe this would enable to classify the detail automatically.

Acknowledgment. This work was supported by JSPS KAKENHI Grant Number 17K06455.

References

1. Malware-Traffic-Analysis.net. <http://www.malware-traffic-analysis.net/>
2. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: *Advances in Neural Information Processing Systems*, pp. 3111–3119 (2013)
3. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: *Proceedings of 31st International Conference on Machine Learning*, pp. 1188–1196 (2014)
4. Wang, K., Stolfo, S.J.: Anomalous payload-based network intrusion detection. In: Jonsson, E., Valdes, A., Almgren, M. (eds.) *RAID 2004*. LNCS, vol. 3224, pp. 203–222. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30143-1_11
5. Moore, D., Shannon, C., Brown, D.J., Voelker, G.M., Savage, S.: Inferring internet denial-of-service activity. *ACM Trans. Comput. Syst.* **24**(2), 115–139 (2006)
6. Bailey, M., Oberheide, J., Andersen, J., Mao, Z.M., Jahanian, F., Nazario, J.: Automated classification and analysis of internet malware. In: Kruegel, C., Lippmann, R., Clark, A. (eds.) *RAID 2007*. LNCS, vol. 4637, pp. 178–197. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74320-0_10
7. Song, H., Turner, J.: Toward advocacy-free evaluation of packet classification algorithms. *IEEE Trans. Comput.* **60**(5), 723–733 (2011)
8. Karagiannis, T., Papagiannaki, K., Faloutsos, M.: BLINC: multilevel traffic classification in the dark. In: *Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 229–240 (2005)
9. Gu, G., Perdisci, R., Zhang, J., Lee, W.: BotMiner: clustering analysis of network traffic for protocol and structure independent botnet detection. In: *Proceedings of USENIX Security Symposium*, vol. 5, pp. 139–154 (2008)
10. Bilge, L., Balzarotti, D., Robertson, W., Kirda, E., Kruegel, C.: Disclosure: detecting botnet command and control servers through large-scale netflow analysis. In: *Proceedings of the 28th Annual Computer Security Applications Conference*, pp. 129–138 (2012)
11. Antonakakis, M., Perdisci, R., Dagon, D., Lee, W., Feamster, N.: Building a dynamic reputation system for DNS. In: *Proceedings of the 19th USENIX Security Symposium* (2010)

12. Antonakakis, M., Perdisci, R., Lee, W., Vasiloglou II, N., Dagon, D.: Detecting malware domains at the upper DNS hierarchy. In: Proceedings of 20th USENIX Security Symposium (2011)
13. Antonakakis, M., Perdisci, R., Nadji, Y., Vasiloglou, N., Abu-Nimeh, S., Lee, W., Dagon, D.: From throw-away traffic to bots: detecting the rise of DGA-based malware. In: Proceedings of 21th USENIX Security Symposium (2012)
14. Rahbarinia, B., Perdisci, R., Antonakakis, M.: Segugio: efficient behavior-based tracking of new malware-control domains in large ISP networks. In: Proceedings of the 2015 IEEE/IFIP International Conference on Dependable Systems and Networks (2015)
15. Kruegel, C., Vigna, G.: Anomaly detection of web-based attacks. In: Proceedings of the 10th ACM Conference on Computer and Communications Security, pp. 251–261 (2003)
16. Choi, H., Zhu, B.B., Lee, H.: Detecting malicious web links and identifying their attack types. In: Proceedings of the 2nd USENIX Conference on Web Application Development, pp. 1–11 (2011)
17. Ma, J., Saul, L.K., Savage, S., Voelker, G.M.: Learning to detect malicious URLs. In: ACM Transactions on Intelligent Systems and Technology, vol. 23, Article no. 30 (2011)
18. Zhao, P., Hoi, S.C.: Cost-sensitive online active learning with application to malicious URL detection. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 919–927 (2013)
19. Invernizzi, L., Miskovic, S., Torres, R., Saha, S., Lee, S., Mellia, M., Kruegel, C., Vigna, G.: Nazca: detecting malware distribution in large-scale networks. In: Proceedings of the Network and Distributed System Security Symposium (2014)
20. Nelms, T., Perdisci, R., Antonakakis, M., Ahamad, M.: Webwitness: investigating, categorizing, and mitigating malware download paths. In: Proceedings of the 24th USENIX Security Symposium, pp. 1025–1040 (2015)
21. Bartos, K., Sofka, M.: Optimized invariant representation of network traffic for detecting unseen malware variants. In: Proceedings of the 25th USENIX Security Symposium, pp. 806–822 (2016)
22. Shibahara, T., Yamanishi, K., Takata, Y., Chiba, D., Akiyama, M., Yagi, T., Ohsita, Y., Murata, M.: Malicious URL sequence detection using event de-noising convolutional neural network. In: Proceedings of the IEEE ICC 2017 Communication and Information Systems Security Symposium (2017)
23. Takata, Y., Akiyama, M., Yagi, Y., Hariu, T., Goto, G.: MineSpider: extracting URLs from environment-dependent drive-by download attack. In: Proceedings of the 2015 IEEE 39th Annual Computer Software and Applications Conference, vol. 2, pp. 444–449 (2015)
24. Jodavi, M., Abadi, M., Parhizkar, E.: DbDHunter: an ensemble-based anomaly detection approach to detect drive-by download attacks. In: Proceedings of the 2015 5th International Conference on Computer and Knowledge Engineering (ICCKE), pp. 273–278 (2015)
25. tshark - Dump and analyze network traffic. <https://www.wireshark.org/docs/man-pages/tshark.html>
26. gensim. <https://radimrehurek.com/gensim/>
27. scikit-learn. <http://scikit-learn.org/>
28. Hatada, M., Akiyama, M., Matsuki, T., Kasama, T.: Empowering anti-malware research in Japan by sharing the MWS datasets. *J. Inf. Process.* **23**(5), 579–588 (2015)



Friend-Safe Adversarial Examples in an Evasion Attack on a Deep Neural Network

Hyun Kwon¹, Hyunsoo Yoon¹, and Daeseon Choi²(✉)

¹ Korea Advanced Institute of Science and Technology (KAIST),
Daejeon, South Korea

{khkh, hyoon}@kaist.ac.kr

² Kongju National University, Gongju-si, South Korea
sunchoi@kongju.ac.kr

Abstract. Deep neural networks (DNNs) perform effectively in machine learning tasks such as image recognition, intrusion detection, and pattern analysis. Recently proposed adversarial examples—slightly modified data that lead to incorrect classification—are a severe threat to the security of DNNs. However, in some situations, adversarial examples might be useful, i.e., for deceiving an enemy classifier on a battlefield. In that case, friendly classifiers should not be deceived. In this paper, we propose adversarial examples that are *friend-safe*, which means that friendly machines can classify the adversarial example correctly. To make such examples, the transformation is carried out to minimize the friend’s wrong classification and the adversary’s correct classification. We suggest two configurations of the scheme: targeted and untargeted class attacks. In experiments using the MNIST dataset, the proposed method shows a 100% attack success rate and 100% friendly accuracy with little distortion (2.18 and 1.53 for each configuration, respectively). Finally, we propose a mixed battlefield application and a new covert channel scheme.

Keywords: Adversarial example · Covert channel · Deep neural network
Evasion attack

1 Introduction

Deep neural networks [1] are widely used in image recognition [2], speech recognition [3], intrusion tolerance systems [4], natural language processing [5], and games [6]; therefore, the security and safety of neural networks has received considerable attention in security research community. Szegedy et al. [7] presented *adversarial examples* in image classification; in an evasion attack, images that are transformed slightly can be incorrectly classified by a machine learning classifier, even when the changes are so small that a human cannot recognize them easily. An attacker can cause a self-driving car to take unwanted action by making slight changes to road signs [8]. Counter measures to these attacks have been proposed [20–22], and subsequently, more advanced attacks were developed to defeat the counter measures.

This evasion attack can be utilized in military domains, with the adversarial example used to deceive an enemy’s machine classifier. For example, battlefield road signs could be modified to deceive an adversary’s self-driving vehicle. If the battlefield is shared by enemy and friendly forces, friendly self-driving vehicles should not be deceived by the attack.

In this paper, we propose an evasion attack scheme that creates adversarial examples that are incorrectly classified by enemy classifiers and correctly recognized by friendly classifiers. The proposed scheme has two configurations: targeted and untargeted classes. In the targeted scheme, a transformer changes the original sample to be recognized as a specific target class. In the untargeted scheme, the goal of transformation is incorrect classification to any class other than the right class.

We evaluate our scheme on a standard dataset: MNIST [9], a digit-recognition task (0–9). We use a defensive distillation classifier [10], which is a state-of-the-art anti-evasion classifier as the enemy classifier distortion rates similar to those used in state-of-the-art evasion attack schemes [11].

This paper makes the following contributions:

- We propose a scheme for generating adversarial examples that are *friend-safe*, i.e., that can be recognized by a friendly classifier. Our scheme is the first published attempt that handles this problem. We propose two configurations: targeted and untargeted class attacks.
- We apply our scheme to an anti-evasion classifier [10] and deceive it with 100% success, while maintaining a 100% accuracy rate from our friendly classifier without any modification or retraining. We discover that it is possible to achieve both objectives simultaneously while maintaining low distortion.
- We analyze the difference in distortion between the targeted and untargeted scheme, and the difference among targeted digits. This analysis is useful for attack planning because distortion is related to the possibility that a person will detect an attack.
- We propose a new covert channel [12] scheme as another application, in which the roles of the friend and adversary are reversed. The target class of an adversarial example is the hidden information transferred via the covert channel.

The remainder of this paper is structured as follows: in the next section, related work on attacks on machine learning is introduced. In Sect. 3, the proposed friend-safe adversarial example generation is introduced. Experimental results of the proposed scheme and findings are presented in Sect. 4. Discussion of the proposed scheme is presented in Sect. 5. Section 6 concludes the paper.

2 Related Work

Barreno et al. [13] discussed several security issues of machine learning. They categorized attacks on machine learning into causative attacks, which influence learning with control over training data, and exploratory attacks that exploit misclassifications but do not affect training.

Poisoning attacks, a type of causative attack that adds malicious training data have been proposed [14–16]. Although poisoning attacks are effective, they require that the

attacker access training data while it is being used to train a victim model. This assumption is unrealistic, so poisoning attacks are not considered a severe threat to machine learning applications.

Szegedy et al. [7] first presented the adversarial example, a kind of exploratory attack. In this scheme, the attacker transforms an image slightly, causing this adversarial example to be misclassified. The success rate of adversarial examples in [17] on standard image sets such as ImageNet was greater than 97%. The amount of modification needed was so small (about 4.02%) that humans could not detect the difference. This attack was untargeted, meaning that it succeeds if an example is classified to any class other than the correct one. Moosavi-Dezfooli et al. [18] proposed targeted attacks, in which there is a misclassification class goal: the attacker transforms data to be classified as a specific target class. In these attacks, it is assumed that attacker can acquire victim's classification result for any input data. Even in military scenarios, it is not unrealistic to acquire an enemy's weapon.

Counter measures to these attacks have been presented. Biggio et al. [19] proposed a binary classifier for detecting adversarial examples. This work covered conventional machine learning models, such as support vector machines [20] and logistic regression [21], not deep neural networks. Goodfellow et al. [22] proposed a new neural network activation function that is robust to adversarial examples. Applying this method necessitates changing the neural network's architecture.

Recently, Papernot et al. [10] proposed a defensive distillation scheme, in which an initial and a distilled network are used: the class probability of the initial network's output is used as label for training the distilled network. This method prevents overfitting the distilled network, making it more robust to adversarial examples. In experiments, their scheme reduced the success rate of evasion attacks to 0.45% from 95.89%.

One year later, Carlini et al. [11] showed that they could deceive a distilled network with 100% success, and that their scheme could be applied to both targeted and untargeted attacks.

In this manner, advanced attacks and their counter measures have been being proposed continuously. However, there has been no published scheme for building adversarial examples that do not affect friendly classifiers. In this paper, we use these state-of-the-art technologies in our friend-safe adversarial example scheme and its evaluation.

3 Proposed Method

To generate a friend-safe adversarial example, we propose a network architecture that consists of a transformer, a friendly discriminator D_{friend} , and an enemy discriminator D_{enemy} , as shown in Fig. 1. The transformer takes original sample $x \in X$ and original class $y \in Y$ as input and converts the original sample to transformed example x^* .

D_{friend} and D_{enemy} are pretrained classifiers and not changed during transformation. They take x^* as their input and provide their classification result (loss) to the transformer.

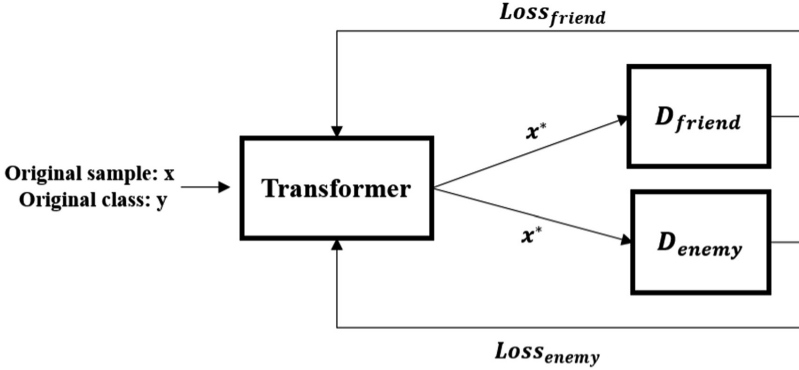


Fig. 1. Proposed architecture

The goal of this architecture is that transformed example x^* is incorrectly classified by D_{enemy} and correctly classified by D_{friend} , minimizing the distortion from the original sample. There are two configurations in which transformed example x^* is incorrectly classified by D_{enemy} : targeted and untargeted adversarial examples. In mathematical expressions, the operation functions of D_{enemy} and D_{friend} are denoted as $f_{enemy}(x)$ and $f_{friend}(x)$.

Given the pretrained D_{friend} , D_{enemy} , and original input $x \in X$, the problem is an optimization problem that generates targeted adversarial example x^* :

$$x^* : \operatorname{argmin}_{x^*} L(x, x^*) \text{ s.t. } f_{friend}(x^*) = y \text{ and } f_{enemy}(x^*) = y^*,$$

where $L(\cdot)$ is a chosen distance measure between original sample x and transformed example x^* , and $y^* \in Y$ is the target class chosen by the attacker. An untargeted adversarial example x^* is generated similarly:

$$x^* : \operatorname{argmin}_{x^*} L(x, x^*) \text{ s.t. } f_{friend}(x^*) = y \text{ and } f_{enemy}(x^*) \neq y.$$

To achieve this goal, the procedure consists of pretraining D_{friend} and D_{enemy} and a transformation that generates friend-safe adversarial example x^* .

First, D_{friend} and D_{enemy} are trained with the original MNIST sample to classify original sample x :

$$f_{enemy}(x) = x \in X \rightarrow y \in Y \text{ and } f_{friend}(x) = x \in X \rightarrow y \in Y.$$

In our experiments, D_{friend} and D_{enemy} were trained to classify the original sample with greater than 99% accuracy.

Second, the transformer accepts the original MNIST sample and original class as input and produces transformed example x^* . For this study, we modified the transformer architecture in [11], and x^* is defined as

$$x^* = \frac{\tanh(x + w)}{2},$$

where w is a modifier that is optimized by using \tanh to smooth the gradient. The classification loss of x^* by D_{friend} and D_{enemy} is returned to the transformer.

The transformer then calculates the total loss $loss_T$ and generates a friend-safe adversarial example by minimizing $loss_T$ iteratively. $loss_T$ is defined as:

$$loss_T = loss_{distortion} + loss_{friend} + loss_{enemy},$$

where $loss_{distortion}$ is the distortion of transformed example, and $loss_{friend}$ and $loss_{enemy}$ are the classification loss of D_{friend} and D_{enemy} . $loss_{distortion}$ is the distance between the original sample x and the transformed example x^* :

$$loss_{distortion} = \left\| x^* - \frac{\tanh(x)}{2} \right\|_2^2.$$

To satisfy $f_{friend}(x^*) = y$, $loss_{friend}$ should be minimized:

$$loss_{friend} = g^f(x^*),$$

where $g^f(k) = \max\{Z(k)_i; i \neq org\} - Z(k)_{org}$ and org is the original class. $Z(\cdot)$ [11, 17] is the probability of the class that is predicted by two discriminators, D_{friend} and D_{enemy} . $f_{friend}(x^*)$ has a higher probability of predicting the original class than other classes by optimally minimizing $loss_{friend}$. $loss_{enemy}$ has two cases, used in targeted and untargeted adversarial examples. To satisfy $f_{enemy}(x^*) = y^*$, $y^* \in Y$ in targeted adversarial examples, $loss_{enemy}$ is defined as:

$$loss_{enemy} = g^{e_t}(x^*),$$

where $g^{e_t}(k) = \max\{Z(k)_i; i \neq t\} - Z(k)_t$ and t is the targeted class. $f_{enemy}(x^*)$ has a higher probability of predicting targeted class y^* than other classes by optimally minimizing $loss_{enemy}$. To satisfy $f_{enemy}(x^*) \neq y$ in an untargeted adversarial example,

$$loss_{enemy} = g^{e_u}(x^*),$$

where $g^{e_u} = Z(k)_{org} - \max\{Z(k)_i; i \neq org\}$ and org is the original class. $f_{enemy}(x^*)$ has a lower probability of predicting the original class than other classes by optimally minimizing $loss_{enemy}$. The detailed procedure for generating a friend-safe adversarial example is described in Algorithm 1.

Algorithm 1. Friend-safe adversarial example generation in a transformer

Input: original sample x , original class y , targeted class y^* , number of iterations n ,**Targeted adversarial example generation:**

1. $w \leftarrow 0$
2. $org \leftarrow y$
3. $t \leftarrow y^*$
4. $x^* \leftarrow x$
5. **For** n step **do**
6. $x^* \leftarrow \frac{\tanh(x^*+w)}{2}$
7. $g^f(x^*) \leftarrow \max\{Z(x^*)_i : i \neq org\} - Z(x^*)_{org}$
8. $g^{e_t}(x^*) \leftarrow \max\{Z(x^*)_i : i \neq t\} - Z(x^*)_t$
9. Update w to minimize the gradient of $\|x^* - \frac{\tanh(x)}{2}\|_2^2 + g^f(x^*) + g^{e_t}(x^*)$
10. **End for**
11. return x^*

Untargeted adversarial example generation:

1. $w \leftarrow 0$
 2. $org \leftarrow y$
 3. $x^* \leftarrow x$
 4. **For** n step **do**
 5. $x^* \leftarrow \frac{\tanh(x^*+w)}{2}$
 6. $g^f(x^*) \leftarrow \max\{Z(x^*)_i : i \neq org\} - Z(x^*)_{org}$
 7. $g^{e_u}(x^*) \leftarrow Z(x^*)_{org} - \max\{Z(x^*)_i : i \neq org\}$
 8. Update w to minimize the gradient of $\|x^* - \frac{\tanh(x)}{2}\|_2^2 + g^f(x^*) + g^{e_u}(x^*)$
 9. **End for**
 10. return x^*
-

4 Experiment and Evaluation

Through experiments, we show that the proposed scheme can generate a friend-safe adversarial example that is incorrectly classified by an enemy classifier and correctly classified by a friendly classifier while minimally distorting the original sample. We used the Tensorflow [23] library, a widely used open source library for machine learning on a Xeon E5-2609 1.7 GHz server.

4.1 Experimental Method

MNIST [9], a collection of handwritten digit images (0–9), was used as dataset in the experiment. The experimental method consisted of (1) pretraining D_{friend} and D_{enemy} , and (2) transforming the friend-safe adversarial example.

First, in pretraining, D_{friend} and D_{enemy} are common convolution neural networks (CNNs) [24]. Their configuration and training parameters are shown in Tables 6 and 7 of the appendix. D_{enemy} is a distilled model [10] in which the classifier’s output class probability is used as input to a second phase of classifier training. To train D_{friend} and D_{enemy} , 60,000 training and another 10,000 test samples were used. In tests, D_{friend} and D_{enemy} correctly classified the original MNIST samples with 99.25% and 99.12% accuracy, respectively.

Second, to generate the friend-safe adversarial sample, Adam [25] was used as an optimizer to minimize the total loss with a learning rate of $1e^{-2}$ and an initial constant equal to $1e^{-3}$. For a given number of iterations, the transformer updates output x^* and gives it to D_{friend} and D_{enemy} , from which it then received feedback. At the end of the iterations, transformation result x^* was evaluated based on the accuracy of D_{friend} , the attack success rate, and the amount of distortion. The accuracy of D_{friend} is the coincidence rate between the original class and the output class of D_{friend} . The attack success rate is the intended success rate that D_{enemy} incorrectly classifies x^* . The attack success rate has two configuration: the targeted attack success rate and untargeted attack success rate. The targeted attack success rate is the coincidence rate between targeted class and the class output by D_{enemy} . The untargeted attack success rate is the rate of inconsistency between the original class and the output class of D_{enemy} . Distortion is measured as the pixel distance from the original sample, such as the mean square error.

4.2 Experimental Results

The evaluation of friend-safe adversarial examples x^* was divided into two sections—targeted and untargeted adversarial examples—in our experimental results.

Targeted adversarial example

Figure 2 shows an example in which friend-safe adversarial examples x^* generated by a transformer are incorrectly classified as a targeted class by D_{enemy} for each original sample over 1000 iterations with an average distortion of 2.03. In Fig. 2, a human classifies all the friend-safe adversarial examples x^* with their original class.

Table 1 shows the average distortion of each targeted class for the original sample “7” in Fig. 2. The average distortion of this sample differs for each targeted class. For example, targeting class “6” results in the maximum distortion of the “7,” whereas targeting class “2” produces the minimum distortion, as shown in Table 1. The total average distortion of the original “7” sample is approximately 2.18. Figure 6 in the appendix shows the average distortion of each targeted class for each original sample, which can be used in selecting targeted classes in some situations.

Table 2 shows the targeted transformation example “7”→“0” whose classification is determined by the class score. For D_{enemy} , the score of target class “0” (694) is slightly higher than that of the original class (693). For D_{friend} , the score of the original class, “7,” is much higher than the scores of other classes. From this result, we know that the transformation is minimized to the extent that the target class score is slightly higher than the score of the original class while maintaining low distortion rates.

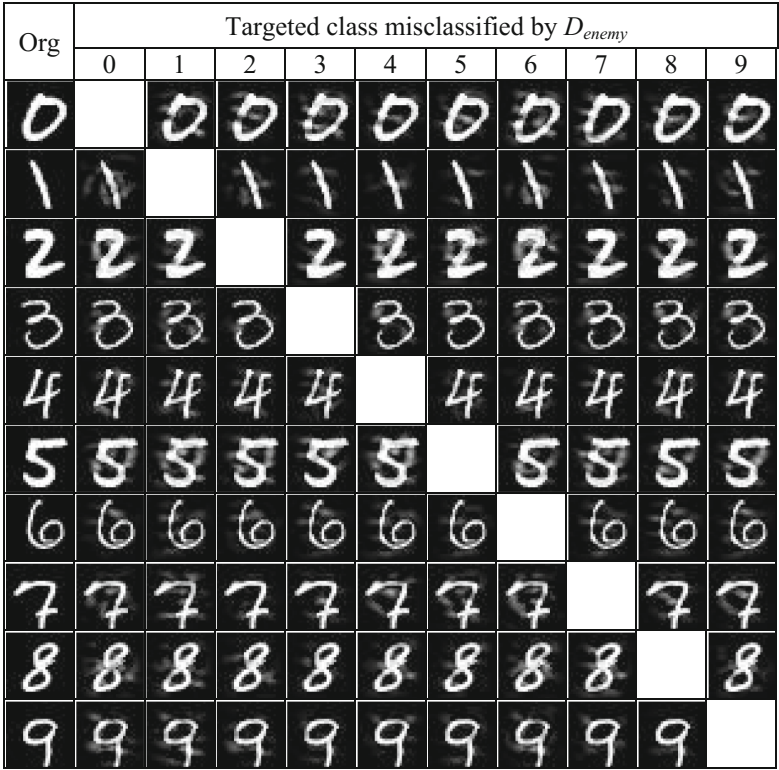


Fig. 2. Adversarial sample for each target class for each original sample

Table 1. Adversarial sample and distortion of an original “7” for each target class in Fig. 2

Org	Targeted classes misclassified by D_{enemy}									
	0	1	2	3	4	5	6	7	8	9
7	7	7	7	7	7	7	7	7	7	7
Rate	2.299	2.676	1.043	1.547	2.443	2.696	3.027	-	1.898	2.041

Table 2. Class score of adversarial example: “7” → “0” and in Table 1

Description	Original (“7”)	Friend-safe adversarial example	
		D_{enemy} (“0”)	D_{friend} (“7”)
Sample image			
Class score	[00000001000]	[694 -225 692 -262 -319 -533 -376 693 -142 -30.1]	[-1.37 1.18 7.12 -0.38 0.43 -7.4 -6.48 10.5 -4.16 -3.76]

Figure 3 shows the targeted attack success rate, D_{friend} accuracy, and average distortion of 1000 examples. As the number of iterations increases, the targeted attack success rate and D_{friend} accuracy increase and the average distortion decreases. When the targeted attack success rate count exceeds 500, D_{friend} accuracy and the targeted attack success rate reach 100%. At this point, the average distortion is less than 2.183.

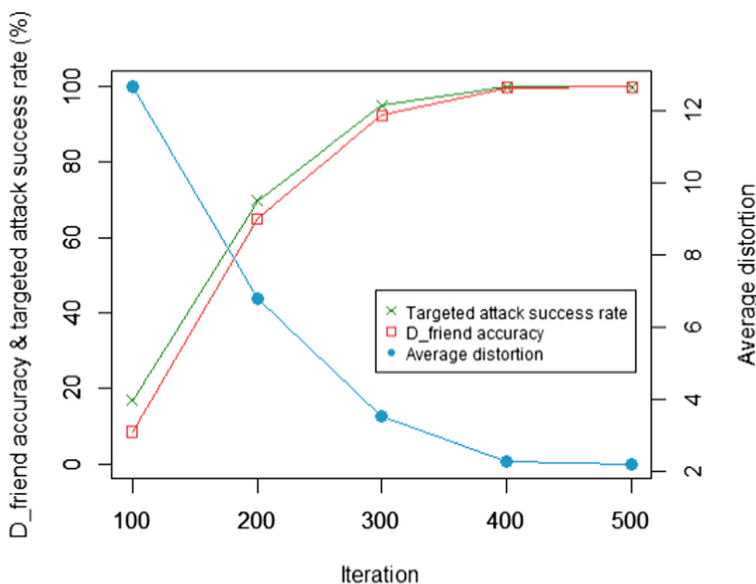


Fig. 3. Targeted attack success rate and D_{friend} accuracy, along with the average distortion, for each number of iterations tested

The attack success rate increases more quickly than the friendly classifier’s accuracy, meaning that it requires more time to generate examples that are correctly classified by a friendly classifier. Table 3 shows the iterative process of generating an example image. We think that because the image is generated from a zero matrix (black image), it is easier to make an enemy result incorrect than to make a friendly result correct.

Table 3. Images of the friend-safe adversarial example for the iterations shown in Fig. 3

Iteration	100	200	300	400	500
Image					

Untargeted adversarial example

Table 4 shows the confusion matrix of the untargeted adversarial example classified by D_{enemy} , testing 100 untargeted adversarial examples per original sample. Transformation mainly affects a few specific classes when a target class is not given. Transformation is made to any classes other than original class that requires minimal modification.

Table 4. Confusion matrix of D_{enemy} for an untargeted class (400 iterations)

Original class	Output class									
	0	1	2	3	4	5	6	7	8	9
0	0	0	11	2	2	7	24	15	4	35
1	0	0	1	1	46	1	0	11	40	0
2	6	26	0	29	1	0	2	25	11	0
3	0	4	14	0	1	57	0	19	5	1
4	1	13	7	0	0	1	6	7	7	58
5	0	0	0	38	0	0	6	0	18	38
6	16	1	1	0	13	63	0	0	6	0
7	0	18	9	21	4	0	0	0	1	47
8	7	2	13	42	2	18	3	3	0	10
9	0	0	0	7	37	2	0	30	24	0

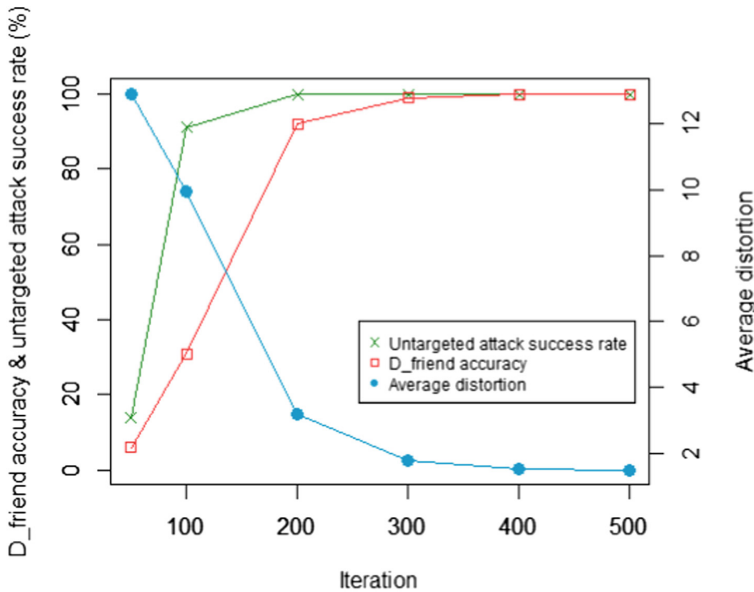


Fig. 4. Untargeted attack success rate, D_{friend} accuracy, and average distortion various numbers of iterations

Figure 4 shows the untargeted attack success rate, D_{friend} accuracy, and average distortion for several iteration counts. As in Fig. 3, as the number of iterations increases, the untargeted attack success rate and D_{friend} accuracy increase and the average distortion decreases. When the iteration count exceeds 400, D_{friend} accuracy and the untargeted attack success rate reach 100%. At this point, the average distortion is less than 1.536. The attack success rate saturates much faster than the friendly classifier’s accuracy; this difference is larger than in targeted attacks. Hence, the lack of target restrictions allows faster successful attacks.

Table 5 shows the iteration count and distortion that are required to achieving 100% accuracy in each case. The untargeted examples reach 100% faster than the targeted examples, and distortion in the untargeted case is also smaller than in the targeted case for the same iteration count. In both cases, the attack success rate is faster than the friend’s accuracy. We discuss the implications of this result in the next section.

Table 5. Comparison between targeted and untargeted attacks when the success rate is 100%

Description	Targeted adversarial example		Untargeted adversarial example	
	Attack success rate	D_{friend} accuracy	Attack success rate	D_{friend} accuracy
Iteration count	500	500	300	400
Max distortion	6.645	6.645	4.016	3.440
Min distortion	0.232	0.232	0.249	0.234
Mean distortion	2.183	2.183	1.788	1.536

5 Discussion

We show that it is possible to generate an adversarial example that achieves a 100% attack success rate and 100% accuracy by friendly classifiers simultaneously. When both classifiers are more than 99% accurate, this is possible because the enemy and

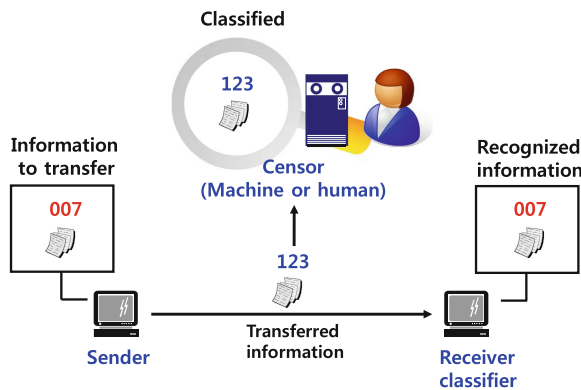


Fig. 5. New covert channel scheme using a friend-safe adversarial example

friendly classifiers are different. It is impossible to generate such examples if the two are exactly same. In the experiments in Sect. 4, the enemy uses a distilled classifier and the friend employs a general CNN classifier. To study the possibility of generating adversarial examples with two very similar models, we tested the same classifier configuration for both the friend and enemy, and provided the same training data with different sample order. With this setup, we found the same results: a friend-safe adversarial example with a 100% attack success rate and 100% accuracy from friendly classifiers (see Figs. 7 and 8 in the Appendix).

From Table 5, we found that untargeted attacks required less distortion, and are ideal for when targeting is unnecessary or minimizing distortion is important. In untargeted attacks, the attacker could estimate the probability of the to-be-recognized class from Table 4. For example, when the original class was “9,” “4,” “7,” and “8” have high probabilities of enemy’s recognition. In cases in which a specific target is not necessary and minimizing distortion is important, but the attacker wants to know the victim (enemy’s) classification result, the attacker can refer to Fig. 6 in the Appendix and select a target class that has less distortion. For example, if an attacker wants to make the victim recognize a road sign with the digit “9” as something other than “9”, then he could select “7” as a target class, because “9” \rightarrow “7” requires the least distortion. In this case, he knows that the victim will recognize the road sign as “7” the target class that the attacker selected.

As mentioned in the introduction, a friend-safe adversarial example is used as an evasion attack against an enemy in a mixed battle field. In addition to this application, we discovered an interesting covert channel scheme, shown in Fig. 5. In this scheme, the roles of the friend and enemy are reversed. The sender makes an example that is correctly recognized by a machine or human censor (enemy) and incorrectly classified by the receiver (friend). The target class is hidden information that is transferred via the covert channel.

6 Conclusion

In this paper, we proposed a new friend-safe adversarial example that will be incorrectly classified by D_{enemy} and correctly classified by D_{friend} , while minimizing distortion of the original sample. In experimental results on MNIST data, D_{friend} correctly classified transformed examples as the original class with 100% accuracy, and the attack success rate was 100% in both targeted and untargeted attacks, when the data distortion was 2.183 and 1.536, respectively. We discovered that distortion differs between target digit classes; this information is useful for selecting a targeted class. We also presented applications of the proposed scheme: a mixed battle field and a covert channel.

Future research will extend our experiments to other standard image datasets, such as CIFAR and ImageNet. We will also work on generating friend-safe adversarial example not through transformation, but by applying a generative scheme, as in a generative adversarial network [26, 27]. Evaluation and analysis of the covert channel that we proposed in our discussion of this study would be another interesting issue. Finally, developing a counter measure to the proposed scheme will be another challenge.

Acknowledgment. This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2017-0-00380, Development of next generation user authentication and No. 2016-0-00173, Security Technologies for Financial Fraud Prevention on Fintech).

Appendix

See Table 8.



Table 6. D_{friend} and D_{enemy} model architecture

Layer type	MNIST data shape
Convolution+ReLU	[3, 3, 32]
Convolution+ReLU	[3, 3, 32]
Max pooling	[2, 2]
Convolution+ReLU	[3, 3, 64]
Convolution+ReLU	[3, 3, 64]
Max pooling	[2, 2]
Fully connected+ReLU	[200]
Fully connected+ReLU	[200]
Softmax	[10]

Table 7. D_{friend} and D_{enemy} model parameters

Parameter	Value
Batch size	128
Dropout	0.5
Momentum	0.9
Learning rate	0.1
Epochs	50

Table 8. Untargeted class of adversarial example safe for friend in D_{enemy}

Description	Original (“1”)	Friend-safe adversarial example	
		D_{enemy} (“8”)	D_{friend} (“1”)
Sample image			
Class score	[0 1 0 0 0 0 0 0 0]	[-441 1161 -388 -37.3 -93.1 -186 -69.5 -459 1164 -245]	[-6.92 22.9 -2.62 -5.18 -0.8 3.34 -3.25 -4.27 1.13 -6.5]

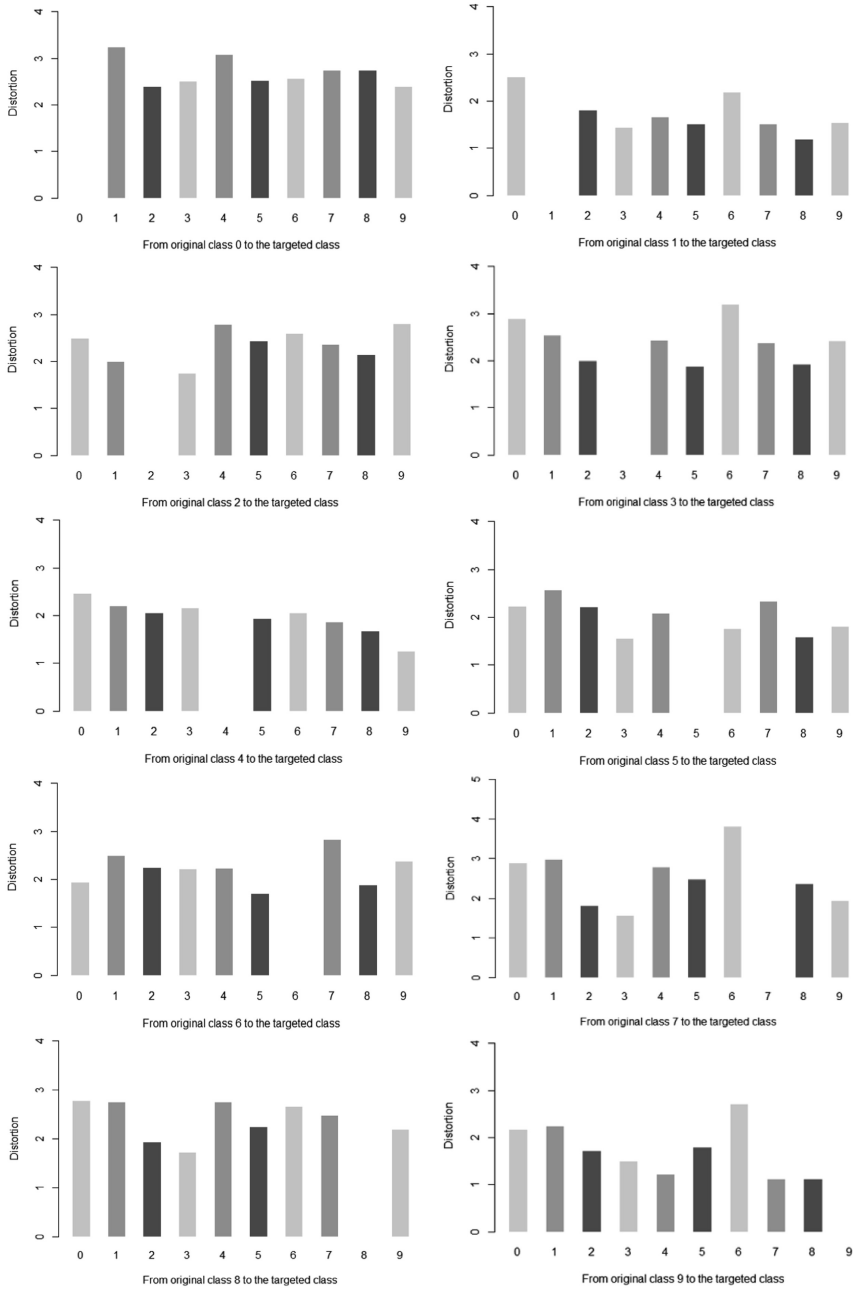


Fig. 6. Average distortion of the targeted class for each of the original classes 0–9 (380 iterations)

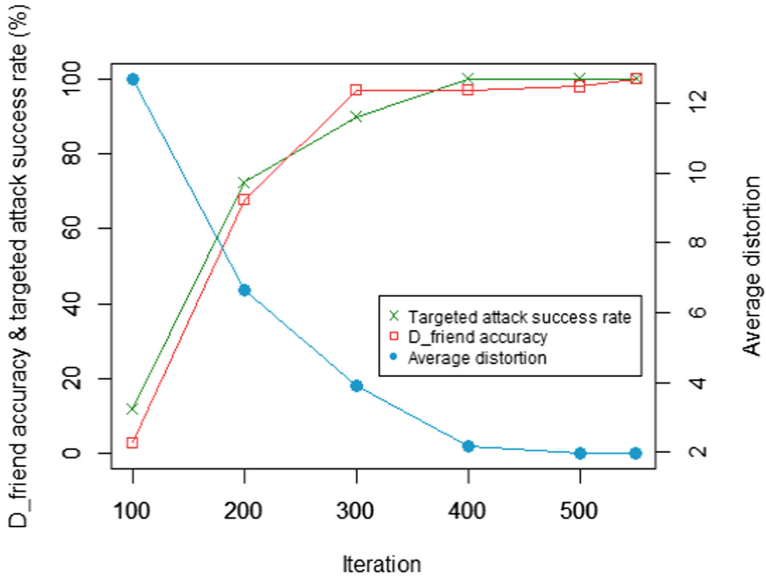


Fig. 7. Targeted attack success rate and D_{friend} accuracy, and the average distortion per iteration, in both models

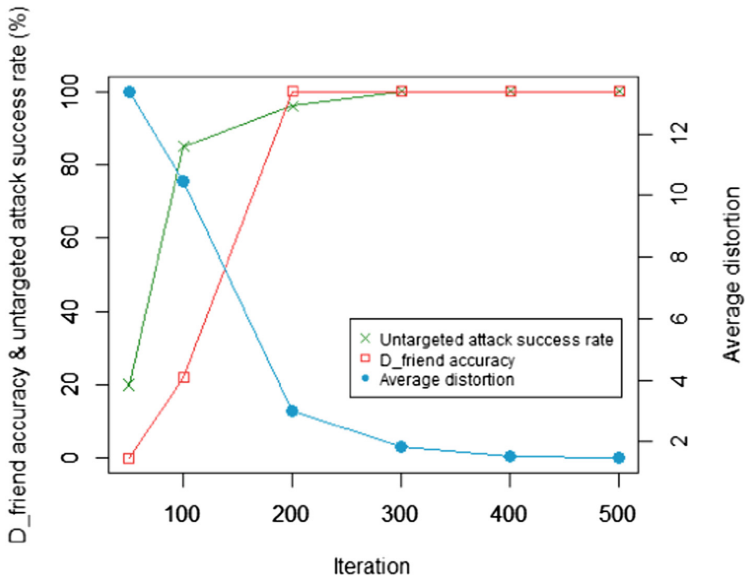


Fig. 8. Untargeted attack success rate and D_{friend} accuracy, as well as the average distortion per iteration, in both models

References

1. Schmidhuber, J.: Deep learning in neural networks: an overview. *Neural Netw.* **61**, 85–117 (2015)
2. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014)
3. Hinton, G., et al.: Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. *IEEE Sig. Process. Mag.* **29**(6), 82–97 (2012)
4. Potluri, S., Diedrich, C.: Accelerated deep neural networks for enhanced Intrusion Detection System. In: 2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (2016)
5. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of the 25th International Conference on Machine Learning (2008)
6. Silver, D., et al.: Mastering the game of Go with deep neural networks and tree search. *Nature* **529**(7587), 484–489 (2016)
7. Szegedy, C., et al.: Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013)
8. McDaniel, P., Papernot, N., Celik, Z.B.: Machine learning in adversarial settings. *IEEE Secur. Privacy* **14**(3), 68–72 (2016)
9. LeCun, Y., Cortes, C., Burges, C.J.C.: MNIST handwritten digit database. AT&T Labs, vol. 2. <http://yann.lecun.com/exdb/mnist> (2010)
10. Papernot, N., et al.: Distillation as a defense to adversarial perturbations against deep neural networks. In: 2016 IEEE Symposium on Security and Privacy (2016)
11. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: 2017 IEEE Symposium on Security and Privacy (2017)
12. Smeets, M., Koot, M.: Covert Channels. Research Report for RPI University of Amsterdam MSc in System and Network Engineering (2006)
13. Barreno, M., et al.: The security of machine learning. *Mach. Learn.* **81**, 121–148 (2010)
14. Biggio, B., Nelson, B., Laskov, P.: Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389* (2012)
15. Mozaffari-Kermani, M., et al.: Systematic poisoning attacks on and defenses for machine learning in healthcare. *IEEE J. Biomed. Health Inform.* **19**(6), 1893–1905 (2015)
16. Yang, C., et al.: Generative poisoning attack method against neural networks. *arXiv preprint arXiv:1703.01340* (2017)
17. Papernot, N., et al.: The limitations of deep learning in adversarial settings. In: 2016 IEEE European Symposium on Security and Privacy (2016)
18. Moosavi-Dezfooli, S.-M., Fawzi, A., Frossard, P.: Deepfool: a simple and accurate method to fool deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
19. Biggio, B., Fumera, G., Roli, F.: Security evaluation of pattern classifiers under attack. *IEEE Trans. Knowl. Data Eng.* **26**(4), 984–996 (2014)
20. Cortes, C., Vapnik, V.: Support vector machine. *Mach. Learn.* **20**(3), 273–297 (1995)
21. Kleinbaum, D.G., Klein, M.: Introduction to Logistic Regression, pp. 1–39. Springer, New York (2010). https://doi.org/10.1007/978-1-4419-1742-3_1
22. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014)
23. Abadi, M., et al.: Tensorflow: large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467* (2016)

24. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998)
25. Kingma, D., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)
26. Goodfellow, I., et al.: Generative adversarial nets. In: *Advances in Neural Information Processing Systems* (2014)
27. Odena, A., Olah, C., Shlens, J.: Conditional image synthesis with auxiliary classifier GANs. arXiv preprint [arXiv:1610.09585](https://arxiv.org/abs/1610.09585) (2016)

Author Index

- Acharya, Kamalesh 191
Arita, Seiko 112
Assidi, Hafsa 267
Ayebie, Edoukou Berenger 267
- Cheng, Chen-Mou 245
Choi, Daeseon 351
- Duquesne, Sylvain 231
Dutta, Ratna 191
- Eom, Jieun 214
- Han, Dong-Guk 139
Handa, Sari 112
He, Yan 59
- Ikuta, Ken 231
- Joichi, Sho 231
Jung, Younghoon 3
- Khandaker, Md. Al-Amin 231
Kim, Howon 175
Kim, Hyeonjin 3
Kim, Joon Sik 214
Kim, Kee Sung 39
Kim, Minkyu 39
Kim, Woo-Hwan 39
Kodera, Kenta 245
Koo, Bonwook 3
Koo, Namhun 310
Kuchta, Veronika 71, 284
Kusaka, Takuya 231
Kwon, Daesung 3
Kwon, Hyeokchan 175
Kwon, Hyun 351
- Lee, Dong Hoon 214
Lee, Dong-Geon 3
Lee, Dongsoo 39
Lee, Sokjoon 175
Lee, Wang Hao 327
Lee, Youngkyung 214
- Lemke-Rust, Kerstin 155
Li, Yan 327
Lin, Dongdai 93
Liu, Renzhang 93
Liu, Zhe 175, 327
- Markowitch, Olivier 71, 284
Meng, Weizhi 327
Mimura, Mamoru 339
Miyaji, Atsuko 245
- Nogami, Yasuyuki 231
- Park, Aesun 139
Park, Cheol-Min 310
Park, Je Hong 39
Park, Taehwan 175
- Roh, Dongyoung 3
- Sahu, Rajeev Anand 71, 284
Samarin, Peter 155
Seo, Hwajeong 175
Sharma, Gaurav 71, 284
Shim, Kyung-Ah 310
Souidi, El Mamoun 267
Su, Chunhua 327
- Tanaka, Hidema 339
Tolba, Mohamed 26
- Uehara, Satoshi 231
- Won, Yoo-Seung 139
- Xu, Shuaijianni 59
- Yamai, Nariyoshi 231
Yoon, Hyunsoo 351
Youssef, Amr M. 26
- Zhang, Liang Feng 59
Zhang, Yang 93